Ionic 2

Table des matières

1. EDI 3
2. Installation 3
3. Création de projet 3
4. Lancer l'application dans le navigateur 3
5. Root page 4
6. Créer une page 4
7. Navigation 4
   a. Naviguer vers une autre page 5
      Avec **directive navPush** 5
   b. Passage de paramètre 5
      Ou avec **directive navPush + navParams** 5
      Récupération de paramètre avec **NavParams** 5
   c. Back 6
      Avec directive navPop 6
   d. Retour à la page d'accueil 6
   e. Création de page 6
   f. Page transition 6
8. Page lifecyle 7
9. Tab Page 8
   Passage de paramètre avec la directive rootParams 9
10. Sidemenu 9
11. Modal 11
    a. Passer des données à une « modal » 12
    b. Réaction à la fermeture de la modal 12
12. Astuce créer des index pour les pages, services, models, etc. 13
13. Components Ionic 13
    a. List 13
       Virtual scroll 14
       Bouton delete dans liste 14
       Naviguer vers la page detail d'un item 16
    b. Grid 16
    c. Toast 16
    d. Alert 17

14. Components....18
15. Forms....18
16. Services / Models....19
17. Ionic Native....19
18. Deploiement....20
a. Icon et splash screen....20
b. config.xml....20
c. Publishing....20

1. EDI

- Visual Studio Code. Extensions : snippets Ionic 2
- WebStorm

2. Installation

Documentation, commandes CLI

```
npm i ionic cordova -g
```

3. Création de projet

Exemple création d'un projet nommé « ionicdemo »

```
ionic start <project-name> blank --v2
```

Types de projets de départ :

- **blank**
- **tabs**
- **sidemenu**
- **tutorial**

Projet avec exemples :

```
ionic start ionicdemo --v2
```

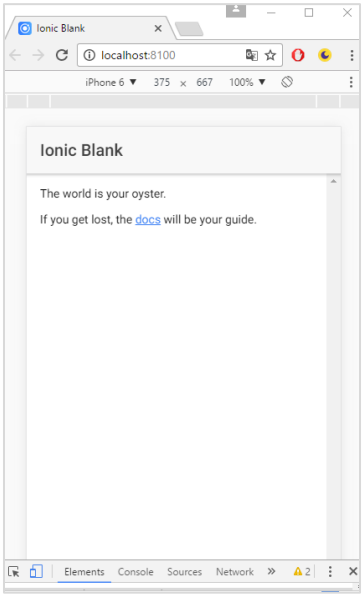
4. Lancer l'application dans le navigateur

(utile pour la phase de développement)

```
ionic serve
```

Se rendre à http://localhost:8100/

On peut utiliser l'émulation avec **Chrome**



5. Root page

Définie dans **AppComponent** avec « **rootPage** »

```
import { Component } from '@angular/core';
import { Platform } from 'ionic-angular';
import { StatusBar, Splashscreen } from 'ionic-native';

import { HomePage } from '../pages/home/home';

@Component({
  templateUrl: 'app.html'
})
export class MyApp {
  rootPage = HomePage;

  constructor(platform: Platform) {
    platform.ready().then(() => {
      StatusBar.styleDefault();
      Splashscreen.hide();
    });
  }
}
```

```
<ion-nav [root]="rootPage"></ion-nav>
```

On peut définir n'importe quelle page en root (exemple une tab page)

6. Créer une page

```
ionic generate page <page-name>
```

Ou

```
ionic g page <page-name>
```

Note n'indiquer que « home » pour avoir une page nommée « HomePage » par exemple. Le suffixe est ajouté automatiquement.

