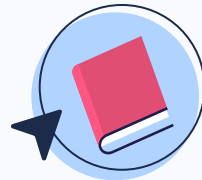


DuocUC

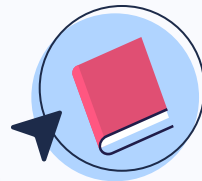


PGY4121: Ionic Crear App IOS, Android



Docente: @VivitaSol
v.pobletel@profesor.duoc.cl

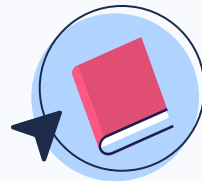
Temas Puntuales



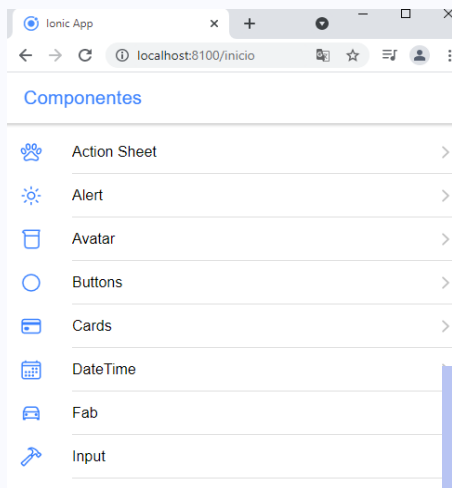
Revisaremos componentes de Ionic creando ejercicios simples para cada componente.



Componentes



Ion-input

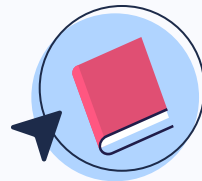


- ❑ Los elementos más comunes son: text, password, email, number, search, tel, url.
- ❑ Crearemos una nueva página dentro de pages:
ionic g page pages/input
- ❑ Al momento de crear /input se agrega a la app.routing.module.ts.
- ❑ Agregaremos al archivo inicio.page.ts el nuevo componente.
- ❑ Recuerde que redirectTo es la dirección registrada en app.routing.module.ts.
- ❑ Implementamos icon-buttons en input.page.html para volver al inicio.

```
<ion-toolbar>
  <ion-buttons slot="start" color="primary">
    <ion-back-button defaultHref="/" color="primary"></ion-back-button>
  </ion-buttons>
  <ion-title color="primary">Input</ion-title>
</ion-toolbar>
```

Componentes

Ion-input



```
<ion-content>

  <ion-list>
    <ion-list-header>
      <ion-label>Input Estándar</ion-label>
    </ion-list-header>

    <ion-item>
      <ion-label>Nombre: </ion-label>
      <ion-input type="text" placeholder="Digite un nombre" [(ngModel)]="nombre">
      </ion-input>
    </ion-item>
  </ion-list>
</ion-content>
```

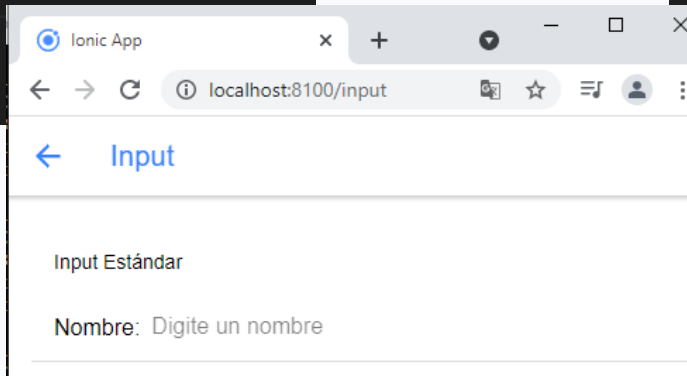
- ❑ Insertamos un elemento de tipo input.
- ❑ Para acceder al valor ingresado, utilizamos expresión angular ngModel.
- ❑ Definimos el parámetro “nombre” en input.page.ts.

```
export class InputPage implements OnInit {

  nombre: string = '';

  constructor() {}

  ngOnInit(): void {}
}
```



Componentes

Ion-input



```
io.page.ts U  input.page.html U  TS input.page.ts U
app > pages > input > input.page.html > ion-content.ion-padding

<form #formulario="ngForm" (ngSubmit)="onSubmit()">
  <ion-list-header>
    <ion-label>Formulario Válido: {{formulario.valid}} </ion-label>
  </ion-list-header>

  <ion-item>
    <ion-label>Email: </ion-label>
    <ion-input type="email" placeholder="Email"
      name="email" [(ngModel)]="usuario.email"
      pattern="^[a-zA-Z0-9_\\-\\.]+@[a-zA-Z0-9_\\-\\.]+\\.([a-zA-Z]{2,5})$"
      required>
    </ion-input>
  </ion-item>

  <ion-item>
    <ion-label>Password: </ion-label>
    <ion-input type="password" placeholder="Contraseña"
      name="password" [(ngModel)]="usuario.password"
      required>
    </ion-input>
  </ion-item>

  <ion-button [disabled]="formulario.invalid"
    type="submit" expand="block">
    Click me
  </ion-button>

</form>
</ion-content>
```

