UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

Centro de e-Learning

Desarrollo de aplicaciones para móviles

Centro de e-Learning SCEU UTN - BA.
Medrano 951 2do piso (1179) // Tel. +54 11 4867 7589 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning

UTN.BA UNIVERSIDAD TECNOLÓGICA NACIONAL FACULTAD REGIONAL BUENOS AIRES | Centro de e-Learning

Modulo 2: Desarrollo de aplicaciones

IONIC Generator

Presentación:

En la presente unidad comenzaremos a desarrollar nuestra primera aplicación.

Para realizar un desarrollo más rápido, utilizaremos las facilidades que nos propone ionic a través de IONIC Creator.

Objetivos:

Que los participantes*:

- Desarrollen la interfaz grafica de la aplicación utilizando IONIC generator
- Que puedan realizar las configuraciones básicas sobre una aplicación de IONIC
- Que comprendan algunos conceptos mas sobre el funcionamiento de Angular 2

Bloques temáticos*:

- IONIC generator
- Cambiar el root de nuestra aplicación
- App.modules.ts
- Aplicación mis sitios
- Cambiar iconos de los tabs
- Ejercicio

Consignas para el aprendizaje colaborativo

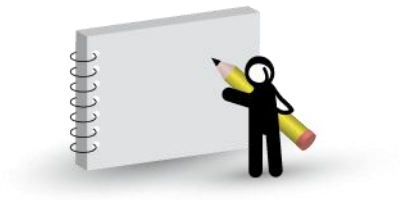
En esta Unidad los participantes se encontrarán con diferentes tipos de actividades que, en el marco de los fundamentos del MEC*, los referenciarán a tres comunidades de aprendizaje, que pondremos en funcionamiento en esta instancia de formación, a los efectos de aprovecharlas pedagógicamente:

- Los foros proactivos asociados a cada una de las unidades.
- La Web 2.0.
- Los contextos de desempeño de los participantes.

Es importante que todos los participantes realicen algunas de las actividades sugeridas y compartan en los foros los resultados obtenidos.

Además, también se propondrán reflexiones, notas especiales y vinculaciones a bibliografía y sitios web.

El carácter constructivista y colaborativo del MEC nos exige que todas las actividades realizadas por los participantes sean compartidas en los foros.



Tomen nota*

Las actividades son opcionales y pueden realizarse en forma individual, pero siempre es deseable que se las realice en equipo, con la finalidad de estimular y favorecer el trabajo colaborativo y el aprendizaje entre pares. Tenga en cuenta que, si bien las actividades son opcionales, su realización es de vital importancia para el logro de los objetivos de aprendizaje de esta instancia de formación. Si su tiempo no le permite realizar todas las actividades, por lo menos realice alguna, es fundamental que lo haga. Si cada uno de los participantes realiza alguna, el foro, que es una instancia clave en este tipo de cursos, tendrá una actividad muy enriquecedora.

Asimismo, también tengan en cuenta cuando trabajen en la Web, que en ella hay de todo, cosas excelentes, muy buenas, buenas, regulares, malas y muy malas. Por eso, es necesario aplicar filtros críticos para que las investigaciones y búsquedas se encaminen a la excelencia. Si tienen dudas con alguno de los datos recolectados, no dejen de consultar al profesor-tutor. También aprovechen en el foro proactivo las opiniones de sus compañeros de curso y colegas.

IONIC generator

Nosotros podemos crear las páginas a mano creando una carpeta dentro de la carpeta pages con su vista html y su controlador .ts, y su css, también podemos crear y configurar los tabs a mano, pero el cli (command line interface o interfaz de linea de comandos) de ionic nos facilita muchísimo el trabajo. Ionic dispone de una herramienta llamada **ionic generator**.

