無料電子ブック

学習

ionic2

Free unaffiliated eBook created from
Stack Overflow contributors.

#ionic2

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: ionic2

It is an unofficial and free ionic2 ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official ionic2.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

1: ionic2をいめる

Ionic 2はクロスプラットフォームのモバイルです。このフレームワークは、ハイブリッドモバイルアプリケーションをするためにされており、デスクトップアプリケーションにもできます。これはいたもので、どこからでもできます。 JavaScript / Typescript、Angular 2、HTML、CSS SCSS / LESSなどのWebテクノロジーをしています。 Ionic2 appsは`>=android 4.4`でうまくいくが、 `android 4.1`で`android 4.3`するには、 クロスウォークをするがある。

Examples

インストールまたはセットアップ

Ionic 2はくなってきているので、のとをするために、 のをにチェックしてください。

Ion 2プロジェクトをするにはは、NodeJSがです。あなたはダウンロードしてインストールするノードをすることができ、こことNPMとイオン2がするパッケージについてのはこちら 。

---

1. Ionic 2のインストール

Ionic 1とに、Ionic CLIまたはGUIをして、ブラウザのすぐにアプリケーションをしてテストすることができます。それはあなたのIonic 1のアプリケーションですべてのをっているので、あなたはものをするはありません

イオン2をするには、にイオンをnpmからインストールします。

```
$ npm install -g ionic
```

EACCESエラーがしたは、 ここでにってノードになをえてください。

2.のアプリケーションをする

CLIがインストールされたら、のコマンドをしてのアプリをします。

```
$ ionic start MyIonic2Project
```

タブテンプレートはデフォルトでされますが、フラグをしてのテンプレートをすることもできます。えば

```
$ ionic start MyIonic2Project tutorial
```

```
$ cd MyIonic2Project
$ npm install
```

これでチュートリアルテンプレートがされます。

アプリケーションをするには、プロジェクトディレクトリにし、 `ionic serve -lc`をします。

```
$ ionic serve -lc
```

-lはページのライブリロードをにし、-cはコンソールログをします。アプリのにがあるは、package.jsonがionic2-app-baseのものとすることをしてください

しいアプリをブラウザですぐにぶことができます。

3.デバイスへのビルド

また、デバイスやデバイスエミュレータでしいアプリケーションをすることもできます。するには、 コルドバがです。

Cordovaをインストールするには、のコマンドをします。

```
$ npm install -g cordova
```

iOSアプリケーションをするためのiOSシミュレータのドキュメントOSXのOSのiOSデバイスやエミュレータにはインストールできません、またはAndroidアプリケーションをするための Genymotionドキュメントをチェックしてください。

iOSで

iOSアプリケーションをするには、OSXコンピュータでするがあります.iOSにできるようにするためには、ココアフレームワークがです。まず、コードをすることでプラットフォームをコードバスにするがあるのコマンド

```
$ ionic cordova platform add ios
```

iOSデバイスにコンパイルするにはXcodeがです。

に、のコマンドでアプリをします

```
$ ionic cordova run ios
```

Androidでする

Androidのはほとんどじです。まず、プラットフォームをします。

```
$ ionic cordova platform add android
```

それから、Androidデバイスにコンパイルできる[Android SDK]をインストールします。 Android SDKにはエミュレータがしていますが、にはいです。 [Genymotion]ははるかにです。インストールがしたら、のコマンドをしてください

```
$ ionic cordova run android
```

ですあなたののIonic 2アプリをしていただきありがとうございます

イオンもライブリロードしています。したがって、あなたがあなたのアプリをし、エミュレータ/デバイスできているをたいは、のコマンドをすることでそれをうことができます

iOSの

```
$ ionic cordova emulate ios -lcs
```

iOS 9.2.2ではライブロードがしないのでしてください。 livereloadでしたいは、をしてconfig.xmlファイルをします。

```
<allow-navigation href="*"/>
```

に、 `<platform name="ios">`

```
<config-file parent="NSAppTransportSecurity" platform="ios" target="*-Info.plist">
  <dict>
    <key>NSAllowsArbitraryLoads</key>
    <true/>
  </dict>
</config-file>
```

アンドロイド

```
$ ionic cordova run android -lcs
```

lライブリロード、のcコンソールログ、およびsサーバーログの。これにより、にエラー/があるかどうかをすることができます。

Windowsビルド

Windowsのプロジェクトをビルドするは、Windowsコンピュータでするがあります。まず、のコマンドをして、ionic2プロジェクトにWindowsプラットフォームをインストールします。

```
$ionic cordova platform add windows
```

に、のコマンドをします。

```
$ionic cordova run windows
```

ブラウザでするには

```
$ionic serve
```

クロムブラウザのクロムブラウザのアドレスバーに

```
chrome://inspect/#devices
```

オンラインでionic2をいめるをむ https://riptutorial.com/ja/ionic2/topic/3632/ionic2をいめる

2: Angularfire2 / Firebaseによるソーシャルログイン

Examples

Angularfire2 / Firebaseのネイティブ Facebookログイン

app.ts

```
import {Component} from '@angular/core';
import {Platform, ionicBootstrap} from 'ionic-angular';
import {StatusBar} from 'ionic-native';
import {LoginPage} from './pages/login/login';
import {FIREBASE_PROVIDERS, defaultFirebase, AuthMethods, AuthProviders, firebaseAuthConfig}
from 'angularfire2';

@Component({
  template: '<ion-nav [root]="rootPage"></ion-nav>'
})

export class MyApp {

  private rootPage: any;

  constructor(private platform: Platform) {
    this.rootPage = LoginPage;

    platform.ready().then(() => {
      // Okay, so the platform is ready and our plugins are available.
      // Here you can do any higher level native things you might need.
      StatusBar.styleDefault();
    });
  }
}

ionicBootstrap(MyApp, [
  FIREBASE_PROVIDERS,
  defaultFirebase({
    apiKey: myAppKey,
    authDomain: 'myapp.firebaseapp.com',
    databaseURL: 'https://myapp.firebaseio.com',
    storageBucket: 'myapp.appspot.com',
  }),
  firebaseAuthConfig({})
]);
```

login.html

```
<ion-header>
  <ion-navbar>
    <ion-title>Home</ion-title>
  </ion-navbar>
</ion-header>
```

```
<ion-content padding class="login">
  <button (click)="facebookLogin()">Login With Facebook</button>
</ion-content>
```

login.ts

```
import {Component} from '@angular/core';
import {Platform} from 'ionic-angular';
import {AngularFire, AuthMethods, AuthProviders} from 'angularfire2';
import {Facebook} from 'ionic-native';

declare let firebase: any; // There is currently an error with the Firebase files, this will
fix it.

@Component({
  templateUrl: 'build/pages/login/login.html'
})
export class LoginPage {

  constructor(private platform: Platform, public af: AngularFire) {

  }

  facebookLogin() {
    Facebook.login(['public_profile', 'email', 'user_friends'])
      .then(success => {
        console.log('Facebook success: ' + JSON.stringify(success));
        let creds =
firebase.auth.FacebookAuthProvider.credential(success.authResponse.accessToken);
        this.af.auth.login(creds, {
          provider: AuthProviders.Facebook,
          method: AuthMethods.OAuthToken,
          remember: 'default',
          scope: ['email']
        }).then(success => {
          console.log('Firebase success: ' + JSON.stringify(success));
        }).catch(error => {
          console.log('Firebase failure: ' + JSON.stringify(error));
        });
      }).catch(error => {
        console.log('Facebook failure: ' + JSON.stringify(error));
      });
  }
}
```