- ❑ Definimos un formulario como #formulario.
- ❑ Se implementa método angular (ngSubmit)
- ❑ Verificamos la validación del formulario (valid).
- ❑ Implementamos dos elementos, input y tipo password.
- ❑ En input definimos un pattern para validar el ingreso de información.
- ❑ Angular requiere para validar los datos ingresados, vincular ambos inputs al formulario.
- ❑ Implementamos una clase llamada usuario en input.page.ts. Por medio de ngModel enlazamos los input al formulario para validar.
- ❑ El botón se habilita cuando se ingresan datos correctos.

Componentes

Ion-input



```
io.page.ts U  input.page.html U  TS input.page.ts U X
app > pages > input > TS input.page.ts > InputPage > onSubmit
export class InputPage implements OnInit {

  nombre: string='';

  usuario = {
    email:'',
    password:''
  }

  constructor() {}

  ngOnInit() {}

  onSubmit(){
    console.log('submit');
    console.log(this.usuario);
  }
}
```

Back Input

Input Estándar

Nombre: Vivita Sol

Formulario Válido: true

Email: vivitasol@cachupin.cl

Password:

Click me

localhost:8100/input

iPhone 6/7/8 375 x 667 74%

Back Input

Input Estándar

Nombre: Digite un nombre

Formulario Válido: false

Email: Email

Password: Contraseña

Click me

Elements Console

html.plt-iphone.plt-ios.plt-mobile.plt-mobilewebios.hydrated body

Styles Computed Layout Event Listeners

Filter :hov .cls +

element.style { }

Console Network conditions Issues

Angular is running in development mode. core.js:28072
Call enableProdMode() to enable production mode.
[WDS] Live Reloading enabled. index.js:52

html.plt-iphone.plt-ios.plt-mobile.plt-mobilewebios.hydrated body

Styles Computed Layout Event Listeners

Filter :hov .cls +

element.style { }

Console Network conditions Issues

Angular is running in development mode. core.js:28072
Call enableProdMode() to enable production mode.
[WDS] Live Reloading enabled. index.js:52

submit input.page.ts:23

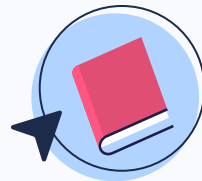
input.page.ts:23

{email: "vivitasol@cachupin.cl", password: "1234567"}

Docente: @VivitaSol
v.pobletel@profesor.duoc.cl

Componentes

Ion-menu



- ❑ Accedemos a la documentación oficial y trasladamos un ejemplo de menú en el **archivo app.component.html**.
- ❑ En el archivo inicio.page.html agregamos un botón para activar el menú.
- ❑ En el archivo inicio.page.ts, implementamos el método que active el menú a través del botón.

```
component.html M • inicio.page.html U x TS inicio.page.ts U
pp > pages > inicio > inicio.page.html > ion-header > ion-toolbar > ion-button
<ion-header>
  <ion-toolbar>
    <ion-buttons slot="start">
      <ion-button (click)="mostrarMenu()" color="primary">
        <ion-icon slot="icon-only" name="menu-outline">/ion-icon>
      </ion-button>
    </ion-buttons>
    <ion-title color="primary">Componentes</ion-title>
  </ion-toolbar>
</ion-header>
```

Usage

ANGULAR JAVASCRIPT REACT STENCIL

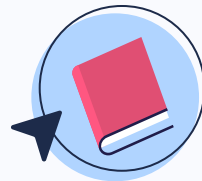
```
<ion-menu side="start" menuId="first" contentId="main">
  <ion-header>
    <ion-toolbar color="primary">
      <ion-title>Start Menu</ion-title>
    </ion-toolbar>
  </ion-header>
  <ion-content>
    <ion-list>
      <ion-item>Menu Item</ion-item>
      <ion-item>Menu Item</ion-item>
      <ion-item>Menu Item</ion-item>
    </ion-list>
  </ion-content>
</ion-menu>
```

```
component.html M • inicio.page.html U TS inicio.page.ts U
pp > app.component.html > ion-app
<ion-app>
  <ion-menu side="start" menuId="first" contentId="main">
    <ion-header>
      <ion-toolbar color="primary">
        <ion-title>Start Menu</ion-title>
      </ion-toolbar>
    </ion-header>
    <ion-content>
      <ion-list>
        <ion-item>Menu Item</ion-item>
        <ion-item>Menu Item</ion-item>
        <ion-item>Menu Item</ion-item>
        <ion-item>Menu Item</ion-item>
      </ion-list>
    </ion-content>
  </ion-menu>
  <ion-router-outlet id="main">/ion-router-outlet<
</ion-app>
```

```
constructor(private menuController: MenuController) {}
ngOnInit() {
}
mostrarMenu(){
  this.menuController.open('first');
}
```