Con el siguiente comando obtenemos la lista de elementos disponible que podemos generar con ionic generator:

```
C:\Users\gilca>ionic g --list
```

- **Component**: Los componentes son un conjunto de html, con su css y su comportamiento que podemos reutilizar en cualquier lugar sin tener que reescribir de nuevo todo.
- **Directive**: Una directiva sirve para modificar atributos de un elemento.
- **Page**: Páginas.
- **Pipe**: Los pipes son lo que angular 1 se consideraban filtros.
- **Provider**:Los providers son proveedores que se encargan del manejo de datos, bien extraídos de la base de datos, desde una api, etc.
- **Tabs**: Nos genera la página maestra de Tabs y una página para cada tab.

Desde la versión 3 de ionic al crear una página con **ionic generator** además del archivo ***nombre_pagina.ts*** también se genera un archivo ***nombre_pagina.module.ts*** en cada página generada. Este archivo se utiliza para poder realizar lo que se conoce como **Lazy Loading**, que permite cargar paginas y recursos bajo demanda acelerando así la carga de las páginas.

Cambiar el root de nuestra aplicación

Para indicarle a la aplicación que deseamos modificar la página principal debemos editar el archivo **src/app/app.component.ts** e importar la pagina deseado con import y luego a la variable **rootPage** asignarle el nombre de la clase de la página que queremos asignar como página principal.

El archivo **app.component.ts** es el punto de entrada de nuestra aplicación, vemos que se define como un componente que utiliza como plantilla el archivo **app.html.**

```
@Component({
  templateUrl: 'app.html'
})
```

Si vemos el contenido de **app.html** podemos observar que únicamente contiene la siguiente linea de código:

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

Le asigna como **root** (es decir como página inicial o raíz) la variable **rootPage**

App.modules.ts

Todas las páginas y servicios que vayamos a utilizar los tenemos que declarar en el archivo app.module.ts.

Todas las páginas que creemos tienen que ser añadidas tanto al array declarations como al array entryComponents.

Todos los providers que creemos tienen que ser añadidos al array providers. Los providers los veremos más adelante.

Por último todos los componentes personalizados o pipes deben ser añadidos al array declarations.

Siempre que vayamos a declarar o utilizar alguna página, modulo o componente en cualquier lugar debemos importarlo primero, así que en app.module.ts debemos importar tanto la página maestra de los tabs como las tres páginas. Un vez importadas tal y como hemos mencionado, debemos declararlas en declarations y en entryComponents.

Generar una nueva pagina

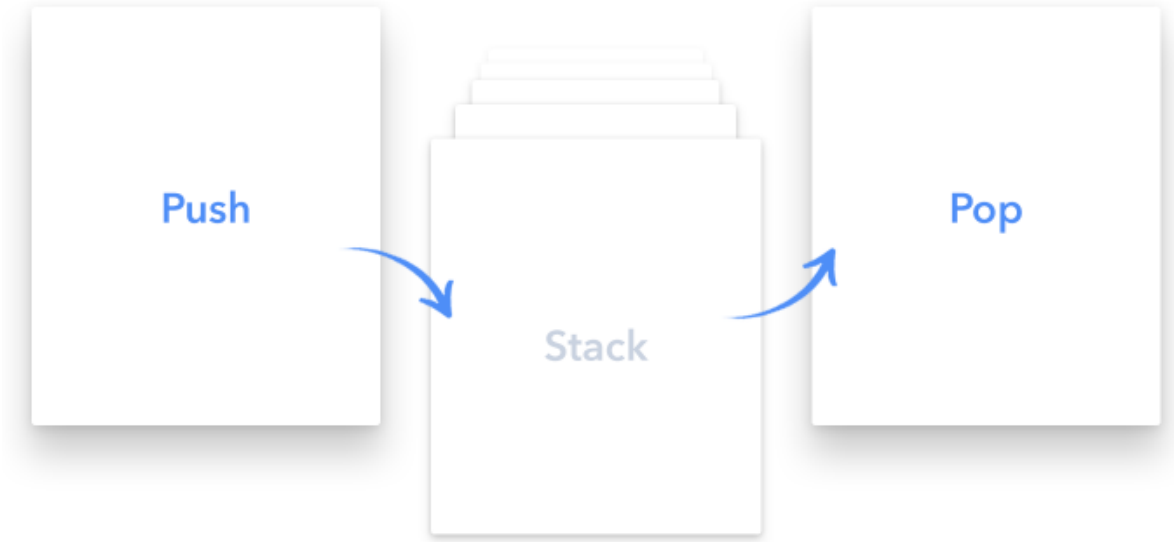
Con IONIC generator también podemos crear una nueva página, de la siguiente forma:

```
>ionic g page nuevaPagina
```