オンラインでAngularfire2 / Firebaseによるソーシャルログインをむ
https://riptutorial.com/ja/ionic2/topic/5518/angularfire2---firebaseによるソーシャルログイン

3: Angularfire2 with Ionic2

き

ここではAngularFire2をし、このリアルタイムデータベースをIonic Appでするをします。

Examples

AngularFireの

まず、あなたのアプリケーションモジュールでファイアモジュールをのようにするがあります

```
  const firebaseConfig = {
  apiKey: 'XXXXXXXXXX',
  authDomain: 'XXXXXXXXXX',
  databaseURL: 'XXXXXXXXXX',
  storageBucket: 'XXXXXXXXXX',
  messagingSenderId: 'XXXXXXXXXX'
};
```

firebaseにサインインしてしいプロジェクトをすることで、このキーをにれることができます。

```
imports: [
    AngularFireModule.initializeApp(firebaseConfig),
    AngularFireDatabaseModule,
    AngularFireAuthModule
  ],
```

AngularFire2をする

あなたのアプリにそれをって、それをインポートする

```
import { AngularFireDatabase } from 'angularfire2/database';
constructor (private _af: AngularFireDatabase) {}
```

このObservable Listをすると、パスのにあるアイテムのリストにアクセスできます。たとえば、root / items / foodがあるは、のようなアイテムをできます。

```
this._af.list('root/items/food');
```

ここにしいアイテムをくだけで、あなたのfirebaseデータベースにされるか、またはアイテムを1つすればデータベースでされます。このようにプッシュしてすることができます

```
this._af.list('root/items/food').push(myItemData);
this._af.list('root/items/food').update(myItem.$key, myNewItemData);
```

あるいは、あなたのベリストからのアイテムさえもできます

```
this._af.list('root/items/food').remove(myItem.$key);
```

オンラインでAngularfire2 with Ionic2をむ https://riptutorial.com/ja/ionic2/topic/10918/angularfire2-with-ionic2

4: InAppBrowser

き

クライアントがモバイルアプリでウェブアプリをくがあるもあります。これは、InAppBrowserをアプリのようにえるようにするためです。わりに、ユーザーがアプリアイコンをタップするとすぐにモバイルでウェブサイト/ウェブアプリをきますのアプリケーションビューをくわりにInAppBrowserをくことができます。

Examples

こののはこのアプリです

このアプリでは、ユーザーがアプリののページをみむわりにアプリアイコンをタップすると、InAppBrowserをきます。だから、らはじウェブサイト/ webappのアプリをているようにえるだろう。

InAppBrowserをするコード

```
  platform.ready().then(() => {
      // Okay, so the platform is ready and our plugins are available.
      // Here you can do any higher level native things you might need.
      var url= "https://blog.knoldus.com/";
      var browserRef = window.cordova.InAppBrowser.open(url, "_self", "location=no",
"toolbar=no");
      browserRef.addEventListener("exit", (event) => {
          return navigator["app"].exitApp();
        }
      );
```

オンラインでInAppBrowserをむ https://riptutorial.com/ja/ionic2/topic/9801/inappbrowser

5: ionic appをionic viewにする

き

ionic viewはモバイルにインストーールするがあるモバイルアプリです。そのため、.apkファイルをせずにアプリをできます。あなたのアプリのIDをすることで、のユーザーはイオンビューをってモバイルであなたのアプリをることもできます。

サイト https : //view.ionic.io/

Examples

あなたのアプリをイオンビューにする

app.ionic.ioでのをするがあります

1. アカウントをするか、あなたのイオンアカウントにログインしてください
2. ダッシュボードで[しいアプリ]をクリックし、アプリのをします

```
I named my app as 'MyIonicApp'
```

3. このしくされたアプリのセクションには、アプリのにIDがされます。

```
MyIonicApp ID is 4c5051c1
```

のは**Node.js**コマンドプロンプトでいます

1. することであなたのイオンアカウントにログインする

```
$ ionic login
```

2. アプリケーションフォルダをルートします。
3. あなたのアプリをionic viewにアップロードするには、まずあなたのアプリをionic siteでしたIDとリンクさせるがあります。リンクするには、のコマンドをし、

```
$ ionic link [your-app-id]
```

MyIoincAppの、コマンドは、

```
$ ionic link 4c5051c1
```

のコマンドはMyIonicAppのファイルのapp idをします。

4. リンクがしたら、することによってアプリをアップロードします

```
$ ionic upload
```

アップロードがしたら、でイオンビューをき、アプリをします。

のユーザーは、イオンビューで[アプリをプレビュー]セクションのアプリIDをすることで、アプリをできます。

オンラインでionic appをionic viewにするをむ https://riptutorial.com/ja/ionic2/topic/10542/ionic-appをionic-viewにする

6: Ionic2 CSSコンポーネント

Examples

グリッド

Ionicのグリッドシステムは、Ionicがサポートするすべてのデバイスでサポートされ ているCSSであるflexboxにづいています。グリッドは、グリッド、ロー、およびカラムの3つのユニットでされています。はをりつぶすようにされ、のにわせてサイズがされます。

クラス	
-10	10
-20	20
- 25	25
33	33.3333
50	50
67	66.6666
- 75	75
80	80
-90	90

。

```
    <ion-grid>
      <ion-row>
        <ion-col width-10>This column will take 10% of space</ion-col>
      </ion-row>
    </ion-grid>
```

カード

カードはなコンテンツをするのになであり、アプリケーションのコアデザインパターンとしてにしています。これらは、をしするれたであり、ユーザーになをします。にするがく、のがほとんどないため、くのにとってカードはすぐにパターンになりました。

。

```
<ion-card>
  <ion-card-header>
        Header
  </ion-card-header>
  <ion-card-content>
        The British use the term "header", but the American term "head-shot" the English
simply refuse to adopt.
  </ion-card-content>
</ion-card>
```

オンラインでIonic2 CSSコンポーネントをむ https://riptutorial.com/ja/ionic2/topic/8011/ionic2-cssコンポーネント

7: Visual StudioコードでのIonic 2のセットアップとデバッグ

き

Visual Studioは、Intellisenseとコードのをするオープンソースの IDEです。このIDEは、Ionic、C、C、AngularJs、TypeScript、Androidなどのようなくのをサポートしています。これらのは、VSCodeにをすることでコードをすることができます。 VSCodeをすることにより、なるのコードをしてデバッグすることができます。

Examples

VSCodeのインストール

まず、VSCodeをダウンロードしてインストールするがあります。このVSCodeバージョンは、サイトでダウンロードできます。 VSCodeをダウンロードしたら、それをインストールしてくがあります。

```
Introduction of Extensions in VSCode
```

VSCodeはオープンエディタなので、すべてののエディタをしますが、そののをするコードをします。イオンコードのとには、あなたの**VSCode**に**ionic2-vscode** Extensionをするがあります。 VSCodeエディタのには、5つのアイコンがあり、アイコンのうちもさいアイコンがのためにされています。 ショートカットキー**ctrl + shift + X**をしてできる。

```
Add Extension for Ionic2 in VsCode
```

ctrl + shift + Xをすと、の**3**つのドットがされるがされます。これらのドットは、よりくのアイコンとしてられています。クリックすると、ダイアログがき、するオプションのがされます。にしてオプションをしますが、すべてのをするには、 をしてください`(ionic2-vscode),npm(ionic2-vscode),npm`インストールできるすべてののリスト`(ionic2-vscode),npm`