Componentes

Ion-menu



- ❑ Implementaremos ahora las opciones de inicio.page.html en el menú (app.component.html).
- ❑ Implementamos el arreglo de inicio.page.ts en app.component.ts.

```
app.component.ts - componentes - Visual Studio Code
app > TS app.component.ts > AppComponent > constructor
import { Component } from '@angular/core';

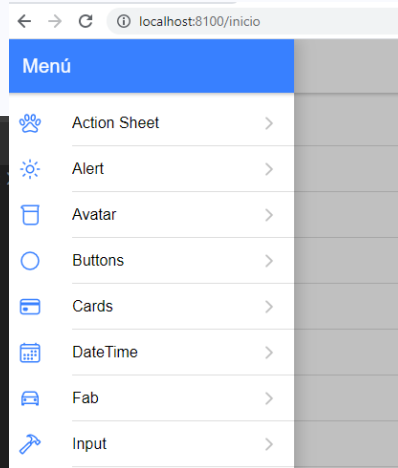
interface Componente {
  icon:string;
  name:string;
  redirectTo:string;
}

@Component({
  selector: 'app-root',
  templateUrl: 'app.component.html',
  styleUrls: ['app.component.scss'],
})
export class AppComponent {
  componentes : Componente[] = [
    {
      icon: 'paw-outline',
      name: 'Action Sheet',
      redirectTo: '/action-sheet'
    },
    {
      icon: 'sunny-outline',
      name: 'Alert',
      redirectTo: '/alert'
    },
    {
      icon: 'beaker-outline',
      name: 'Avatar',
      redirectTo: '/avatar'
    },
  ],
}
```

```
component.html M x inicio.page.html U TS inicio.page.ts U
app > TS app.component.ts > AppComponent > constructor
<ion-app>
  <ion-menu side="start" menuId="first" contentId="main">
    <ion-header>
      <ion-toolbar color="primary">
        <ion-title>Menú</ion-title>
      </ion-toolbar>
    </ion-header>
    <ion-list>
      <ion-item *ngFor="let c of componentes" [routerLink]="c.redirectTo" detail=true>
        <ion-icon slot="start" color="primary"></ion-icon>
        {{ c.name }}
      </ion-item>
    </ion-list>
  </ion-menu>
</ion-app>
```

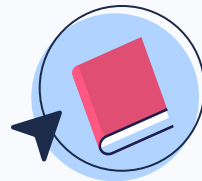
- ❑ Para que el botón de menú se oculte, debemos **utilizar <ion-menú-toggle>**:










```
<ion-menu-toggle *ngFor="let c of componentes">
  <ion-item [routerLink]="c.redirectTo"
    detail=true>
```



Componentes

Ion-tabs



Componentes		
	Action Sheet	>
	Alert	>
	Avatar	>
	Buttons	>
	Cards	>
	DateTime	>
	Fab	>
	Input	>
	Tabs	>

- ❑ Crearemos una nueva página dentro de pages:
ionic g page pages/tabs
- ❑ Al momento de crear /input se agrega a la app.routing.module.ts.
- ❑ Agregaremos al archivo inicio.page.ts el nuevo componente.
- ❑ Recuerde que redirectTo es la dirección registrada en app.routing.module.ts.
- ❑ Implementamos icon-buttons en tabs.page.html para volver al inicio.

```
<ion-toolbar>
  <ion-buttons slot="start" color="primary">
    <ion-back-button defaultHref="/" color="primary"></ion-back-button>
  </ion-buttons>
  <ion-title color="primary">Input</ion-title>
</ion-toolbar>
```