Enregistrer la page crée dans les **déclarations** et les **entryComponents** d'**AppModule**

7. Navigation

navController, navParams

- Ionic n'utilise pas le router d'Angular
- Ionic fonctionne avec un **stack** de « **pages** » (root).
- On peut avoir plusieurs stacks (tabs)
- Une page est component (qui peut contenir d'autres components)

a. Naviguer vers une autre page

Ajout d'un bouton et d'un event click

```
<button ion-button (click)="onGoPageOne()">Go page 1</button>
```

```
onGoPageOne() {
     this.navCtrl.push(PageOnePage);
}
```

Avec **directive navPush**

```
<button ion-button
  [navPush]="pageOne"
  [navParams]="{myParam:'my param value'}">Go page 1</button>
```

b. Passage de paramètre

```
<button ion-button (click)="onGoPageTwo()">Go page 2</button>
```

On peut passer tout type de données (exemple string, object)

```
  onGoToPageTwo() {
    // this.navCtrl.push(PageTwoPage, 'My parameter');
    // or object
    this.navCtrl.push(PageTwoPage, { myParam: 'My parameter value' });
  }
```

Ou avec **directive navPush + navParams**

```
<button ion-button
  [navPush]="pageTwo"
  [navParams]="{myParam:'my param value'}">Go page 2</button>
```

Récupération de paramètre avec **NavParams**

```
import { Component, OnInit } from '@angular/core';
import {  NavParams } from 'ionic-angular';

@Component({
  selector: 'page-two',
  templateUrl: 'page-two.html'
})
export class PageTwoPage implements OnInit {
  param: string;
  constructor(public navParams: NavParams) { }

  ngOnInit() {
    // this.param = this.navParams.data;
    // with object
    this.param = this.navParams.get('myParam');
  }
```

```
}
```

c. Back

```
<button ion-button (click)="onGoBack()">Go back</button>
```

Avec **NavController** (à injecter dans le constructor)

```
import { NavController } from 'ionic-angular';
```

```
onGoBack() {
    if (this.navCtrl.canGoBack()) {
      this.navCtrl.pop();
    }
  }
```

Avec directive navPop

```
<button ion-button navPop>Go back</button>
```

d. Retour à la page d'accueil

```
 <button ion-button (click)="onGoToHome()" icon-left><ion-
icon name="home"></ion-icon>Go Home</button>
```

Avec **NavController**

```
onGoToHome() {
    this.navCtrl.popToRoot();
  }
```

e. Création de page

```
ionic generate page page-one
```

Un dossier du nom de la page avec le template html, une feuille de styles et le component est ajouté au dossier « pages »

f. Page transition

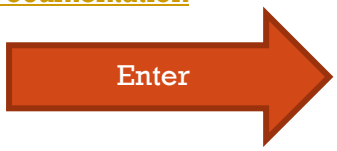
Documentation

Exemple

```
  onGoToPageOne() {
     this.navCtrl.push(PageOnePage, {}, {
       direction: 'back',
       duration: 1000,
       easing: 'ease-out'
     });
  }
```

8. Page lifecyle

Documentation



- ionViewCanEnter (boolean ou promise) route enter guard
- ionViewDidLoad (pas exécuté si page cachée)
- ionViewWillEnter la page va être activée
- ionViewDidEnter la page est active

- ionViewCanLeave (boolean ou promise) route leave guard
- ionViewWillLeave la page va être déactivée
- ionViewDidLeave la page est déactivée
- ionViewDidUnload la page va être détruite

Exemple de code de page

```
import { Component } from '@angular/core';
import { NavController, NavParams } from 'ionic-angular';

@Component({
  selector: 'page-page-liflecycle',
  templateUrl: 'page-liflecycle.html'
})
export class PageLiflecyclePage {

  constructor(public navCtrl: NavController, public navParams: NavParams) {
}

  ionViewCanEnter(): boolean | Promise<boolean> {
    console.log('ionViewCanEnter');
    // navigation enter guard
    return true;
  }

  ionViewDidLoad() {
    console.log('ionViewDidLoad');
  }

  ionViewWillEnter() {
    console.log('ionViewWillEnter');
  }
```

```
  ionViewDidEnter() {
    console.log('ionViewDidEnter');
  }

  ionViewCanLeave(): boolean | Promise<boolean> {
    console.log('ionViewCanLeave');
    // navigation leave guard
    return new Promise((resolve) => {
      setTimeout(function () {
        resolve(true);
      }, 2000);
    });
  }

  ionViewWillLeave() {
    console.log('ionViewWillLeave');
  }

  ionViewDidLeave() {
    console.log('ionViewDidLeave');
  }

  ionViewWillUnload() {
    console.log('ionViewWillUnload');
  }
}
```

