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 2 componentes y mas IONIC generator

Presentación:

En la presente unidad analizaremos la documentación disponible en la página de IONIC.

Trabajaremos con sus componentes y directivas y así podremos mejorar nuestra aplicación.

Objetivos:

Que los participantes*:

- Conozcan directivas y componentes propios de IONIC
- Implementes directivas y componentes de IONIC
- Conozcan la documentación disponible en la web
- Mejoren la aplicación creada en la unidad anterior.

Bloques temáticos*:

- IONIC docs
- Angular 2
- UI Components
- Action sheets
- Alert
- Buttons
- Cards
- Checkbox
- Datetime
- FABs
- Grids
- Icons
- Inputs
- Lists
- Loading

- Menú
- Modal
- Más componentes

Consignas para el aprendizaje colaborativo

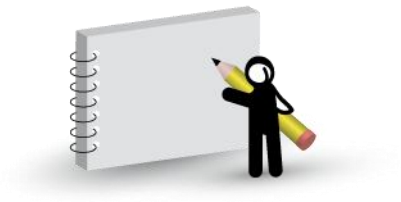
En esta Unidad los participantes se encontrarán con diferentes tipos de actividades que, en el marco de los fundamentos del MEC*, los referenciarán a tres comunidades de aprendizaje, que pondremos en funcionamiento en esta instancia de formación, a los efectos de aprovecharlas pedagógicamente:

- 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.



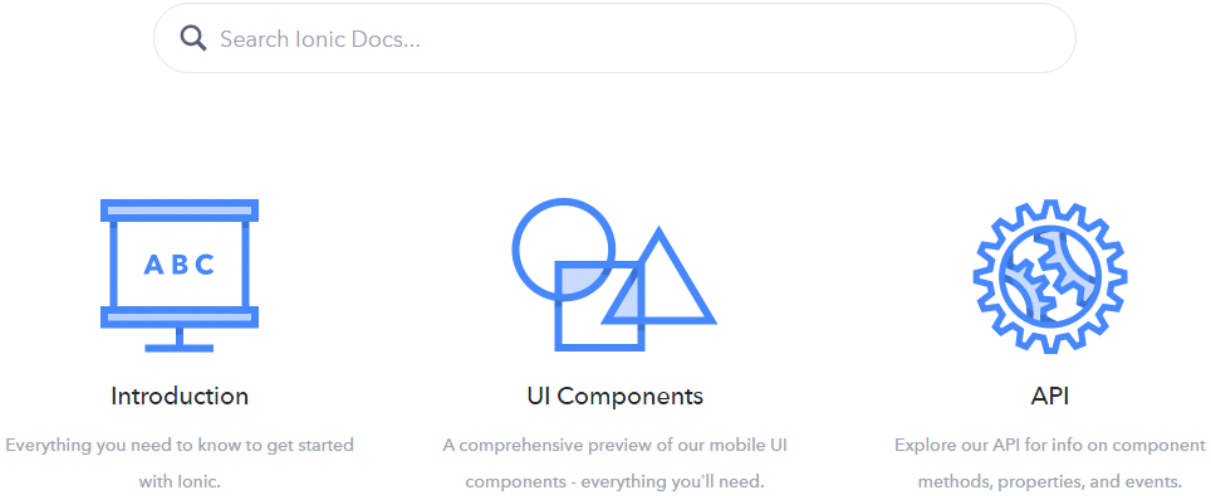
Tomen nota*

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 docs

Para ver la documentacion de IONIC debemos ingresar a:
http://ionicframework.com/docs/
Vamos a ver la siguiente pantalla:



Angular 2

Directiva Click

Si queremos que al hacer un clic en un button por ejemplo, se ejecute un determinado método de nuestro constructor debemos declarar la directive **(clic)** en nuestro elemento button.

Ejemplo:

```
<ion-content padding>
  <button ion-button (click)="presentActionSheet" >Action Sheet</button>
</ion-content>
```

UI Components

Debemos acceder a: ***http://ionicframework.com/docs/components/***

Los componentes son porciones de código de alto nivel que permiten que el desarrollo de las interfaces de nuestras aplicaciones sea mucho más sencillo y estandarizado.

IONIC nos provee entre otras cosas de modales, popups, cartas, etc.

Action sheets

Es un conjunto de opciones que nos permiten aceptar o cancelar una acción.

La misma se despliega en la parte de debajo de la pantalla de nuestro dispositivo.