VSCodeでIonicプロジェクトをしてする

VsCodeはコードエディタであるため、イオンプロジェクトをできません.**CLI**または**cmd**をしてイオンプロジェクトをできます。のコマンドでプロジェクトをする

```
$ ionic start appName blank
```

のコマンドはのテンプレートイオンアプリケーションをするためにします。 Ionic2は3のテンプレートブランク、タブ、サイドメニューをします。そう。のテンプレートは、にしての2つのテン

プレートときえることができます。

、あなたのIonicプロジェクトがされました。したがって、プロジェクトをVSCodeにしてすることができます。あなたのプロジェクトをするにはのにってください。

1. VScodeのファイルメニューにします。
2. [ファイルのフォルダをく]メニューをクリックします。
3. プロジェクトフォルダをつけてきます。

    ショートカットキー**ctrl + O**または**Ctrl + k**をしてフォルダをくことができます

、

し、あなたのイオンプロジェクトをデバッグする

> *Chrome*でのとデバッグ

ィオンプロジェクトをするには、ターミナルまたは**cmd**または**CLI**でのコマンドをします。

```
$ ionic serve
```

ionicプロジェクトをデバッグするには、まず、**Debugger for chrome**をし、にlaunch.jsonファイルをこのようにするがあります。

```
{
        "version": "0.2.0",
        "configurations": [

            {
          "name": "Launch in Chrome",
          "type": "chrome",
          "request": "launch",
          "url": "http://localhost:8100",
          "sourceMaps": true,
          "webRoot": "${workspaceRoot}/src"
        }
]
}
```

> ***Android***でのとデバッグ

Androidで**Run** ionicプロジェクトをするには、またはcmdまたはCLIでのコマンドでAndroidプラットフォームをするがあります

```
$ ionic cordova platform add android
```

このコマンドでAndroidをビルドする

```
$ ionic cordova build android
```

アンドロイドプラットフォームのコマンドをする

```
$ ionic cordova run android
```

、あなたのアプリケーションは、のAndroidデバイスでされます。

Androidデバイスへのデバッグでは、VSCodeに**Cordova**または**Android Extension**をするがあります。このようなlaunch.jsonファイルをします。

```
{
    "version": "0.2.0",
    "configurations": [
    {
                "name": "Run Android on device",
                "type": "cordova",
                "request": "launch",
                "platform": "android",
                "target": "device",
                "port": 9222,
                "sourceMaps": true,
                "cwd": "${workspaceRoot}",
                "ionicLiveReload": false
            },
            {
                "name": "Run iOS on device",
                "type": "cordova",
                "request": "launch",
                "platform": "ios",
                "target": "device",
                "port": 9220,
                "sourceMaps": true,
                "cwd": "${workspaceRoot}",
                "ionicLiveReload": false
            },
            {
                "name": "Attach to running android on device",
                "type": "cordova",
                "request": "attach",
                "platform": "android",
                "target": "device",
                "port": 9222,
                "sourceMaps": true,
                "cwd": "${workspaceRoot}"
            },
            {
                "name": "Attach to running iOS on device",
                "type": "cordova",
                "request": "attach",
                "platform": "ios",
                "target": "device",
                "port": 9220,
                "sourceMaps": true,
                "cwd": "${workspaceRoot}"
            },
            {
                "name": "Run Android on emulator",
                "type": "cordova",
                "request": "launch",
```

```
            "platform": "android",
            "target": "emulator",
            "port": 9222,
            "sourceMaps": true,
            "cwd": "${workspaceRoot}",
            "ionicLiveReload": false
        },
        {
            "name": "Run iOS on simulator",
            "type": "cordova",
            "request": "launch",
            "platform": "ios",
            "target": "emulator",
            "port": 9220,
            "sourceMaps": true,
            "cwd": "${workspaceRoot}",
            "ionicLiveReload": false
        },
        {
            "name": "Attach to running android on emulator",
            "type": "cordova",
            "request": "attach",
            "platform": "android",
            "target": "emulator",
            "port": 9222,
            "sourceMaps": true,
            "cwd": "${workspaceRoot}"
        },
        {
            "name": "Attach to running iOS on simulator",
            "type": "cordova",
            "request": "attach",
            "platform": "ios",
            "target": "emulator",
            "port": 9220,
            "sourceMaps": true,
            "cwd": "${workspaceRoot}"
        },
        {
            "name": "Serve to the browser (ionic serve)",
            "type": "cordova",
            "request": "launch",
            "platform": "serve",
            "cwd": "${workspaceRoot}",
            "devServerAddress": "localhost",
            "sourceMaps": true,
            "ionicLiveReload": true
        },
        {
            "name": "Simulate Android in browser",
            "type": "cordova",
            "request": "launch",
            "platform": "android",
            "target": "chrome",
            "simulatePort": 8000,
            "livereload": true,
            "sourceMaps": true,
            "cwd": "${workspaceRoot}"
        },
        {
            "name": "Simulate iOS in browser",
```

```
                "type": "cordova",
                "request": "launch",
                "platform": "ios",
                "target": "chrome",
                "simulatePort": 8000,
                "livereload": true,
                "sourceMaps": true,
                "cwd": "${workspaceRoot}"
            }
]
}
```

、デバッグのためにのまたはショート·キーをします。

1. デバッグメニューにします。
2. [デバッグ]をクリックします。

   または

```
Short keys
```

- デバッグ - F5
- ステップオーバー - F10
- ステップインとステップアウト - F11
- デバッグをする - Shift + F5
- デバッグをする-ctrl + shift_F5

オンラインでVisual Studioコードでのionic 2のセットアップとデバッグをむ
https://riptutorial.com/ja/ionic2/topic/10559/visual-studioコードでのionic-2のセットアップとデバッグ

8: コードからApp Storeへ - Android

き

プロダクションイオンアプリをしてGoogle Playにアップロードするについて、をにします。

Examples

アプリプロジェクトの

アプリストアのができているAndroidアプリをするときは、 `ionic start`をするときに、 `--appname|-a`と`--id|-i`フラグをすることがです。これはのアプリからあなたのアプリをするためにGoogle Playでされます。

しいモバイルアプリプロジェクトをするは、のcliサンプルをできます。

```
$ ionic start --v2 -a "App Example" -i "com.example.app" -t "tabs"
```

1.アプリケーションファイル

のアプリでこのをしたいは、 `config.xml`をすることができ`config.xml`。のコマンドをって`config.xml`をすることをおめします。

`widget id`、 `name`、 `description`、および`author`を/します。

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<widget id="com.example.app" version="1.0.0" xmlns="http://www.w3.org/ns/widgets"
xmlns:cdv="http://cordova.apache.org/ns/1.0">
  <name>Example App</name>
  <description>Example app for stackoverflow users</description>
  <author email="admin@example.com" href="http://example.com/">Your name or team</author>
  ...
</widget>
```

2.アイコンとスプラッシュ

アイコンとスプラッシュイメージでサポートされているファイルタイプは、どちらもpng、psd、またはaiであり、 `icon`や`splash`するファイルがプロジェクトのルートにあるリソースディレクトリのにかれているがあります。アイコンのは192x192ピクセルで、みのあるがありません。スプラッシュははるかにですので、ここをクリックしてをごください。それにもかかわらず、サイズは2208x2208ピクセルです。

このコマンドをすることをするために、アイコンファイルをっているは`ionic resources --icon`あなたは、このコマンドをすることをするために、ファイルをスプラッシュしているは`ionic resources --splash`

ビルディングプロダクションアプリ

プロダクションアプリをするに、ログデータをしてください。

デフォルトですべてのがわれたリリースをするには、 - release--prodタグをします

