|  |
| --- |
| http://www.becas.sep.gob.mx/images/logo.png  TECNOLÓGICO NACIONAL DE MÉXICO  Instituto Tecnológico de Chihuahua II |
| Tabs & Navegation (abstract) |
| Ejemplo de states abstractos con tabs |
|  |
|  |
| 08 de Noviembre del 2016  Unidad II  Edwin Daniel Romero Terrazas  12550540  Programación web para dispositivos móviles |

Contenido

[Introducción 2](#_Toc466443583)

[Desarrollo 3](#_Toc466443584)

[Paso 1.- 3](#_Toc466443585)

[Paso 2.- 3](#_Toc466443586)

[Paso 3.- 3](#_Toc466443587)

[Conclusión. 8](#_Toc466443588)

# 

# Introducción

El objetivo de esta práctica es conocer, entender y poner en práctica el conocimiento acerca de los states abstractos, utilizando de manera simple y clara tabs. Prácticamente es realizar “a mano” desde cero la starter app que incluye ionic por defecto que se crea con “ionic start name tabs”.

# Desarrollo

## Paso 1.-

Abrimos una terminal y tecleamos el siguiente comando.



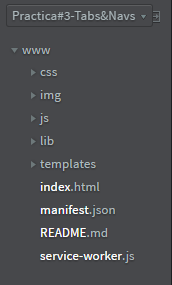
## Paso 2.-

Creamos un proyecto con el siguiente comando, el cual nos creara un proyecto en blanco solo con los plugins necesarios para la posterior ejecución.

$ ionic start Tabs&Navs blank

## Paso 3.-

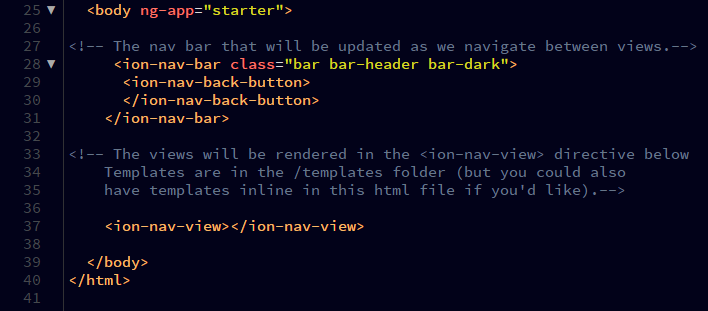
Con un editor de texto (Brackets) abrimos la carpeta que contiene todos los archivos de nuestro proyecto anteriormente creado.



Paso 4.-

Para comenzar con nuestro código, abriremos nuestro index.html.

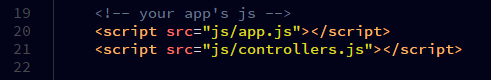
Tenemos que borrar todo el contenido del contenedor <body> sin quitar la directiva ng-app que contiene para así poder escribir lo siguiente:



Coloqué comentarios antes de cada pieza de código para mejorar aún más su comprensión.

Cómo dice, <ion-nav-bar> creará una barra que se actualizará cuando cambiemos entre los views que crearemos más adelante.

Luego <ion-nav-view> será donde se renderizará todo.



Por último, recordemos que tenemos que adjuntar nuestro archivo creado anteriormente como script.

Paso 5.-

Iremos a nuestro app.js (archivo creado por default) para crear los states, los states controlará la navegación en nuestra app. Crearemos un state abstracto que será el ‘padre’ de los demás states.

Se agregarán también un par de dependencias que hacen que sea posible el correcto funcionamiento de los states las cuales son $stateProvider y $urlRouterProvider.

Una vez más se agregaron comentarios para maximizar su comprensión.



Notese que dentro de cada state declarado, se asigna un controlador; controladore que crearemos más adelante en nuestro controller.js. Ademas se muestra el url del template al que va dirigido.

Por último se declara $urlRouterProvider.otherwise(‘’) donde se coloca como expresión el url por defecto al que se redirigirá la app en caso de que no se haga conexión con algún state.

Paso 5.-

Ya tenemos los states creados y hemos configurado a donde nos llevará cada uno, pero esos archivos aun no tienen nada, comenzaremos por ahí.

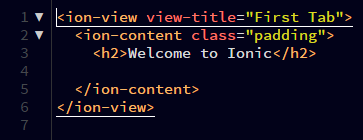
El más importante es nuestro template tabs.html pues aquí se confiugaran cada una de las tabs que vayamos a utilizar en nuestra app.

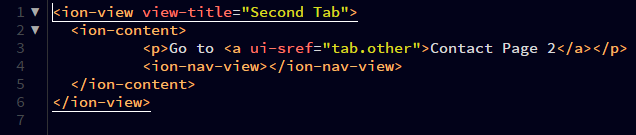


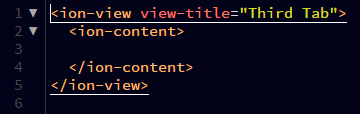
<ion-tabs> será lo que nos creará en sí las tabs, lo que está dentro del class=”” es la configuración del estilo de las tabs, así como el color de la letra y del fondo (todo con bootstrap)

Dentro del contenedor anterior, se crearan <ion-tab> que crea individualmente cada tab, en donde le configuramos el títuo, el ícono de activo e inactivo, así como un href con la dirección url y el nombre de view con <ion-nav-view> (Note que los nombres concuerdan con lo previamente configurado en nuestos states).

Luego mostraremos los otros 3 templates que son donde se configura cada una de las tabs: first-tab.html, second-tab.html y third-tab.html respectivamente.





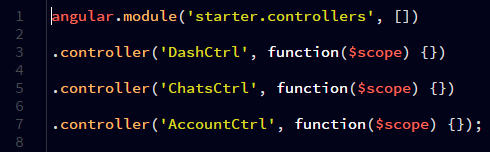


<ion-view> contiene toda la vista de esa tab y el contenedor <ion-content> lo utilizamos para almacenar todo el contenido que se renderizará de cada tab.

Paso 6.-

Como dijimos anteriormente, ahora crearemos los controladores que se mencionan en cada state y que harán posible la renderizacion de cada tab.

Esto lo haremos en el archivo que creamos llamado controllers.js



Se creó un módulo llamado starter.controllers; ahí declaramos todos los controladores con la directica $scope que hace posible la renderización.

Pero para que se puedan conectar los controladores con los states es necesario injectar esta dependencia en nuestro app.js, así como se muestra a continuación:



Y con esto terminamos, funcionando de manera correcta.

# Conclusión.

Se desarrolló esta práctica para lograr una aplicación que mostrara tabs para así poder entender el concepto y funcionamiento de states y states abrstractos; utilizándolos en conjunto de controladores, injección de dependencias y directivas de ionic. Además se logró el objetivo de manera estética también pues se utilizaron estilos de bootstrap para complementar la aplicación.