1. Debemos importar el componente dentro de la pagina donde lo queremos utiliza, por ejemplo en **scr/app/componentes/componentes.ts** incluimos el siguiente código:

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

2. Luego en el constructor debemos recibir como parámetro el componente:

```
constructor(public navCtrl: NavController, public navParams: NavParams,public actionSheetCtrl: ActionSheetController) {
```

3. Dentro de la clase del modulo principal (debajo del constructor) declaramos el método que al ser ejecutado mostrara las action sheet:

```
presentActionSheet() {
  let actionSheet = this.actionSheetCtrl.create({
    title: 'Modify your album',
    buttons: [
      {
        text: 'Destructive',
        role: 'destructive',
        handler: () => {
          console.log('Destructive clicked');
        }
      },{
        text: 'Archive',
        handler: () => {
          console.log('Archive clicked');
        }
      },{
        text: 'Cancel',
        role: 'cancel',
        handler: () => {
          console.log('Cancel clicked');
        }
      }
    ]
  });
  actionSheet.present();
}
```

Si observamos en los parámetros de buttons se definirán todas las acciones que queramos visualizar en nuestra aplicación.

4. Por último creamos un botón y le asociamos a la directiva clic del mismo el llamado al método creado.

```
<button ion-button (click)="presentActionSheet()" >Action Sheet</button>
```

Alert

Basic Alert

1. Importar el componente AlertController

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

2. Incluir en el constructor la clase AlertController

```
constructor(public navCtrl: NavController, public navParams: NavParams,public alertCtrl: AlertController) {
```

3. Declarar el método **showAlert**

```
showAlert() {
  let alert = this.alertCtrl.create({
    title: 'New Friend!',
    subTitle: 'Your friend, Obi wan Kenobi, just accepted your friend requ
    buttons: ['OK']
  });
  alert.present();
}
```

4. Desde el html realizar el llamado al evento declarado en el punto anterior

```
<button ion-button (click)="showAlert()" >Basic Alert</button>
```

Confirmation Alert

1. Lo único que debemos modificar en caso de querer realizar un alert con confirmación del usuario, es el método del controlador. Por ejemplo:

```
showConfirm() {
  let confirm = this.alertCtrl.create({
    title: 'Use this lightsaber?',
    message: 'Do you agree to use this lightsaber to do good across the intergalactic galaxy?',
    buttons: [
      {
        text: 'Disagree',
        handler: () => {
          console.log('Disagree clicked');
        }
      },
      {
        text: 'Agree',
        handler: () => {
          console.log('Agree clicked');
        }
      }
    ]
  });
  confirm.present();
}
```

*En este caso vemos que al parámetro **buttons** se le pasa un array pero en lugar de definir con un string el label del botón, definimos mediante json su label y su handler, es decir el evento que se ejecutara en caso de hacer clic.*

Radio Alert

En este ejemplo vemos la utilización de un alert con distintas opciones para seleccionar.

```
showRadio() {
  let alert = this.alertCtrl.create();
  alert.setTitle('Lightsaber color');

  alert.addInput({
    type: 'radio',
    label: 'Blue',
    value: 'blue',
    checked: true
  });
  alert.addInput({
    type: 'radio',
    label: 'Red',
    value: 'red',
    checked: true
  });

  alert.addButton('Cancel');
  alert.addButton({
    text: 'OK',
    handler: data => {
      this.testRadioOpen = false;
      this.testRadioResult = data;
    }
  });
  alert.present();
}
```

*En este ejemplo vemos que con el método **setTitle** podemos configurar el titulo de nuestro alert.*

*Por otro lado podemos incluir inputs al mismo con la utilización de **addInput** y pasándole las configuraciones que queramos por parámetro.*

*Por último añadimos los botones de diferente manera utilizando **addButton** y a este le pasamos su label y su handler.*

Buttons

La utilización de buttons es bastante sencilla, podremos ver todas sus posibles utilizaciones en ***http://ionicframework.com/docs/components/#buttons***

También podremos ver algunos ejemplos aplicaciones en la app mobile brindada.

Cards

Las cards son elementos contenedores HTML.

Su utilización es bastante sencilla podemos ver los diferentes tipos en: ***http://ionicframework.com/docs/components/#cards***

También podremos ver algunos ejemplos aplicaciones en la app mobile brindada.

Checkbox

Ver más información en:

http://ionicframework.com/docs/components/#checkbox

También podremos ver algunos ejemplos aplicaciones en la app mobile brindada.