```
ionic build android --release --prod
```

なのについては、 @ ionic / app-scriptsリポジトリをしてください。

4.をする

これで、されていないAPK `android-release-unsigned.apk` に`android-release-unsigned.apk` 、アライメントユーティリティをしてし、アプリストアのをするがあります。すでにキーがあるは、これらのをスキップし、わりにそのキーをしてください。

に、されていないAPKファイル`android-release-unsigned.apk`をプロジェクトdir `/platforms/android/build/outputs/apk/`し、apkファイルのにする`keytools`コマンドをします。のをすることができます

```
$ keytool -genkey -v -keystore my-release-key.keystore -alias androidKey -keyalg RSA -keysize
2048 -validity 10000
```

のディレクトリに`my-release-key.keystore`があります。

JDKにのkeytoolコマンドをしてプライベートキーをしましょう。このツールがつからないは、インストールガイドをしてください。

に、キーストアのパスワードをするようにめられます。その、らしいツールののりのにえて、それがすべてしたら、のディレクトリにされたmy-release-key.keystoreというファイルをするがあります。

このファイルはなにしてください。したは、アプリにアップデートをできません。

5. APKにする

のないAPKにするには、JDKにまれているjarsignerツールをします。

```
$ jarsigner -verbose -sigalg SHA1withRSA -digestalg SHA1 -keystore my-release-key.keystore
HelloWorld-release-unsigned.apk alias_name
```

これはAPKににする。に、zip alignツールをしてAPKをするがあります。 zipalignツールは/ path / to / Android / sdk / build-tools / VERSION / zipalignにあります。

```
$ zipalign -v 4 HelloWorld-release-unsigned.apk HelloWorld.apk
```

HelloWorld.apkとばれるのバイナリがしました。これをのGoogle Playストアですることができます。

Google Playストアにアプリをします。 Google PlayストアけにリリースAPKをしましたので、Playストアのリストをして APKをアップロードすることができます。まず、Google Playストアデベロッパーコンソールにアクセスして、しいデベロッパーアカウントをするがあります。 125ドルのがかかります。

デベロッパーアカウントをしたら、「Google PlayでAndroidアプリをする」をクリックし、のにいます。

オンラインでコードからApp Storeへ - Androidをむ https://riptutorial.com/ja/ionic2/topic/9659/コードからapp-storeへ----android

9: コンストラクタとOnInit

き

ionic2にして、 `constructor`はにえば、プラグインやサービスなどのインスタンスをするためにします。たとえば、すべてののリストをするページビューがあり、jsonファイルがありますあなたがしなければならないことは、このサービスでサービスをすることです。メソッドをし、http.getリクエストをしてjsonデータをします。ここでがですか httpはにこのようにします

Examples

コンストラクタでHttpをするためのサービスメソッドの

```
import {Http} from '@angular/http';
@Injectable()
export class StudentService{
    constructor(public http: Http){}
    getAllStudents(): Observable<Students[]>{
        return this.http.get('assets/students.json')
        .map(res => res.json().data)
        }
    }
```

このサービスメソッドをしたいはコンストラクタにもうしてください。たちはview / pageにきます

```
import {StudentService} from './student.service';
import { SocialSharing } from '@ionic-native/social-sharing';
export class HomePage implements OnInit {

  constructor(public _studentService: StudentService, public socialSharing: SocialSharing) {
   }
```

ここでもコンストラクタにしてください。コンストラクタでStudentServiceのインスタンスをしています。もう1つは、socialSharingプラグインをしているため、コンストラクタでそのインスタンスをしています。

ngOnInitメソッドをして、ビューのみみにのリストをする

`OnInit` これはionic2のにすばらしいことです。またはAngularJs2でうことができます。のでは、ngOnInitがであるかをることができます。だから、サービスメソッドでがいました。あなたのビュー/ページには、ビューがされるとすぐにそのリストデータをできるようになります。これは、ロードににこるのでなければなりません。リストをするがあります。したがって、クラスはOnInitをし、ngOnInitをします。

ngOnInitページ/ビューののリストをする

```
export class HomePage implements OnInit {
...
....
constructor(....){}

ngOnInit(){
    this._studentService.getAllStudents().subscribe(
     (students: Students[]) => this.students = students,
    )
```

オンラインでコンストラクタとOnInitをむ https://riptutorial.com/ja/ionic2/topic/9907/コンストラクタとoninit

10: サービスの

サービスのにするなの1つは、するがあるのコンポーネントの`providers`にまれなければならないことです。

ですかさて、`Component providers`に`MyService`リファレンスをめるとしましょう。かのようなもの

```
@Component({
  templateUrl:"page1.html",
  providers: [MyService]
})
```

そして

```
@Component({
  templateUrl:"page2.html",
  providers: [MyService]
})
```

こうすることで、サービスのしいインスタンスがコンポーネントごとにされるので、1ページでデータをするインスタンスは、データをするためにされたインスタンスとはなります。それはうまくいきません。

アプリケーションがじインスタンスをするようにサービスをシングルトンサービスとしてさせるため、のようにそのを`App Component`することができます

```
@Component({
  template: '<ion-nav [root]="rootPage"></ion-nav>',
  providers: [MyService]
})
```

また、 `MyService`リファレンスを`ionicBootstrap(MyApp, [MyService]);`することもでき`ionicBootstrap(MyApp, [MyService]);`しかしAngular2スタイルのガイドによると

> Angular 2インジェクタは、されるのコンポーネントにサービスをします。
>
> どうして Angular 2インジェクタはです。
>
> どうしてトップレベルのコンポーネントにサービスをする、そのインスタンスはされ、そののコンポーネントのすべてのコンポーネントができます。
>
> どうしてこれは、サービスがメソッドやをしているときにです。
>
> どうしてこれは、2つのなるコンポーネントがサービスのなるインスタンスをとするにはではありません。このシナリオでは、およびのインスタンスをとするコンポーネントレベルでサービスをするがよいでしょう。

そして

> それがします。それはベストプラクティスではありません。 ブートストラッププロバイダオプションは 、ルーティングサポートなど、 **Angular**のサービスをおよびきするためのものです 。

... `App Component`がなになります。

Examples

なるページでをする

サービスをするもなの1つは、アプリケーションののページからいくつかのデータをし、そのデータをのページからするです。

1つのオプションは、そのデータをパラメータとしてすることですたとえば、1つのページがのページをびすが、アプリケーションのになるからそのデータをする、これはのではないようですそれ。 サービスがされるのはそのときです。

このでは、をする`MyService saveMessage()`と、をする`getMessage()` 2つのなメソッドしかない`MyService`というなサービスをします。このコードは、 このプランナーのであり、にそれをることができます。

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

@Injectable()
export class MyService {

  private message: string;

  constructor(){ }

  public saveMessage(theMessage: string): void {
    this.message = theMessage;
  }

  public getMessage(): string {
    return this.message;
  }
}
```

に、しいメッセージをするは、 `saveMessage(theMessageWeWantToSave);`できる`saveMessage(theMessageWeWantToSave);`メソッドを`MyService`インスタンスに`service`とばれ`service` から`MyService`ます。

```
import { Component } from "@angular/core";
import { MyService } from 'service.ts';

@Component({
  templateUrl:"page1.html"
})
```

```
export class Page1 {

  message: string;

  // ...

  public saveSecretMessage(): void {
    this.service.saveMessage(this.message);
  }
}
```