9. Tab Page

Exemple création d'une page nommée « tabs »

```
ionic generate page tabs
```

Ajout de la tabs page aux **declarations** et **entryComponents** d'**AppModule**

On peut remplacer la root page par cette tabs page dans AppComponent

```
export class MyApp {
  rootPage = TabsPage;
```

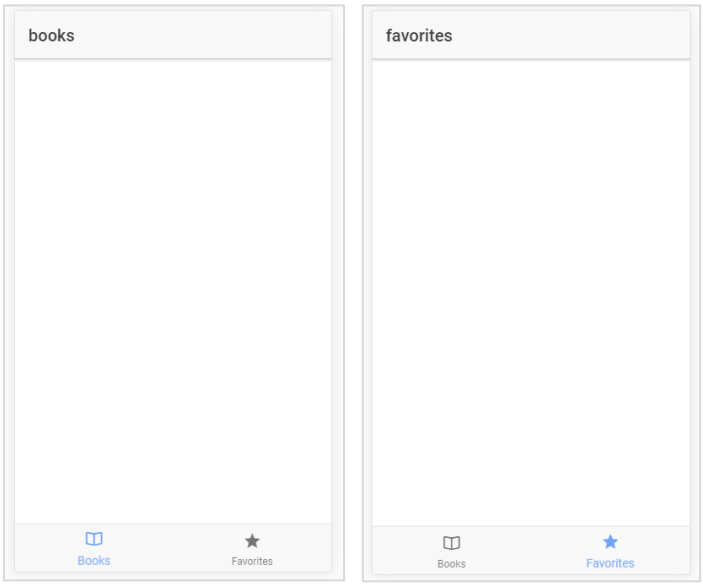
Template de la tabs page. Chaque tab a son propre stack.

```
<ion-tabs>
  <ion-tab tabTitle="Books" [root]="booksPage" tabIcon="book"></ion-tab>
  <ion-tab tabTitle="Favorites" [root]="favoritesPage" tabIcon="star"></ion-
tab>
</ion-tabs>
```

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

```
import { BooksPage } from '../books/books';

@Component({
  selector: 'page-tabs',
  templateUrl: 'tabs.html'
})
export class TabsPage {
  booksPage = BooksPage;
  favoritesPage = FavoritesPage;
}
```



Passage de paramètre avec la directive rootParams

Exemple passage de paramètre à la page « books »

```
<ion-tab tabTitle="Books" [root]="booksPage"
[rootParams]="{myParam:'my value'}" tabIcon="book"></ion-tab>
```

On utilise **NavParams** pour récupérer le paramètre passé

```
ionViewDidLoad() {
    console.log(this.navParams.get('myParam'));
}
```

10. Sidemenu

Création du menu dans **AppComponent**

```
<ion-menu [content]="nav">
  <ion-header>
    <ion-toolbar>
      <ion-title>Menu</ion-title>
    </ion-toolbar>
  </ion-header>
  <ion-content>
    <ion-list>
      <button ion-item (click)="onLoad(tabsPage)">
        <ion-icon name="quote" item-left></ion-icon>
        Home
      </button>
      <button ion-item (click)="onLoad(settingsPage)">
        <ion-icon name="settings" item-left></ion-icon>
        Settings
      </button>
    </ion-list>
  </ion-content>
</ion-menu>
<ion-nav [root]="tabsPage" #nav></ion-nav>
```

```
import { Component, ViewChild } from '@angular/core';
import { MenuController, NavController, Platform } from 'ionic-angular';
import { StatusBar, Splashscreen } from 'ionic-native';
import { TabsPage } from '../pages/tabs/tabs';
import { SettingsPage } from '../pages/settings/settings';

@Component({
  templateUrl: 'app.html'
})
export class MyApp {
  tabsPage = TabsPage;
  settingsPage = SettingsPage;
  @ViewChild('nav') nav: NavController;
  constructor(platform: Platform, private menuCtrl: MenuController) {
    platform.ready().then(() => {
      StatusBar.styleDefault();
      Splashscreen.hide();
    });
  }
  onLoad(page: any) {
    this.nav.setRoot(page);
    this.menuCtrl.close();
  }
}
```