Datetime

Ver más información en:

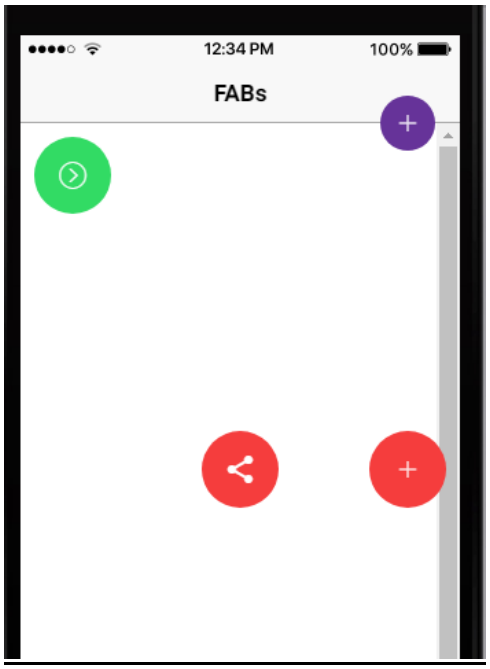
http://ionicframework.com/docs/components/#datetime

FABs

Estos son los famosos elementos flotantes de material design

Ver más información en:

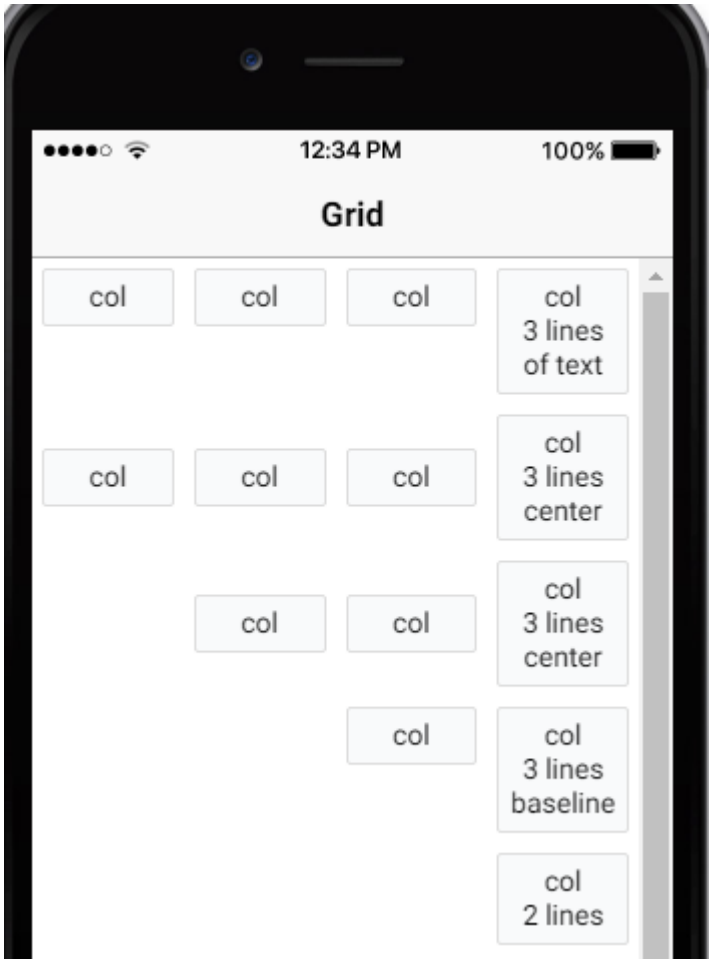
http://ionicframework.com/docs/components/#fabs



Grids

Mediante los grids podremos definir las filas y columnas que deseamos insertar.

Para lo cual debemos insertar las clases correspondientes para definir el número de filas y columnas que tendremos.



Ver más información en:

http://ionicframework.com/docs/components/#grid

Icons

IONIC nos ofrece más de 700 iconos.

Podremos utilizar los mismos de la siguiente manera

```
<ion-icon name="heart"></ion-icon>
```

Podemos especificar un icono para cada plataforma, por ejemplo:

```
<ion-icon ios="logo-apple" md="logo-android"></ion-icon>
```

También podemos definir iconos de manera dinámica, por ejemplo:

```
<ion-icon [name]="myIcon"></ion-icon>
```

```
export class MyFirstPage {
  // use the home icon
  myIcon: string = "home";
}
```