じように、そのデータをするは、のようにサービスインスタンスから`getMessage()`メソッドをできます。

```
import { Component } from "@angular/core";
import { MyService } from 'service.ts';

@Component({
  templateUrl:"page2.html"
})
export class Page2 {

  enteredMessage: string;

  constructor(private service: MyService) {
    this.enteredMessage = this.service.getMessage();
  }

  // ...
}
```

`MyService`サービスのがどこにまれるべきか、なぜそれがまれるべきかをるには、 *Remarks*セクションをチェックしてください。

オンラインでサービスのをむ https://riptutorial.com/ja/ionic2/topic/4407/サービスの

11: ジオロケーション

Examples

ない

あなたの`package.json`にのをめてください

```
{
    ...
    "dependencies": {
        ...
        "ionic-native": "^1.3.10",
        ...
    },
    ...
}
```

ジオロケーションをするには

```
// custom-component.ts

import {Geolocation} from 'ionic-native';
import template from './custom-component.html';

@Component({
    selector: 'custom-component',
    template: template
})
export class CustomComponent {

    constructor() {

        // get the geolocation through a promise
        Geolocation.getCurrentPosition().then((position:Geoposition)=> {
            console.log(
                position.coords.latitude,
                position.coords.longitude);
        });
    }
}
```

ポジションをる

よりリアルタイムなソリューションのためには、エラーまたはのがしたときにするGeolocationのwatchPositionをできます。 getCurrentPositionとはなり、watchPositionはObservableをします

```
import {Geolocation} from 'ionic-native';
import template from './custom-component.html';

@Component({
```

```
selector: 'custom-component',
template: template
})
export class CustomComponent {
constructor() {

    // get the geolocation through an observable
        Geolocation.watchPosition(<GeolocationOptions>{
            maximumAge: 5000, // a maximum age of cache is 5 seconds
            timeout: 10000, // time out after 10 seconds
            enableHighAccuracy: true // high accuracy
        }).subscribe((position) => {
            console.log('Time:' + position.timestamp);
            console.log(
                'Position:' + position.coords.latitude + ',' +
                position.coords.longitude);
            console.log('Direction:' position.coords.heading);
            console.log('Speed:' position.coords.speed);

        });
}
```

オンラインでジオロケーションをむ https://riptutorial.com/ja/ionic2/topic/5840/ジオロケーション

12: タブの

のとをるために、 Ionic 2 Tabのドキュメントをチェックしてください。

Examples

ページからプログラムによってしたタブをする

こののPlunkerで、なコードをることができます。

このでは、サービスをして、タブのページページとタブコンテナタブをするコンポーネントのをします。 イベントでそれをやることはできますが、サービスのアプローチがきです。わかりやすく、またアプリケーションのがまるときにもつからです。

TabService

```
import {Injectable} from '@angular/core';
import {Platform} from 'ionic-angular/index';
import {Observable} from 'rxjs/Observable';

@Injectable()
export class TabService {

  private tabChangeObserver: any;
  public tabChange: any;

  constructor(private platform: Platform){
    this.tabChangeObserver = null;
    this.tabChange = Observable.create(observer => {
        this.tabChangeObserver = observer;
    });
  }

  public changeTabInContainerPage(index: number) {
    this.tabChangeObserver.next(index);
  }
}
```

したがって、に`TabService`は`Observable`をして、タブコンテナがそれにサブスクライブできるようにし、ページからびされる`changeTabInContainerPage()`メソッドもします。

に、ページタブのページで、ボタンをして`click`イベントをサービスをびすメソッドにバインドし`click`。

Page1.html

```
<ion-content class="has-header">
    <h1>Page 1</h1>
    <button secondary (click)="changeTab()">Select next tab</button>
</ion-content>
```

Page1.ts

```
import { Component } from '@angular/core';
import { Observable } from 'rxjs/Observable';
import { TabService } from 'tabService.ts';

@Component({
  templateUrl:"page1.html"
})
export class Page1 {

  constructor(private tabService: TabService) { }

  public changeTab() {
    this.tabService.changeTabInContainerPage(1);
  }
}
```

に、 `TabsPage`では、サービスにするだけで、したタブを`this.tabRef.select(index);`でします
`this.tabRef.select(index);`

```
import { Component, ViewChild } from "@angular/core";
import { Page1 } from './page1.ts';
import { Page2 } from './page2.ts';
import { TabService } from 'tabService.ts';

@Component({
  templateUrl: 'tabs.html'
})
export class TabsPage {
  @ViewChild('myTabs') tabRef: Tabs;

  tab1Root: any = Page1;
  tab2Root: any = Page2;

  constructor(private tabService: TabService){
    this.tabService.tabChange.subscribe((index) => {
      this.tabRef.select(index);
    });
  }
}
```

`ion-tabs`に`#myTabs`をしてTabsインスタンスへのをしていることにしてください。これは
`@ViewChild('myTabs') tabRef: Tabs;`コンポーネントからし`@ViewChild('myTabs') tabRef: Tabs;`

```
<ion-tabs #myTabs>
  <ion-tab [root]="tab1Root" tabTitle="Tab 1"></ion-tab>
  <ion-tab [root]="tab2Root" tabTitle="Tab 2"></ion-tab>
</ion-tabs>
```

selectedIndexでタブを

DOMへのをするわりに、イオンタブのselectedIndexをしてタブのインデックスをすることができます

HTML

```
<ion-tabs [selectedIndex]="tabIndex" class="tabs-icon-text" primary >
     <ion-tab tabIcon="list-box"    [root]="tabOne"></ion-tab>
     <ion-tab tabIcon="contacts"  [root]="tabTwo"></ion-tab>
     <ion-tab tabIcon="chatboxes"  [tabBadge]="messagesReceived" [root]="tabFive"></ion-tab>
</ion-tabs>
```

TS

```
import { Events} from "ionic-angular";

export class tabs {
 public tabIndex: number;
 constructor(e: Events) {
   tabs.mySelectedIndex = navParams.data.tabIndex || 0;
   e.subscribe("tab:change", (newIndex) => this.tabIndex = newIndex);
 }
}
```

のコントローラサービスからしたいは、イベントをすることができます

```
e.publish("tab:change",2);
```

オンラインでタブのをむ https://riptutorial.com/ja/ionic2/topic/5569/タブの

13: での 'show-delete'の

Examples

は2のイオン2をってモバイルアプリをしています。

はイオンリストにイオンアイテムをめました。はそれらのイオンアイテムが、イオンウェブサイトでここにされているように、にじてされることをみます。

しかし、くはのバージョンイオン2にされていると1でてのイオンのアイテムをくつのボタンののスタイルは、もはやサポートされているショーのとshow-するので、もはやではないではありません。できるのオプションはイオンアイテムとしてイオンアイテムをスライドさせることです。これにより、ボタンをするためにアイテムを1つずつスライドさせることができます。

それはがんでいたものではありません。はすべてのイオンアイテムをにく1つのボタンがしかった。

しばらくをやした、はなソリューションをえし、イオン2をってのをすることができました。はそれをあなたとします。

ここにのソリューションです

.htmlファイル

```
<ion-header>
  <ion-navbar>
    <ion-buttons start (click)="manageSlide()">
      <button>
        <ion-icon name="ios-remove"></ion-icon>
      </button>
    </ion-buttons>
    <ion-title>PageName</ion-title>
  </ion-navbar>
</ion-header>
```

リストのために

```
<ion-list #list1>
  <ion-item-sliding #slidingItem *ngFor="let contact of contacts | sortOrder">
    <button #item ion-item>
      <p>{{ item.details }}</p>
      <ion-icon id="listIcon" name="arrow-forward" item-right></ion-icon>
    </button>
    <ion-item-options side="left">
      <button danger (click)="doConfirm(contact, slidingItem)">
        <ion-icon name="ios-remove-circle-outline"></ion-icon>
      Remove
      </button>
    </ion-item-options>
  </ion-item-sliding>
```

```
</ion-list>
```