Pages switchées + accès par la référence définie sur « ion-nav »

On switch de root page et on ferme le menu

Ajout du bouton de menu sur chaque page switchée. Exemple

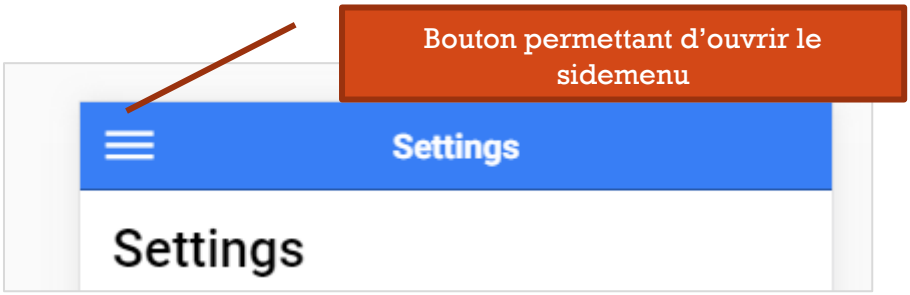
```
<ion-header>
```

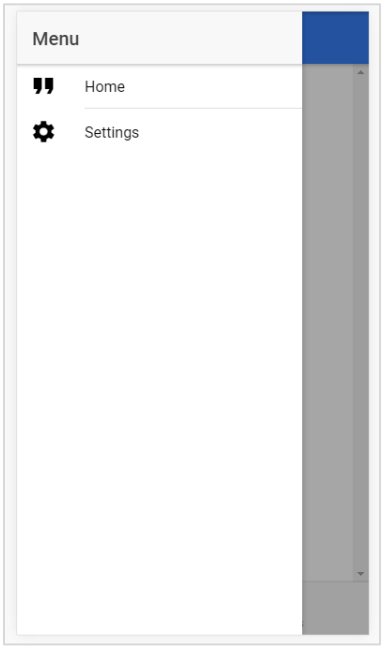
La directive menuToggle ouvre/ ferme le sidemnu

```
  <ion-navbar color="primary">
    <button menuToggle ion-button icon-only>
            <ion-icon name="menu"></ion-icon>
    </button>
    <ion-title>Settings</ion-title>
  </ion-navbar>
</ion-header>

<ion-content padding>
  <h1>Settings</h1>
</ion-content>
```

⇨ Tabs page Avec Sidemenu : ajouter le bouton de menu sur chaque page affichée par la tab (pas sur la « tabs » page)





11. Modal

Documentation

- Créer une page quelconque.
- Utiliser « **ModalController** » pour afficher cette page en tant que modal

```
import {  ModalController } from 'ionic-angular';
```

Exemple au clic sur un bouton

```
onShowModal() {
    const modal = this.modalCtrl.create(MyModalPage);
    modal.present();
  }
```

Cacher la « modal »

Exemple :ajout d'un bouton dans le header

```
<ion-header>
  <ion-navbar>
    <ion-title>Modal</ion-title>
    <ion-buttons start>
      <button ion-button (click)="onClose()">Close</button>
    </ion-buttons>
  </ion-navbar>
</ion-header>
```