Podremos ver que en **/scr/app/pages** se habrá creado una nueva carpeta con el nombre que nosotros creamos con los archivos correspondientes.

Navegar páginas

Para poder navegar las páginas a través de IONIC necesitamos utilizar las funcionalidades de push y pop, veamos un esquema representativo:



1. Creamos una nueva página a la que llamaremos **navigation**

   ```
   s>ionic g page navigation
   ```

2. En la página en la cual haremos el enlace a **navigation** debemos importar el modulo de la siguiente manera:

   ```
   import {Navigation} from '../navigation/navigation';
   ```

3. Luego crearemos la variable de clase **navigation**, haciendo referencia al modulo **Navigation**

```
export class Inicio {
  navigation = Navigation;
  constructor(public navCtrl: Na
```

4. En la pagina que tendrá el enlace a la pagina **navigation** incluimos el botón correspondiente:

```
<ion-content padding>
    <button ion-button [navPush]="navigation" >Navegar</button>
</ion-content>
```

5. Por último en **app.modules.ts** debemos importar el componente Navigation de la siguiente forma:

a.
```
import { Navigation } from '../pages/navigation/navigation';
```

b.
```
declarations: [
  MyApp,
  MisTabs,
  Inicio,
  Listado,
  Info,
  Navigation
],
```

c.
```
entryComponents: [
  MyApp,
  MisTabs,
  Inicio,
  Listado,
  Info,
  Navigation
],
```

Con esto ya deberíamos tener el enlace a nuestra página **navigation** funcionando.

Link a imágenes

Para poder hacer referencias a nuestras imágenes, debemos colocar las mismas en **src/app/assets/**. Para mejorar la organización nos conviene crear una carpeta llamada **img.**

Luego en el html haremos referencia a nuestras imágenes de la siguiente manera:

```
<img src="assets/img/importar5_a.jpg"/>
```

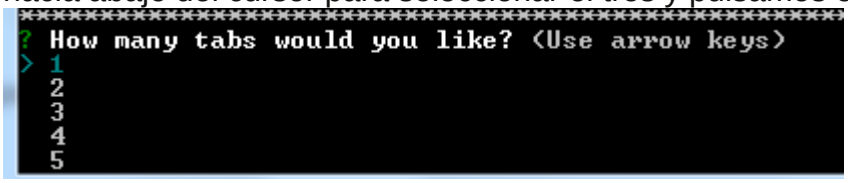
Aplicación mis sitios

1. Creamos nuestra nueva aplicación con **ionic start**

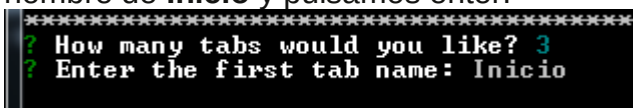
2. Creamos tabs para nuestra aplicación utilizando **ionic generator**

```
ionic g tabs misTabs
```

   a. Al ejecutar el comando primero nos pregunta el número de tabs que queremos crear, para este ejemplo vamos a utilizar 3.Utilizamos la flecha hacia abajo del cursor para seleccionar el tres y pulsamos enter.

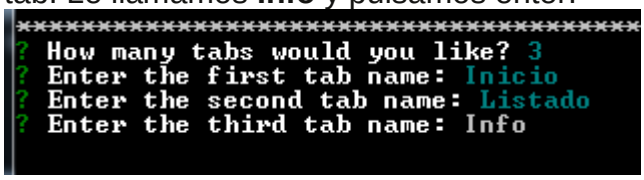


   b. Acto seguido debemos introducir el nombre del primer tab: le damos el nombre de **Inicio** y pulsamos enter.



   c. Después nos pide que introduzcamos el nombre para el segundo Tab: A este tab le vamos a llamar **Listado**.

   d. Pulsamos enter una vez mas y por ultimo introducimos el nombre del tercer tab. Le llamamos **Info** y pulsamos enter.