.tsファイルでは、まずインポートをいます。

```
import { ViewChild } from '@angular/core';
import { Item } from 'ionic-angular';
import { ItemSliding, List } from 'ionic-angular';
```

ViewChildをしてhtmlをします。

```
@ViewChild(List) list: List;
```

に、クラスをしてをします。

```
public manageSlide() {

    //loop through the list by the number retreived of the number of ion-item-sliding in the
list
    for (let i = 0; i < this.list.getElementRef().nativeElement.children.length; i++) {

        // retreive the current ion-item-sliding
        let itemSlide = this.list.getElementRef().nativeElement.children[i].$ionComponent;

        // retreive the button to slide within the ion-item-sliding
        let item = itemSlide.item;

        // retreive the icon
        let ic = item._elementRef.nativeElement.children[0].children[1];

        if (this.deleteOpened) {
            this.closeSlide(itemSlide);
        } else {
            this.openSlide(itemSlide, item, ic);
        }
    }

    if (this.deleteOpened) {
        this.deleteOpened = false;
    } else {
        this.deleteOpened = true;
    }
}
```

にオープニングクラス

```
private openSlide(itemSlide: ItemSliding, item: Item, inIcon) {
  itemSlide.setCssClass("active-sliding", true);
  itemSlide.setCssClass("active-slide", true);
  itemSlide.setCssClass("active-options-left", true);
  item.setCssStyle("transform", "translate3d(72px, 0px, 0px)")
}
```

クローズクラス

```
private closeSlide(itemSlide: ItemSliding) {
  itemSlide.close();
  itemSlide.setCssClass("active-sliding", false);
  itemSlide.setCssClass("active-slide", false);
  itemSlide.setCssClass("active-options-left", false);
```

}

はそれがあなたのうちのいくつかをけることをっています。

しんでいいコーディング...

オンラインでの 'show-delete'のをむ https://riptutorial.com/ja/ionic2/topic/6620/での--show-delete-の-ion-list-

14: プッシュの

のでされているSenderIDは、Googleからされたgcmidです。プラグインをインストールするときにもされます。

```
ionic plugin add phonegap-plugin-push --variable SENDER_ID="XXXXXXX"
```

プッシュにデータをするは、このリンクをして、のをしてください。
https://github.com/phonegap/phonegap-plugin-push/blob/master/docs/TYPESCRIPT.md

Examples

プッシュプラグインは、されたIDをしてプラグインのをするようにinitにをします。

```
 let push = Push.init({
     android: {
       senderID: "------------",
     },
     ios: {
       alert: "true",
       badge: true,
       sound: "false",
     },
     windows: {},
   });
```

ステップでは、アプリをデバイスのシステムにし、IDをします

```
import { Push, RegistrationEventResponse} from "ionic-native";

       //the push element is created in the initialization example
       push.on("registration", async (response: RegistrationEventResponse) => {
                   //The registration returns an id of the registration on your device
                   RegisterWithWebApi(response.registrationId);

       });
```

プッシュをする

プッシュをするには、プッシュにするようにプラグインにするがあります。このステップは、とにされます。

```
import { Push, NotificationEventResponse} from "ionic-native";

       //the push element is created in the initialization example
       push.on("notification", (response: NotificationEventResponse) => {
           let chatMessage: ChatMessage = <ChatMessage>{
             title: response.title,
             message: response.message,
```

```
            receiver: response.additionalData.replyTo,
            image: response.image
          };
          DoStuff(chatMessage));
      });
```

オンラインでプッシュのをむ https://riptutorial.com/ja/ionic2/topic/5874/プッシュの

15: モーダル

Examples

モーダルの

モーダルは、なUIをするためににスライドします。ログインやサインアップページ、メッセージの、オプションのによくされます。

```
import { ModalController } from 'ionic-angular';
import { ModalPage } from './modal-page';

export class MyPage {
  constructor(public modalCtrl: ModalController) {
  }

  presentModal() {
    let modal = this.modalCtrl.create(ModalPage);
    modal.present();
  }
}
```

モーダルは、ユーザーののページをきするコンテンツペインです。

モーダルによるデータのけし

`Modal.create()`をしてしいモーダルにデータをすことができます。その、いたページから`NavParams`をしてデータにアクセスできます。モーダルとしていたページにはな「モーダル」ロジックは`NavParams`ませんが、 `NavParams`はページと`NavParams`ません。

ページ

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

export class HomePage {

 constructor(public modalCtrl: ModalController) {

 }

 presentProfileModal() {
   let profileModal = this.modalCtrl.create(Profile, { userId: 8675309 });
   profileModal.present();
 }

}
```

*2*のページ

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

```
export class Profile {

 constructor(params: NavParams) {
   console.log('UserId', params.get('userId'));
 }

}
```

オンラインでモーダルをむ https://riptutorial.com/ja/ionic2/topic/6415/モーダル

16: モーダル

Examples

なモーダル

モーダルは、のページのにされるなUIです。これは、ログイン、サインアップ、のオプションの、オプションのによくされます。

たちはモーダルをったなをてみましょう。まず、イオンブランクプロジェクトをしています。メッセージをするなモーダルをし、ボタンクリックでしましょう。にこれをうには、モーダルのビューをしています。

Message.html

```
<ion-header>
  <ion-toolbar>
    <ion-title>
      Modal
    </ion-title>
    <ion-buttons start>
      <button (click)="dismiss()">
        <span primary showWhen="ios">Cancel</span>
        <ion-icon name="md-close" showWhen="android,windows"></ion-icon>
      </button>
    </ion-buttons>
  </ion-toolbar>
</ion-header>
<ion-content padding>
  <h1>Modal Without Params is created successfully.</h1>
  <button full (click)="dismiss()"> Exit </button>
</ion-content>
```

Message.ts

```
import { Component } from '@angular/core';
import { ViewController } from 'ionic-angular';
@Component({
  templateUrl: 'build/pages/message/message.html',
})
export class MessagePage {
  viewCtrl;
  constructor(viewCtrl: ViewController) {
    this.viewCtrl = viewCtrl;
  }
  dismiss(){
    this.viewCtrl.dismiss();
  }
}
```

このモーダルはメッセージをします。モーダルは、View controllers **dismiss**メソッドをしてした

り、 ""することができます。

Home.html

```
<ion-header>
  <ion-navbar>
    <ion-title>
      Modal Example
    </ion-title>
  </ion-navbar>
</ion-header>
<ion-content padding>
  <button full (click)="openModal()">ModalWithoutParams-Message</button>
</ion-content>
```

Home.ts

```
import { Component } from '@angular/core';
import { ModalController } from 'ionic-angular';
import {MessagePage} from '../message/message';
@Component({
  templateUrl: 'build/pages/home/home.html'
})
export class HomePage {
  modalCtrl;
  data;
  constructor(modalCtrl: ModalController) {
    this.modalCtrl = modalCtrl;
    this.data = [{name: "aaa", email: "aaa.a@som.com", mobile: "1234567890", nickname: "zzz"},
      {name: "bbb", email: "bbb.a@som.com", mobile: "1234567890", nickname: "yyy"},
      {name: "ccc", email: "ccc.a@som.com", mobile: "1234567890", nickname: "xxx"}]
  }
  openModal() {
    let myModal = this.modalCtrl.create(MessagePage);
    myModal.present();
  }
}
```

は、 **ModalController**とデータモデルMessagePageをインポートするホームページをしています。 ModalControllerの**create**メソッドは、myModalをするためにされたデータモデルMessagePageのモーダルをします。 のメソッドは、のページのにモーダルをきます。

パラメータののモーダル

たちは、モーダルをするをっています。しかし、モーダルからたちのホームページにいくつかのデータをすはどうでしょうか。これをうには、モーダルをしてページにパラメータをすというをてみましょう。

Register.html