+ Utilisation du **ViewController** pour fermer la modal (ViewController permet d'accéder à la page active affichée)

```
import { ViewController } from 'ionic-angular';
```

```
onClose() {
    this.viewCtrl.dismiss();
  }
```

a. Passer des données à une « modal »

```
 onShowModal() {
    const myData = { title: "My data" };
    const modal = this.modalCtrl.create(MyModalPage, myData);
    modal.present();
  }
```

En second paramètre

Récupération avec **NavParams**

```
  ionViewDidLoad() {
    this.title = this.navParams.get('title');
  }
```

b. Réaction à la fermeture de la modal

Passage de paramètre (exemple un boolean) à la fermeture de la modal

```
<button ion-button (click)="onClose(true)">Ok</button>
```

```
onClose(p: any) {
```

```
    this.viewCtrl.dismiss(p);
  }
```

Réaction à la fermeture de la modal depuis une page

```
onShowModal() {
    const modal = this.modalCtrl.create(MyModalPage);
    modal.present();
    modal.onDidDismiss((p) => {
      // do something
      console.log(p);
    });
  }
```

12. Astuce créer des index pour les pages, services, models, etc.

Créer un fichier « pages.ts » par exemple à la racine du dossier pages

```
export { BookPage } from './book/book';
export { BooksPage } from './books/books';
export { FavoritesPage } from './favorites/favorites';
export { SettingsPage } from './settings/settings';
export { TabsPage } from './tabs/tabs';
```

Utilisation plus simple pour les imports (AppModule par exemple)

```
import { BookPage, BooksPage, FavoritesPage, TabsPage, SettingsPage }
from '../pages/pages';
```