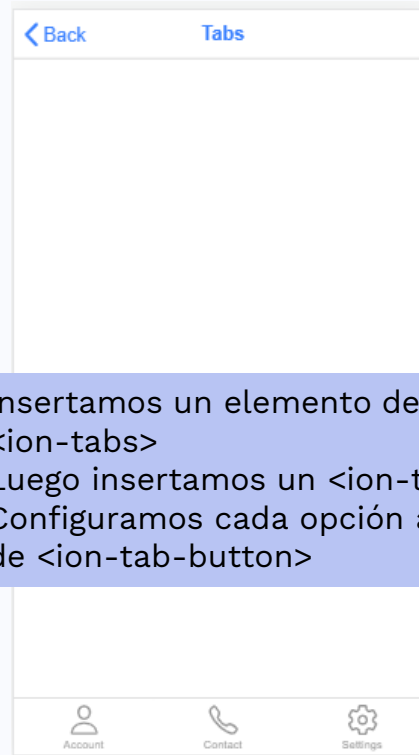
Componentes

Ion-tabs

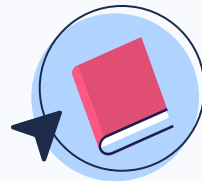


```
tabs.page.html U X
c > app > pages > tabs > < tabs.page.html > ion-content
9
10 <ion-content>
11   <ion-tabs>
12     <ion-tab-bar slot="bottom">
13       <ion-tab-button tab="account">
14         <ion-icon name="person-outline"></ion-icon>
15         <ion-label>Account</ion-label>
16       </ion-tab-button>
17
18       <ion-tab-button tab="contact">
19         <ion-icon name="call-outline"></ion-icon>
20         <ion-label>Contact</ion-label>
21       </ion-tab-button>
22
23       <ion-tab-button tab="settings">
24         <ion-icon name="settings-outline"></ion-icon>
25         <ion-label>Settings</ion-label>
26       </ion-tab-button>
27
28     </ion-tab-bar>
29   </ion-tabs>
30 </ion-content>
```

- ☐ Insertamos un elemento de tipo `<ion-tabs>`
- ☐ Luego insertamos un `<ion-tab-bar>`
- ☐ Configuramos cada opción a través de `<ion-tab-button>`



Componentes Ion-tabs

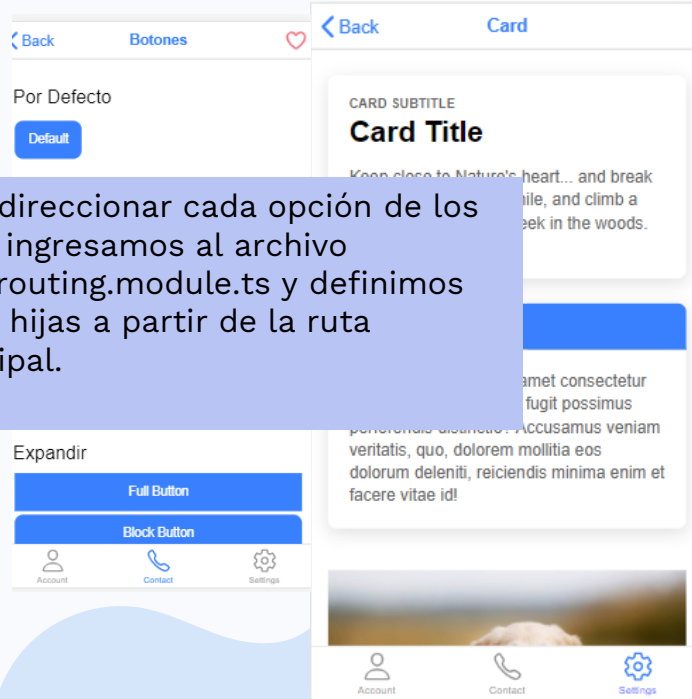


```
page.html U TS tabs-routing.module.ts U X
pp > pages > tabs > TS tabs-routing.module.ts > routes
import { NgModule } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';

import { TabsPage } from '../tabs.page';

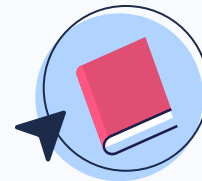
const routes: Routes = [
  {
    path: '',
    redirectTo: '/tabs/account',
    pathMatch: 'full',
  },
  {
    path: '',
    component: TabsPage,
    children: [
      {
        path: 'account',
        loadChildren: () => import('../avatar/avatar-routing.module').then(m => m.AvatarPageRoutingModule)
      },
      {
        path: 'contact',
        loadChildren: () => import('../botones/botones-routing.module').then(m => m.BotonesPageRoutingModule)
      },
      {
        path: 'settings',
        loadChildren: () => import('../card/card-routing.module').then(m => m.CardPageRoutingModule)
      }
    ]
  }
];
```

- ❑ Para direccionar cada opción de los tabs, ingresamos al archivo tabs.routing.module.ts y definimos rutas hijas a partir de la ruta principal.



Docente: @VivitaSol
v.pobletel@profesor.duoc.cl

¡Gracias!



¿Alguna pregunta?

- v.pobletel@profesor.duoc.cl



Docente: @VivitaSol
v.pobletel@profesor.duoc.cl