Podemos ver más en: ***http://ionicframework.com/docs/components/#icons***

Inputs

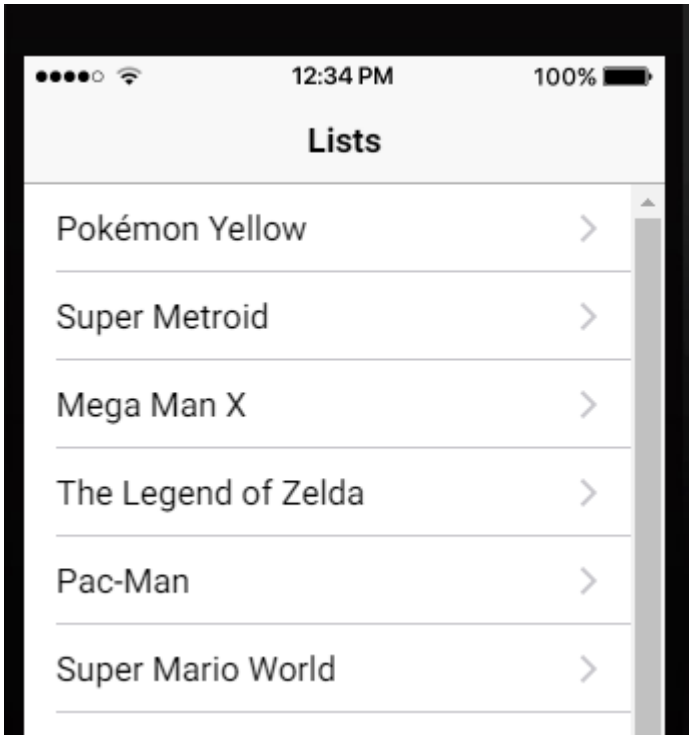
Los inputs los podemos insertar mediante **ion-inputs**. Por ejemplo:

```
<ion-list>

  <ion-item>
    <ion-label fixed>Username</ion-label>
    <ion-input type="text" value=""></ion-input>
  </ion-item>

  <ion-item>
    <ion-label fixed>Password</ion-label>
    <ion-input type="password"></ion-input>
  </ion-item>

</ion-list>
```

Tenemos diferentes maneras de configurar los inputs, las cuales podemos ver en: ***http://ionicframework.com/docs/components/#inputs***

Lists

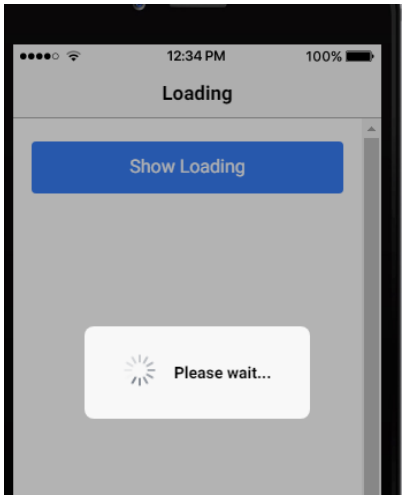
Las listas son usadas para mostrar filas de información como puede ser contactos, playlist o un menú.



Podemos ver más información en: ***http://ionicframework.com/docs/components/#lists***

Loading

El loading nos permite impeder que el usuario siga interactuando con la aplicación hasta bien no haya terminado la actividad ejecutada.



Para su implementación debemos realizar lo siguiente:

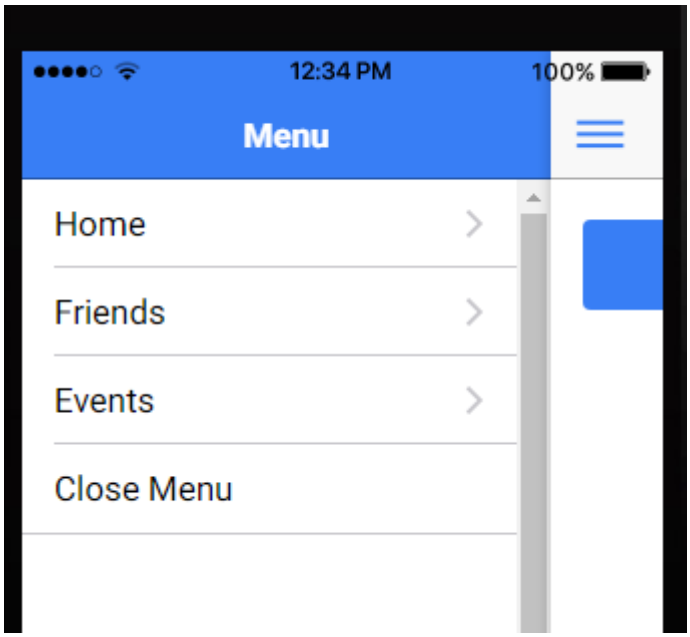
```
import { LoadingController } from 'ionic-angular';

export class MyPage {
  constructor(public loadingCtrl: LoadingController) {
  }

  presentLoading() {
    let loader = this.loadingCtrl.create({
      content: "Please wait...",
      duration: 3000
    });
    loader.present();
  }
}
```

La función presentLoading definirá un elemento de tipo loadingCtrl (Ver que para poder usarlo debemos importar el componente LoadingController).