3. Eliminar el archivo ***xxx.module.ts*** de todas las páginas.

4. En **mis-tabs.ts** vemos que ha creado una variable para cada tab (***tab1Root***,***tab2Root,*** y ***tab3Root***) y les ha asignado el controlador de una página a cada uno. Sin embargo debemos importar dichas páginas para poder utilizarlas y asignárselas a cada tab.

```
import { Component } from '@angular/core';
import { NavController } from 'ionic-angular';
import { Inicio } from '../inicio/inicio';
import { Listado } from '../listado/listado';
import { Info } from '../info/info';
```

Debemos eliminar el import de ionic page y el **@IonicPage()** de todas las paginas.

```
import { IonicPage, NavController } from 'ionic-angular';
import { Inicio } from '../inicio/inicio';
import { Listado } from '../listado/listado';
import { Info } from '../info/info';
/**
 * Generated class for the MisTabs tabs.
 *
 * See https://angular.io/docs/ts/latest/guide/dependency-injection.html for
 * more info on providers and Angular DI.
 */
@Component({
  selector: 'page-mis-tabs',
  templateUrl: 'mis-tabs.html'
})
@IonicPage()
```

5. Bien, si ejecutamos **ionic seve -l** para ver e resultado ahora nos muestra la página home que se crea por defecto al crear un proyecto ionic vacío, tenemos que indicarle a la aplicación que la página inicial debe ser la página maestra con los tabs que acabamos de crear.

6. Ejecutar **ionic serve**, en este caso solo visualizaremos la página en blanco que por default nos prove IONIC.

7. Modificar el root de la aplicación:

    a. Para indicarle a la aplicación que la página principal sea la página maestra de los tabs que acabamos de crear debemos editar el archivo **src/app/app.component.ts** e importar la página **MisTabsPage** con import y luego a la variable **rootPage** asignarle el nombre de la clase de la página que queremos asignar como página principal, en este caso **MisTabsPage**, es decir debemos sustituir:

```
rootPage = HomePage;
```

Por:

```
rootPage = MisTabsPage;
```

El código del archivo **src/app/app.component.ts** completo quedaría así:

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

import { MisTabs } from '../pages/mis-tabs/mis-tabs';


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

  constructor(platform: Platform) {
    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();
      Splashscreen.hide();
    });
  }
}
```

El import de la página home ya no lo necesitamos así que podemos eliminarlo. de hecho podemos eliminar la carpeta de la pagina home ya que no la vamos a utilizar en esta aplicación.

b. Siempre que vayamos a declarar o utilizar alguna página, modulo o componente en cualquier lugar debemos importarlo primero, así que en **app.module.ts** debemos importar tanto la página maestra de los tabs como las tres páginas. Un vez importadas tal y como hemos mencionado, debemos declararlas en **declarations** y en **entryComponents.**

Vemos como quedaría el código del archivo **app.module.ts**. Destaco con fondo amarillo los cambios que debes realizar:

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule, ErrorHandler } from '@angular/core';
import { IonicApp, IonicModule, IonicErrorHandler } from 'ionic-angular';
import { SplashScreen } from '@ionic-native/splash-screen';
import { StatusBar } from '@ionic-native/status-bar';
import { MyApp } from './app.component';
import { MisTabs } from '../pages/mis-tabs/mis-tabs';
import { Inicio } from '../pages/inicio/inicio';
import { Listado } from '../pages/listado/listado';
import { Info } from '../pages/info/info';

@NgModule({
  declarations: [
    MyApp,
    MisTabs,
    Inicio,
    Listado,
    Info
  ],
  imports: [
    IonicModule.forRoot(MyApp)
  ],
  bootstrap: [IonicApp],
  entryComponents: [
    MyApp,
    MisTabs,
    Inicio,
    Listado,
    Info
  ],
  providers: [{provide: ErrorHandler, useClass: IonicErrorHandler}]
})
export class AppModule {}
```