```
<ion-header>
  <ion-toolbar>
    <ion-title>
      Login
```

```
    </ion-title>
    <ion-buttons start>
      <button (click)="dismiss()">
        <span primary showWhen="ios">Cancel</span>
        <ion-icon name="md-close" showWhen="android,windows"></ion-icon>
      </button>
    </ion-buttons>
  </ion-toolbar>
</ion-header>
<ion-content padding>
  <ion-list>
    <ion-item>
      <ion-label>Name</ion-label>
      <ion-input type="text" [(ngModel)]="name"></ion-input>
    </ion-item>
    <ion-item>
      <ion-label>Email</ion-label>
      <ion-input type="text" [(ngModel)]="email"></ion-input>
    </ion-item>
    <ion-item>
      <ion-label>Mobile</ion-label>
      <ion-input type="number" [(ngModel)]="mobile"></ion-input>
    </ion-item>
    <ion-item>
      <ion-label>Nickname</ion-label>
      <ion-input type="text" [(ngModel)]="nickname"></ion-input>
    </ion-item>
  </ion-list>
  <button full (click)="add()">Add</button>
</ion-content>
```

Register.ts

```
import { Component } from '@angular/core';
import { ViewController } from 'ionic-angular';
@Component({
  templateUrl: 'build/pages/register/register.html',
})
export class ResisterPage {
  viewCtrl;
  name;
  email;
  mobile;
  nickname;
  constructor(viewCtrl: ViewController) {
    this.viewCtrl = viewCtrl;
    this.name = "";
    this.email = "";
    this.mobile = "";
    this.nickname = "";
  }
  dismiss(){
    this.viewCtrl.dismiss();
  }
  add(){
    let data = {"name": this.name, "email": this.email, "mobile": this.mobile, "nickname":
this.nickname};
    this.viewCtrl.dismiss(data);
  }
}
```

Register modalは、ユーザーがしたをつデータオブジェクトをし、パラメータはviewControllers dismissメソッドをしてじたときにのページにされます。これでパラメータがされます。

では、ホームページのパラメータをどのようにするのでしょうかこれをうには、ホームページにボタンをし、クリックにモーダルをびします。ユーザーをするために、リストをしています。

Home.html

```
<ion-list>
  <ion-item *ngFor="let datum of data">
    <h1>{{datum.name}}</h1>
  </ion-item>
</ion-list>
<button full secondary (click)="openModalParams()">ModalWithParams-Register</button>
```

Home.ts

```
import {ResisterPage} from '../register/register';

  openModalParams(){
    let modalWithParams = this.modalCtrl.create(ResisterPage);
    modalWithParams.present();

    modalWithParams.onDidDismiss((result) =>{
      if(result){
        this.data.unshift(result);
      }
    });
  }
```

ViewController **onDidDismiss**メソッドは、モーダルがじられるたびにされます。データがモーダルからパラメータとしてされた、onDidDismissメソッドをしてデータをできます。ここでは、ユーザーがしたデータがのデータにされます。パラメータとしてされるデータがない、りはnullになります。

のパラメータをつモーダル

パラメータをモーダルにすことは、NavControllerにをすとています。そうするために、リストをクリックし、なパラメータを**create**メソッドの2のとしてすときに、home.htmlのリストをしてモーダルをきます。

Home.html

```
 <ion-list>
    <ion-item *ngFor="let datum of data" (click)="openModalwithNavParams(datum)">
      <h1>{{datum.name}}</h1>
    </ion-item>
  </ion-list>
```

Home.ts

```
import {EditProfilePage} from '../edit-profile/edit-profile';

  openModalwithNavParams(data){
    let modalWithNavParams = this.modalCtrl.create(EditProfilePage,{Data: data});
    modalWithNavParams.present();
  }
```

のビューとに、NavParamsをして、のビューからされたデータをします。

Edit-Profile.html

```
<ion-header>
  <ion-toolbar>
    <ion-title>
      Login
    </ion-title>
    <ion-buttons start>
      <button (click)="dismiss()">
        <span primary showWhen="ios">Cancel</span>
        <ion-icon name="md-close" showWhen="android,windows"></ion-icon>
      </button>
    </ion-buttons>
  </ion-toolbar>
</ion-header>
<ion-content padding>
  <h2>Welcome {{name}}</h2>
  <ion-list>
    <ion-item>
      <ion-label>Email</ion-label>
      <ion-input type="text" value={{email}}></ion-input>
    </ion-item>
    <ion-item>
      <ion-label>Mobile</ion-label>
      <ion-input type="number" value={{mobile}}></ion-input>
    </ion-item>
    <ion-item>
      <ion-label>Nickname</ion-label>
      <ion-input type="text" value={{nickname}}></ion-input>
    </ion-item>
  </ion-list>
  <button full (click)="dismiss()">Close</button>
</ion-content>
```

Edit-Profile.ts

```
import { Component } from '@angular/core';
import { ViewController, NavParams } from 'ionic-angular';
@Component({
  templateUrl: 'build/pages/edit-profile/edit-profile.html',
})
export class EditProfilePage {
  viewCtrl;
  navParams;
  data;
  name;
  email;
  mobile;
  nickname;
```

```
  constructor(viewCtrl: ViewController, navParams: NavParams) {
    this.viewCtrl = viewCtrl;
    this.navParams = navParams;
    this.data = this.navParams.get('Data');
    this.name = this.data.name;
    this.email = this.data.email;
    this.mobile = this.data.mobile;
    this.nickname = this.data.nickname;

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

オンラインでモーダルをむ https://riptutorial.com/ja/ionic2/topic/6612/モーダル

17: ユニットテスト

き

ユニットテスティングはに、フィーチャの/にをぐために、にをします。「すべてのがしている」とうネット。テストは、のユーザーがなQAがうことができるかどうかをテストするをきえるものではありません。

このドキュメントでは、このリポジトリのをベースにします https : //github.com/driftyco/ionic-unit-testing-example

Examples

カルマ/ジャスミンのユニットテスト

イオンでのユニットテストは、どのアプリでもじです。

これをうためにいくつかのフレームワークをします。

カルマ - テストをするためのフレームワーク

ジャスミン - テストをくためのフレームワーク

PhantomJS - ブラウザなしでjavascriptをするアプリケーション

まずにすべてをインストールできるので、package.jsonにdevのにこれらのがまれていることをしてください。は、のはあなたのアプリケーションにまったくをえず、をけるためだけにあることにすることがだとじています。

```
"@ionic/app-scripts": "1.1.4",
"@ionic/cli-build-ionic-angular": "0.0.3",
"@ionic/cli-plugin-cordova": "0.0.9",
"@types/jasmine": "^2.5.41",
"@types/node": "^7.0.8",
"angular2-template-loader": "^0.6.2",
"html-loader": "^0.4.5",
"jasmine": "^2.5.3",
"karma": "^1.5.0",
"karma-chrome-launcher": "^2.0.0",
"karma-jasmine": "^1.1.0",
"karma-jasmine-html-reporter": "^0.2.2",
"karma-sourcemap-loader": "^0.3.7",
"karma-webpack": "^2.0.3",
"null-loader": "^0.1.1",
"ts-loader": "^2.0.3",
"typescript": "2.0.9"
```