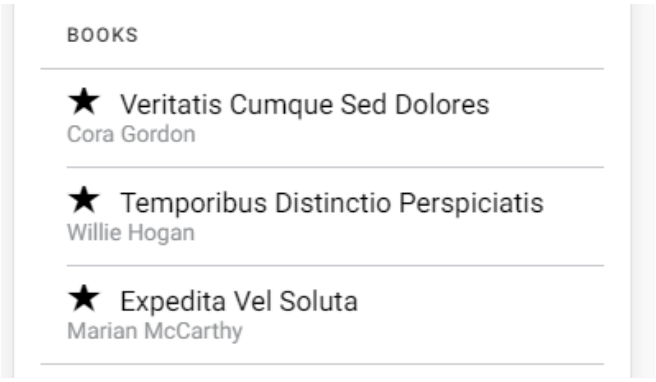
13. Components Ionic

button,toolbar, icons

a. List

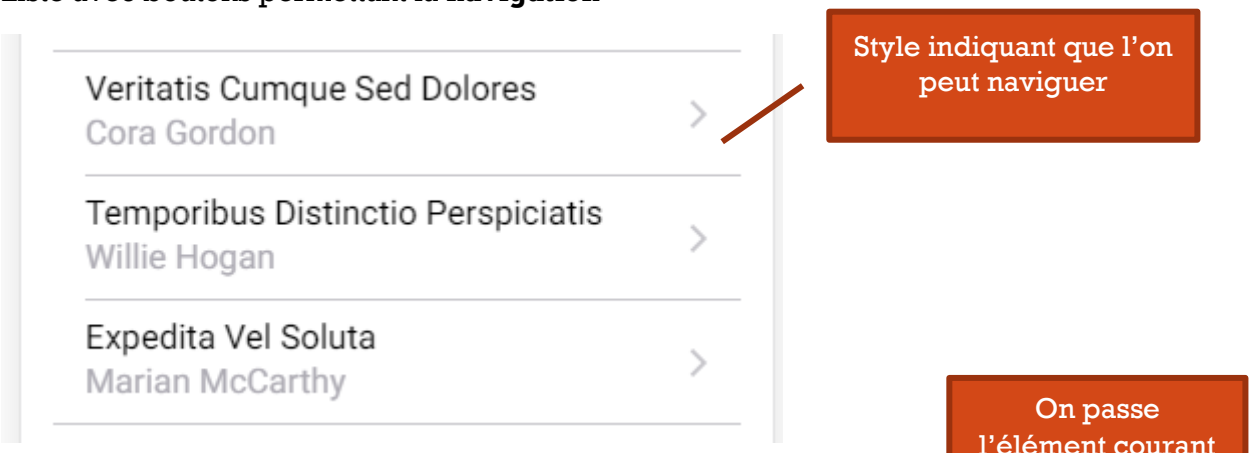
Documentation, badges

Simple liste d'items ou liste de boutons permettant la navigation




```
<ion-list>
  <ion-list-header>Books</ion-list-header>
  <ion-item icon-left *ngFor="let book of books">
      <ion-icon name="star"></ion-icon>
              {{book.title}}
              <p>{{book.author}}</p>
      </ion-item>
</ion-list>
```

Liste avec boutons permettant la **navigation**




```
<ion-list>
  <button ion-item *ngFor="let book of books" (click)="onLoadBook(book)">
    <h2>{{ book.title }}</h2>
    <ion-note>{{ book.author }}</ion-note>
  </button>
</ion-list>
```

On navigue vers la page BokkPage et on passe le paramètre

```
onLoadBook(book) {
  this.navCtrl.push(BookPage, book);
}
```

Divider

Utile pour grouper les éléments

Documentation

Virtual scroll

Documentation

Permet de charger au scrolling, utile par exemple sur les longues listes pour un gain en performance.

Sur ion-list, on indique :

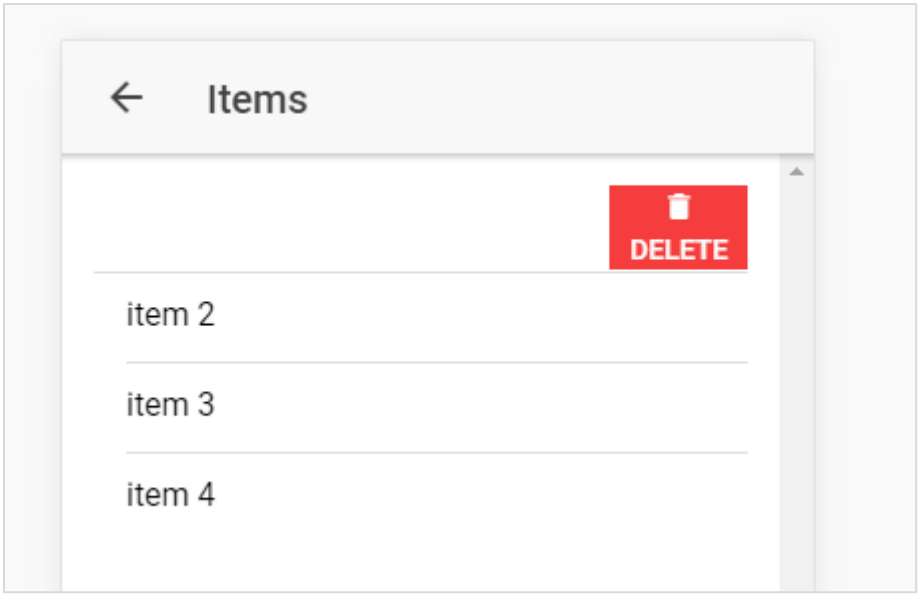
- la source de données avec la directive virtualScroll
- la taille approximative de chaque élément

Sur l'item (ion-item) on indique la variable locale utilisée pour le bionding

```
<ion-list [virtualScroll]="books" [approxItemHeight]="'40px'">
  <button ion-item *virtualItem="let book"  (click)="onLoadBook(book)">
    <h2>{{ book.title }}</h2>
    <ion-note>{{ book.author }}</ion-note>
  </button>
</ion-list>
```

Bouton delete dans liste

Utilisation de **ion-item-sliding** pour afficher le bouton delete




```
<ion-content padding>
  <ion-list>
    <ion-item-sliding *ngFor="let item of items">
      <ion-item>
        <h2>{{item.name}}</h2>
      </ion-item>
      <ion-item-options>
        <button ion-button color="danger" (click)="onRemoveItem(item)">
          <ion-icon name="trash"></ion-icon>
          Delete
        </button>
      </ion-item-options>
    </ion-item-sliding>
  </ion-list>
</ion-content>
```

