**Ionic 学习笔记**

# 1、环境搭建

## 1.1 安装jdk

## 1.2 安装nodeJS

## 1.3 安装npm

## 1.4 安装ionic 和cordova

# 2、开始创建ionic应用

## 2.1 使用ionic命令行创建ionic应用

### 2.1.1 命令

$ **ionic** **start** **MyIonicProject** **tutorial**

其中：

**start** 将会告诉命令行创建一个新的app；

**MyIonicProject** 是创建的应用的根目录名称和app的名称；

**tutorial** 为你刚刚创建的项目提供一个模板，ionic还可以提供一下的几种模板：

tabs : a simple 3 tab layout

sidemenu: a layout with a swipable menu on the side

blank: a bare starter with a single page

super: starter project with over 14 ready to use page designs

tutorial: a guided starter project

注意：在应用创建的同时会将node的相关模块下载到项目中，同时会提示是否希望安装Cordova

### 2.1.2 运行

$ cd MyIonicProject/ ：切换到项目目录下

$ ionic serve ：serve命令将会打开常用的浏览器并且运行应用；

## 2.2 ionic应用的项目结构

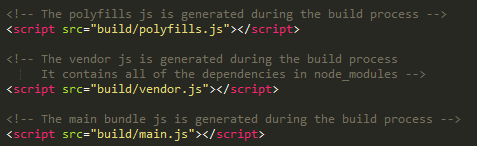
本章解刨一下创建的ionic应用，在我们创建的项目目录内，有一个典型的Cordova项目结构，在这里我们可以安装原生的插件，创建平台相关的项目文件；

### 2.2.1 ./src/index.html

./src/index.html 这个文件的应用主要的开始位置，他的目的是为应用设置脚本，样式等，不必花大量的时间研究；

在我们的应用中ionic将会在这个html文件中寻找<ion-app></ion-app>标签；

<ion-app></ion-app>标签下面：



这些脚本文件都是通过构建系统自动生成的，不用管；

### 2.2.2 ./src/

我们将会在这个目录中找到我们的代码。这里将是我们之后的工作中最常访问的地方。当我们运行ionic serve命令时，src/下的代码将会转变成当前浏览器版本兼容的javascript版本。

**1、src/app/app.module.ts**

这个文件是我们的应用的入口：

在这个文件的开始的部分我们会看到：

**@NgModule({**

**declarations: [**

**MyApp,**

**HomePage**

**],**

**imports: [**

**BrowserModule,**

**IonicModule.forRoot(MyApp)**

**],**

**bootstrap: [IonicApp],**

**entryComponents: [**

**MyApp,**

**HomePage**

**],**

**providers: [**

**StatusBar,**

**SplashScreen,**

**{provide: ErrorHandler, useClass: IonicErrorHandler}**

**]**

**})**

**export class AppModule {}**

每一个应用都会有一个root module来控制应用的其他部分。这也是我们使用**ionicBootstrap**加载应用的地方；在这个module中我们将MyApp设置成根元素（root component src/app/app.component.ts），这是我们的应用启动后加载的第一个组件，通常情况下它只是一个空壳子，用来将其他的组件加载进来。在**app.component.ts**文件中我们将**src/app/app.html**设置成模板：

