Can use jasmine as the test framework and karma as the test runner.

Npm install –save-dev jasmine karma karma-jasmine karma-firefox-launcher grunt-cli grunt-karma

karma-unit.conf.js

Angular 4 app :  
index includes cordova .

App folder :

main.ts bootstrap our module.

**app.module.ts**

import { NgModule, ErrorHandler } from '@angular/core';

import { IonicApp, IonicModule, IonicErrorHandler } from 'ionic-angular';

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

import { AboutPage } from '../pages/about/about';

import { ContactPage } from '../pages/contact/contact';

import { HomePage } from '../pages/home/home';

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

import {ReactiveFormsModule} from '@angular/forms';

@NgModule({

declarations: [

MyApp,

AboutPage,

ContactPage,

HomePage,

TabsPage

],

imports: [

IonicModule.forRoot(MyApp),

ReactiveFormsModule

],

bootstrap: [IonicApp],

entryComponents: [

MyApp,

AboutPage,

ContactPage,

HomePage,

TabsPage

],

providers: [{provide: ErrorHandler, useClass: IonicErrorHandler}]

})

export class AppModule {}

Above class is our app module

**bootstrap: [IonicApp]**

Here we are bootstrapping our ionic app, that we have imported from 'ionic-angular’.

**IonicModule.forRoot(MyApp) :** See this line

**IonicModule** is provided by the ionic 2. It also contains the FormsModule and the HttpModule. We do not need to import it separately. **IonicModule contains all of that.**

MyApp is our component that is of course normal. It configures ionic 2 to wrap itself around our app component.It uses our own app component.

**entryComponents**

This is something new that we found here. It is normally used in angular web app. It is used to provide some factory for our components. In **declarations** we name our components so that the angular get aware of them.

Note : In angular 2 we generally use the routing for our pages navigation , but in ionic 2 we have separate folder pages. Inside the pages folder we have different pages such as home.page :

<ion-header>

<ion-navbar>

<ion-title>Home</ion-title>

</ion-navbar>

</ion-header>

<ion-content padding>

<h2>Welcome to Ionic!</h2>

<p>

This starter project comes with simple tabs-based layout for apps

that are going to primarily use a Tabbed UI.

</p>

<p>

Take a look at the <code>src/pages/</code> directory to add or change tabs,

update any existing page or create new pages.

</p>

</ion-content>

In ionic2 we have different idea for navigating through the pages.

**How navigation works in ionic 2 :**

What are pages in ionic and how they are rendered?

In ionic pages are just like the components in angular 2.

Note : Ionic 2 does not use the angular 2 router. It has it’s own stack of pages. The stack of pages needs to be initialized. The imp thing to note is that we always see the top most page inside the stack . **Stack follows LIFO** : **last in first out**

We are then able to push these pages within the stack. The important concept here is that we always see the topmost page.

There might be multiple page within the stack but we will see the top page within stack.

We can also go back by popping out the page.

**Initializing**

**Let us now see app.html**

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

Here ion-nav is readymade component shipped along with the ionic 2 framework. It initializes the stack of pages. This is because we are initializing IonicApp in app.module.ts

**bootstrap: [IonicApp]**

The bottom page of the stack is the first page to be displayed on our app. Now we can see the definition of rootpage inside **app.component.ts**

import { Component } from '@angular/core';

import { Platform } from 'ionic-angular';

import { StatusBar, Splashscreen } from 'ionic-native';

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

@Component({

templateUrl: 'app.html'

})

export class MyApp {

**rootPage = TabsPage;**

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();

});

}

}

The above TabsPage imported : **import { TabsPage } from '../pages/tabs/tabs'**

For now we are at the position that we can create more pages for navigation , so let us do the same. Create our first custom page

**ionic generate page users**

By above command new component folder gets created named users. Let us see the user.html

<ion-header>

<ion-navbar>

<ion-title>users</ion-title>

</ion-navbar>

</ion-header>

<ion-content padding>

….

</ion-content>

We can add the footer adding following line :

**<ion-footer>**

**<h1>The footer</h1>**

**</ion-footer>**

Let us add some users buttons on the users page.

<ion-content padding> //here padding is a directive by ionic-2

<button ion-button>User ‘Mohan’</button>

<button ion-button>User ‘Pankaj’</button>

</ion-content>

**Note :** ion-button is special directive from ionic2.

Now we need to load the users page. By default home page will come first.

So, let us go to the home page. We will do some changes inside the ion-content section.

**<button ion-button (click)=”users()”>Users</ion-button>**

Now let us create a function **users()** inside **HomeController.** Also in HomeController we have **NavController** injected there. It is shipped along with ionic 2. **NavController is used to manage the stack of pages.** It is used to switch between the pages.

Example :

**a)this. navCtrl.push() b)this. navCtrl.pop()**

import { Component } from '@angular/core';

import { NavController } from 'ionic-angular';

import {UsersPage} from ‘../users/users’;l

@Component({

selector: 'page-home',

templateUrl: 'home.html'

})

export class HomePage {

constructor(public navCtrl: NavController) {

}

users(){

**this. navCtrl.push(UsersPage);**

}

}

Now let us declare the new page (**UsersPage**) inside the **app.module.ts**

Inside the declarations.

declarations: [

MyApp,

AboutPage,

**UsersPage**,

ContactPage,

HomePage,

TabsPage

]

Summary : Now when we start our app the home page gets loaded first. On the home page when we click on the users buttion we will be rediredted to users page.