On passe l'item à supprimer

```
import { Component } from '@angular/core';
import { NavController, NavParams } from 'ionic-angular';

@Component({
  selector: 'page-list',
  templateUrl: 'list.html'
})
export class ListPage {
  items: any[];
  constructor(public navCtrl: NavController, public navParams: NavParams) {
}

  ionViewDidLoad() {
    this.items = [
      { id: 1, name: 'item 1' },
      { id: 2, name: 'item 2' },
```

```
      { id: 3, name: 'item 3' },
      { id: 4, name: 'item 4' },
    ];
  }

  onRemoveItem(item: any) {
    let index = this.items.indexOf(item);
    if (index !== -1) {
      this.items.splice(index, 1);
    }
  }
}
```

Naviguer vers la page detail d'un item

Ajout d'un event click sur le bouton

```
<ion-content padding>
  <ion-list>
    <ion-item-sliding *ngFor="let item of items">
      <ion-item (click)="onViewItem(item)">
```

+ Utilisation du NavController pour afficher la page detail

```
onViewItem(item: any) {
    this.navCtrl.push(ItemPage, item);
  }
```

b. Grid

Documentation

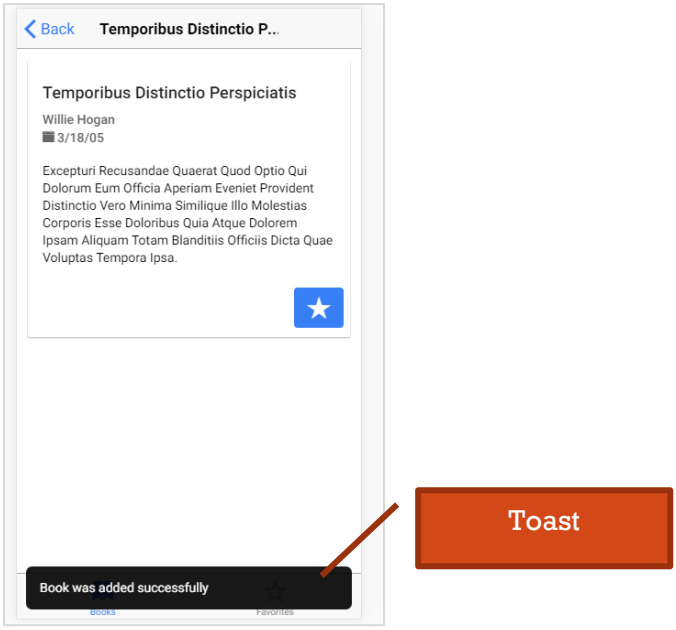
Permet de mettre en forme les éléments, utilisable également dans les controls

c. Toast

Documentation

Exemple après ajout d'un élément à la liste des favoris, on affiche un toast

```
onAddBookToFavorites() {
  this.favoriteService.addBookToFavorites(this.book);
  this.isFavorite = true;
  let toast = this.toastController.create({
    message: 'Book was added successfully',
    duration: 3000
  });
  toast.present();
}
```