Recibirá como parámetro el contenido y la duración.

Menú

Mediante este componente podremos hacer el típico menú hamburguesa utilizado en las aplicaciones móviles.



Para configurar el menú debemos realizar los siguientes pasos:

1. Agregar el siguiente codigo en **app.html**

```
<ion-menu [content]="mycontent">
  <ion-header>
    <ion-toolbar>
      <ion-title>Menú</ion-title>
    </ion-toolbar>
  </ion-header>
  <ion-content>
    <ion-list>
      <button ion-item (click)="openPage(navigation)">
        NavPush
      </button>
      <button ion-item (click)="openPage(signupPage)">
        Signup
      </button>
    </ion-list>
  </ion-content>
</ion-menu>

<ion-nav #mycontent [root]="rootPage" swipeBackEnabled="false"></ion-nav>
```

Con la directiva **ion-menu** definimos el contenido del menú, bajo el identificador **mycontent**. Luego en el elemento **ion-nav** hacemos referencia al identificador del menú con **#mycontent**

2. En **app.component.ts** debemos agregar lo siguiente:

```
import { Platform, Nav } from 'ionic-angular';
```

```
export class MyApp {
  @ViewChild(Nav) nav: Nav;
```

```
openPage(page) {
  // Reset the content nav to have just this page
  // we wouldn't want the back button to show in this scenario

  this.nav.push(page);
}
```

La función **openPage** será la encargada de redireccionar a la página que reciba como parámetro.

3. Por último en cada página en la cual queramos utilizar el menú debemos agregar lo siguiente:

```
<ion-navbar>
  <button ion-button menuToggle>  <ion-icon name="menu"></ion-icon> </button>
  <ion-title>Inicio</ion-title>
</ion-navbar>
```

   La directiva **menuToggle** servirá para cuando hagamos clic en el botón se despliegue el menú.

Para ver más información: ***http://ionicframework.com/docs/components/#menus***

Modal

El modal es una pantalla que se abrirá por encima de la pantalla principal.

Podemos ejecutar un modal de la siguiente manera:

```
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();
  }
}
```

Importamos el componente modal.

Luego debemos declarar un método que al hacer clic ejecute el modal utilizado **modalCtrl.create**. Lo que recibe este método como parámetro es el componente (pagina) que se ejecutar dentro del modal.

Más componentes

IONIC provee un montón de componentes, para ver la utilización de más componentes:

http://ionicframework.com/docs/components

Ejercicio

Continuando con el desarrollo del ejercicio anterior, realizaremos lo siguiente:

- Realizar el formulario de contacto:
  - Campos
    - Nombre y apellido
    - Teléfono
    - Email
    - Descripción
  - Al hacer clic en el formulario ejecutar un método que muestra por consola (console.log) los campos que fueron completados.
- Realizar formulario de login en la página de inicio:
  - Campos
    - Email
    - Password
  - Desarrollar un método que valide ese email y contraseña (Hacer una validación fija con un email y password elegido).

- Agregar un nuevo tab que diga registro
    - Formulario con los siguientes campos:
        - Nombre
        - Apellido
        - Email
        - Password
        - Confirmar password
    - Realizar las validaciones correspondientes, sin almacenar en ninguna base de datos los campos ingresados.

En todos los casos utilizar los elementos vistos como (inputs, checkbox, buttons, modals, loading, etc)

Bibliografía utilizada y sugerida

http://ionicframework.com/docs/

http://ionicframework.com/docs/components/#outlined-buttons

http://ionicons.com/

Lo que vimos:

En esta unidad vimos directivas y componentes disponibles en IONIC framework.

Realizamos una breve guía de la documentación de IONIC para darte las herramientas necesarias para realizar la aplicación que necesites.

Lo que viene:

En la próxima unidad continuaremos viendo parte de la API de IONIC 2 y modificación de CSS.

También veremos distintos mecanismos para realizar la persistencia de datos (local y sesión storage)