**@Component({**

**templateUrl: 'app.html'**

**})**

**2、./src/app/app.html**

这个文件中是这个app的主要模板：

**<ion-nav id="nav" [root]="rootPage" #nav swipeBackEnabled="false"></ion-nav>**

**<ion-menu [content]="nav">**

**<ion-header>**

**<ion-toolbar>**

**<ion-title>Pages</ion-title>**

**</ion-toolbar>**

**</ion-header>**

**<ion-content>**

**<ion-list>**

**<button ion-item \*ngFor="let p of pages" (click)="openPage(p)">**

**{{p.title}}**

**</button>**

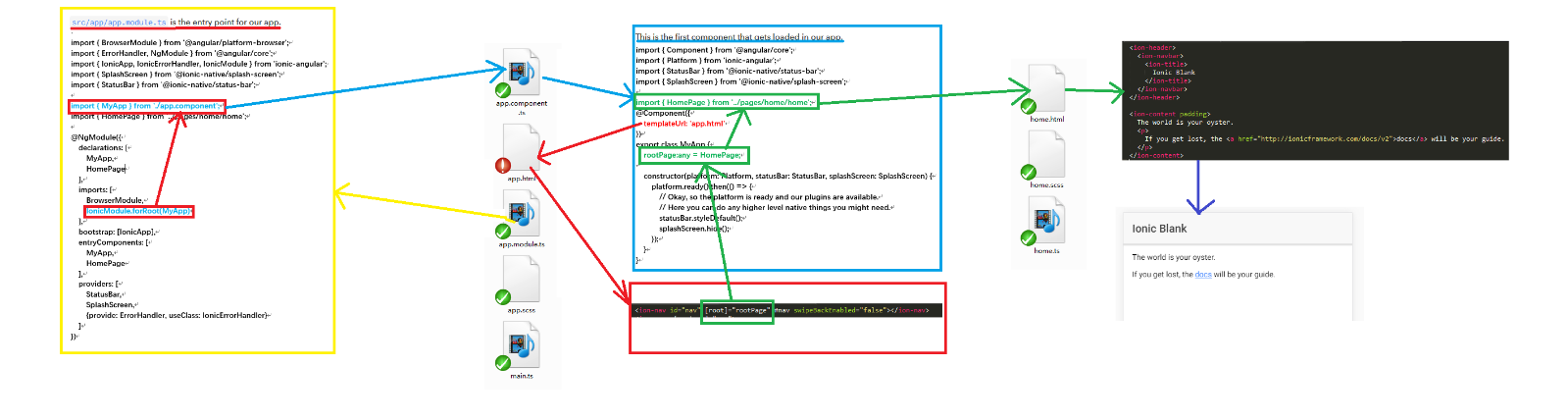
**</ion-list>**

**</ion-content>**

**</ion-menu>**

在这个模板中我们使用一个ion-menu构建一个侧边栏，然后使用ion-nav组件作为主内容区域；ion-menu的content属性绑定到nav变量（属于ion-nav），因此他知道自己的活动范围；

### 2.2.3 ionic项目结构小结



## 2.3 创建、导航页面

2.2中已经基本了解了ionic应用的布局。接下来在我们的应用中创建页面，并且配置他的导航；

注意观察**src/app/app.html**，在这个文件中有以下内容：

<**ion-nav** [root]="rootPage" #content swipeBackEnabled="false"><**/ion-nav**>

这里root属性的设置说明这个**ion-nav** 组件首要的或者root 页面。当**ion-nav** 加载的时候被rootPage 引用的组件将成为root页面。

例如：

在**src/app/app.component.ts** 文件中，MyApp在他的构造函数中指定了rootPage，如以下代码中蓝色字体部分，而绿色字体部分又将HomePage导航到了最终的页面所在的目录；

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

**import { Platform } from 'ionic-angular';**

**import { StatusBar } from '@ionic-native/status-bar';**

**import { SplashScreen } from '@ionic-native/splash-screen';**

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

**@Component({**

**templateUrl: 'app.html'**

**})**

**export class MyApp {**

**rootPage:any = HomePage;**

**constructor(platform: Platform, statusBar: StatusBar, splashScreen: SplashScreen) {**

**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();**

**});**

**}**

**}**

我们可以看到，rootPage设置成了HomePage，因此HonePage将会被加载到nav controller中；

### 2.3.1 创建一个page

接下来我们打开**\src\pages\home** 这个目录下的**home.ts** 文件看一下：

你可能会发现每一个page都会有自己的目录，并且以自己的文件名命名，例如HomePage自己的目录是：**\src\pages\home** 并且在这个目录下我们也会看到相同名称的.html 文件和 .scss文件。例如在**\src\pages\home** 目录下我们看到[home.html、home.scss](http://home.html、home.scss)和home.ts文件。尽管这样的组合并不是必须的，但是他对于保证应用的结构却很有用。

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

**import { NavController } from 'ionic-angular';**

**@Component({**

**selector: 'page-home',**

**templateUrl: 'home.html'**

**})**

**export class HomePage {**

**constructor(public navCtrl: NavController) {**

**}**

**}**

上边的代码中，HomgPage类创建了一个Angular组件，他包含了所有Ionic所支持的准则，并且通过Ionic的单行系统加载。需要注意的是由于页面被设定成动态加载，他们没有必要有自己的**selector** ，但是**selector** 在你需要为指定的页面设置特殊的样式的时候可以提供很大的帮助（**home.scss** 文件）。

所有的页面都需要一个class（HomePage）和一个相关的template（**templateUrl: 'home.html'**）我们查看这个html文件：

**<ion-header>**

**<ion-navbar>**

**<ion-title>**

**Ionic Blank**

**</ion-title>**

**</ion-navbar>**

**</ion-header>**

**<ion-content padding>**

**The world is your oyster.**

**<p>**

**If you get lost, the <a href="http://ionicframework.com/docs/v2">docs</a> will be your guide.**

**</p>**

**</ion-content>**

### 2.3.2 创建一个额外的page

要创建一个额外的页面，除了需要正确的配置标题和导航，我们不需要做其他的大量的工作。