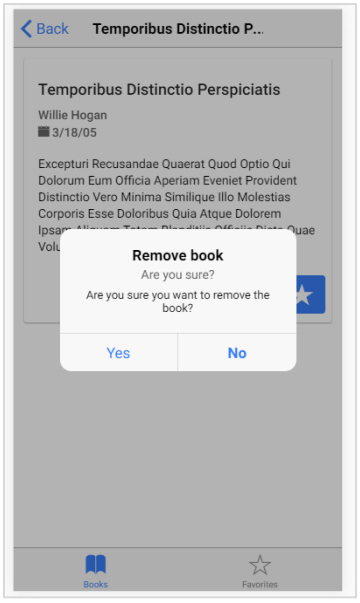


d. Alert

Documentation

Exemple avant supprimer un élément on demande confirmation

```
onRemoveBookFromFavorites() {
  const alert = this.alertController.create({
    title: 'Remove book',
    subTitle: 'Are you sure?',
    message: 'Are you sure you want to remove this book?',
    buttons: [
      {
        text: 'Yes',
        handler: () => {
          if (this.favoriteService.removeBookFromFavorites(this.book)) {
            this.isFavorite = false;
          }
        }
      },
      {
        text: 'No',
        role: 'cancel'
      }
    ]
  });
  alert.present();
}
```



14. Components

On peut créer ses propres components Angular

Par contre il faut les déclarer uniquement dans les déclarations de AppModule (et pas dans les entryComponents)

15. Forms

Avec **Reactive forms**. On **n'a pas à ajouter** le module « ReactiveFormsModule ».

Création du formulaire dans **ngOnInit**

```
 ngOnInit() {
    this.form = new FormGroup({
      'name': new FormControl('', [Validators.required, Validators.minLength(3)]),
      'email': new FormControl('', [Validators.required, Validators.pattern(/^(([^<>()\[\]\\.,;:\s@
9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}])|(([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,}))$/)])
    });
  }
```

Soumission

```
 onSubmit() {
    if (this.form.valid) {
      // save
    }
  }
```

```
<ion-content padding>
  <form [formGroup]="form" (ngSubmit)="onSubmit()">
    <ion-list>
```

```
      <ion-item>
        <ion-label fixed>Name</ion-label>
        <ion-input type="text" formControlName="name"></ion-input>
      </ion-item>
      <ion-item>
        <ion-label fixed>Email</ion-label>
        <ion-input type="text" formControlName="email"></ion-input>
      </ion-item>
      <ion-item>
        <button ion-
button type="submit" [disabled]="!form.valid" block>Add New Friend</button>
      </ion-item>
    </ion-list>
  </form>
</ion-content>
```

Supporte également les forms **Template driven**

16. Services / Models

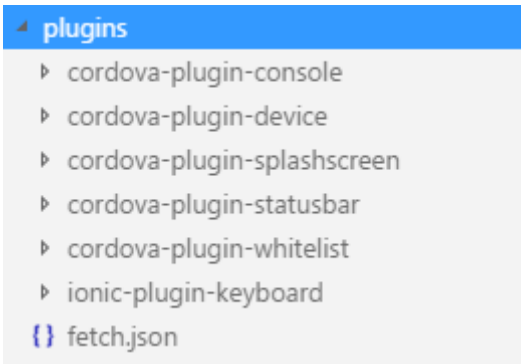
Créer un dossier « services » et un dossier « models » à la racine de « src »

Ajouter les services aux providers d'AppModule.

17. Ionic Native

Documentation

Plugins installés dans le dossier « plugins »



La démarche sera en général d'installer le plugin

```
ionic plugin add <plugin-name>
```

Exemple avec Vibration

```
ionic plugin add cordova-plugin-vibration
```

Et ensuite l'utiliser en l'important

```
import { Vibration } from 'ionic-native';
```

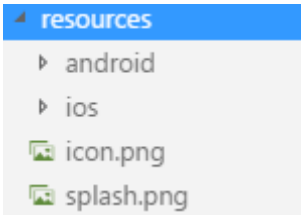
Au clic sur un bouton

```
onVibrate(){
   Vibration.vibrate(2000);
 }
```

Se référer à la documentation pour les différents plugins(Geolocation, Camera,Barcode scanner, Local notifications, etc.)

18. Deploiement

a. Icon et splash screen



On peut remplacer ces assets (en respectant les tailles). Il est possible de définir spécifiquement pour chaque plateforme

Pour générer à partir d'images (png par exemple) les différents assets avec la commande

```
ionic resources
```

b. config.xml

Changer les informations de l'app (nom, la description, etc.)

c. Publishing

Documentation

Android Studio publish your app