Cambiar iconos de los tabs

Vamos a ver como podemos cambiar los iconos de los tabs para ello vamos ahora a ver el contenido de la plantilla **mis-tabs.html**:

```
<ion-tabs>
  <ion-tab [root]="tab1Root" tabTitle="Inicio" tabIcon="home"></ion-tab>
  <ion-tab [root]="tab2Root" tabTitle="Listado" tabIcon="text"></ion-tab>
  <ion-tab [root]="tab3Root" tabTitle="Info" tabIcon="stats"></ion-tab>
</ion-tabs>
```

Como podemos observar la página maestra de los tabs (**mis-tabs.html**) solamente contiene el contenedor de tabs que se define con la etiqueta **<ion-tabs>** y luego una etiqueta **<ion-tab>** por cada tab. También podemos observar que cada tab tiene el atributo **[root]** donde se le asigna una variable que si volvemos al controlador en **mis-tabs.ts** vemos que a cada una de esas variables (tab1Root, tab3Root y tab3Root) les hemos asignado una página que hemos importado. Por lo tanto al pulsar sobre un tab se carga el contenido de la página a la que apunta la variable que tiene asignada.

Luego tenemos el atributo **tabTitle** al que se le asigna el texto que se muestra en el tab, si solo queremos que se muestre el icono podemos eliminar o dejar vacío este atributo.

Y por último tenemos el atributo **tabIcon** y es aquí donde se le asigna el icono que se muestra en cada tab, si solo queremos que se muestre texto podemos eliminar o dejar vacío este atributo.

Si corres la aplicación en el navegador con **ionic serve –lab** puedes observar que en iOS los iconos son diferentes a los iconos de Android. Los iconos cambian dependiendo de la plataforma móvil, adaptandose al estilo propio de la plataforma. Esta es una de las ventajas de ionic 2, que no te tienes que preocupar por adaptar el estilo que tu app para cada plataforma ya que la mayoría los elementos se adaptan automáticamente al estilo propio de cada sistema operativo.

Probablemente te estarás preguntando, pero…¿de donde salen estos iconos?.La respuesta es simple, ionic viene de serie con una colección de los iconos mas comunes lista para ser usada solamente con asignar como hemos visto el nombre del icono al atributo tabIcon.

El listado de iconos disponibles lo podéis consultar el la documentación oficial de ionic 2 desde el siguiente enlace: ***https://ionicframework.com/docs/v2/ionicons/***.

Ejercicio

El colegio Manuel Belgrano de la ciudad de Tres de Febrero nos ha contratado para desarrollar una aplicación para que sus docentes puedan almacenar las notas de los alumnos de los cursos que disponen.

En esta unidad debemos generar la aplicación y generar los siguientes tabs:

- Inicio
- Mis cursos
- Contacto

En la página de inicio crear un enlace a una nueva página denominada **quienes somos** utilizando el componente **NavPush**

Bibliografía utilizada y sugerida

http://ionic.io/products/creator?utm_source=framework&utm_medium=whatsNext&utm_campaign=Creator%20CTA&_ga=1.243866311.353554560.1465009538

https://ionicframework.com/docs/cli/

http://blog.ionic.io/10-minutes-with-ionic-2-adding-pages-and-navigation/

https://www.joshmorony.com/a-simple-guide-to-navigation-in-ionic-2/

https://ionicframework.com/docs/api/components/nav/NavPush/

Lo que vimos:

En esta unidad vimos como generar tabs utilizando IONIC generator.

También vimos algunos elementos básicos de configuración para nuestra aplicación con IONIC

Lo que viene:

En la siguiente unidad veremos más directivas y componentes propios de IONIC que nos permitirá realizar aplicaciones con más funcionalidades.