パッケージをちょっとべる

```
"angular2-template-loader": "^0.6.2", - will load and compile the angular2 html files.

"ts-loader": "^2.0.3", - will compile the actual typescript files

"null-loader": "^0.1.1", - will not load the assets that will be missing, such as fonts and
images. We are testing, not image lurking.
```

また、このスクリプトをpackage.jsonスクリプトにするがあります。

```
"test": "karma start ./test-config/karma.conf.js"
```

また、コンパイルにspec.tsファイルをしていることにしてください。

```
 "exclude": [
    "node_modules",
    "src/**/*.spec.ts"
  ],
```

さて、のテストをることができます。プロジェクトフォルダに`test-config`フォルダをします。 ちょうどpackage.jsonスクリプトでされたようにフォルダのに3つのファイルをします

`webpack.test.js` - テストプロセスのためにロ―ドするファイルをwebpackにします

```
var webpack = require('webpack');
var path = require('path');

module.exports = {
  devtool: 'inline-source-map',

  resolve: {
    extensions: ['.ts', '.js']
  },

  module: {
    rules: [
      {
        test: /\.ts$/,
        loaders: [
          {
            loader: 'ts-loader'
          } , 'angular2-template-loader'
        ]
      },
      {
        test: /\.html$/,
        loader: 'html-loader'
      },
      {
        test: /\.(png|jpe?g|gif|svg|woff|woff2|ttf|eot|ico)$/,
        loader: 'null-loader'
      }
    ]
  },

  plugins: [
    new webpack.ContextReplacementPlugin(
```

```
      // The (\\|\/) piece accounts for path separators in *nix and Windows
      /angular(\\|\/)core(\\|\/)(esm(\\|\/)src|src)(\\|\/)linker/,
      root('./src'), // location of your src
      {} // a map of your routes
    )
  ]
};

function root(localPath) {
  return path.resolve(__dirname, localPath);
}
```

`karma-test-shim.js` - ゾーンライブラリやテストライブラリなどののライブラリをみむだけでなく、モジュールをテストにします。

```
Error.stackTraceLimit = Infinity;

require('core-js/es6');
require('core-js/es7/reflect');

require('zone.js/dist/zone');
require('zone.js/dist/long-stack-trace-zone');
require('zone.js/dist/proxy');
require('zone.js/dist/sync-test');
require('zone.js/dist/jasmine-patch');
require('zone.js/dist/async-test');
require('zone.js/dist/fake-async-test');

var appContext = require.context('../src', true, /\.spec\.ts/);

appContext.keys().forEach(appContext);

var testing = require('@angular/core/testing');
var browser = require('@angular/platform-browser-dynamic/testing');

testing.TestBed.initTestEnvironment(browser.BrowserDynamicTestingModule,
browser.platformBrowserDynamicTesting());
```

`karma.conf.js` - カルマでテストするのをします。ここでは、ChromeからPhantomJSにりえることで、このプロセスをにえないものにすることができます。

```
var webpackConfig = require('./webpack.test.js');

module.exports = function (config) {
  var _config = {
    basePath: '',

    frameworks: ['jasmine'],

    files: [
      {pattern: './karma-test-shim.js', watched: true}
    ],

    preprocessors: {
      './karma-test-shim.js': ['webpack', 'sourcemap']
    },

    webpack: webpackConfig,
```

```
    webpackMiddleware: {
      stats: 'errors-only'
    },

    webpackServer: {
      noInfo: true
    },

    browserConsoleLogOptions: {
      level: 'log',
      format: '%b %T: %m',
      terminal: true
    },

    reporters: ['kjhtml', 'dots'],
    port: 9876,
    colors: true,
    logLevel: config.LOG_INFO,
    autoWatch: true,
    browsers: ['Chrome'],
    singleRun: false
  };

  config.set(_config);
};
```

これですべてをしてのテストをくことができました。このでは、app.componentファイルをします。メインコンポ―ネントではなくペ―ジのテストをしたいは、こちらをごください https://github.com/driftyco/ionic-unit-testing-example/blob/master/src/pages/page1/page1 spec.ts

にうがあるのは、コンストラクタをテストすることです。これにより、app.componentのコンストラクタがされ、されます

```
  beforeEach(async(() => {
    TestBed.configureTestingModule({
      declarations: [MyApp],
      imports: [
        IonicModule.forRoot(MyApp)
      ],
      providers: [
        StatusBar,
        SplashScreen
      ]
    })
  }));
```

にはなイオンアプリがまれます。は、このになです。すべてではない。

プロバイダは、コンストラクタにされたものをインクルードにめますが、インポートのではありません。たとえば、app.componentはPlatformサービスをしますが、IonicModuleのであるため、プロバイダでそれをするはありません。

のテストでは、コンポ―ネントのインスタンスをするがあります。

```
  beforeEach(() => {
    fixture = TestBed.createComponent(MyApp);
    component = fixture.componentInstance;
  });
```

は、すべてがであることをするためのテストです。

```
  it ('should be created', () => {
    expect(component instanceof MyApp).toBe(true);
  });

  it ('should have two pages', () => {
    expect(component.pages.length).toBe(2);
  });
```

に々はこのようなことをするでしょう

```
import { async, TestBed } from '@angular/core/testing';
import { IonicModule } from 'ionic-angular';

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

import { MyApp } from './app.component';

describe('MyApp Component', () => {
  let fixture;
  let component;

  beforeEach(async(() => {
    TestBed.configureTestingModule({
      declarations: [MyApp],
      imports: [
        IonicModule.forRoot(MyApp)
      ],
      providers: [
        StatusBar,
        SplashScreen
      ]
    })
  }));

  beforeEach(() => {
    fixture = TestBed.createComponent(MyApp);
    component = fixture.componentInstance;
  });

  it ('should be created', () => {
    expect(component instanceof MyApp).toBe(true);
  });

  it ('should have two pages', () => {
    expect(component.pages.length).toBe(2);
  });

});
```

テストを

```
npm run test
```

それはなテストのためです。あなたのTestBedをし、にあなたをけるかもしれないテストでをつようなテストライティングをショートカットするいくつかのがあります。

オンラインでユニットテストをむ https://riptutorial.com/ja/ionic2/topic/9561/ユニットテスト

クレジット

S. No		Contributors
1	ionic2をいめる	Akilan Arasu, Cameron637, carstenbaumhoegger, Community, FreeBird72, Guillaume Le Mière, Ian Pinto, Ketan Akbari, misha130, Raymond Ativie, sebaferreras, tymspy, Will.Harris
2	Angularfire2 / Firebaseによるソーシャルログイン	Cameron637, Gianfranco P.
3	Angularfire2 with Ionic2	Fernando Del Olmo
4	InAppBrowser	niks
5	ionic appをionic viewにする	Saravanan Sachi
6	Ionic2 CSSコンポーネント	Ketan Akbari
7	Visual StudioコードでのIonic 2のセットアップとデバッグ	misha130, PRIYA PARASHAR
8	コードからApp Storeへ - Android	Luis Estevez, misha130
9	コンストラクタとOnInit	niks
10	サービスの	sebaferreras
11	ジオロケーション	Matyas, misha130
12	タブの	misha130, sebaferreras
13	での 'show-delete'の	Amr ElAdawy, Roman Lee
14	プッシュの	misha130
15	モーダル	Raymond Ativie

16	ユニットテスト	misha130