目录

[ 移动端的3种主流的开发模式 1](#_Toc492630358)

[ 搭建Android的开发环境 2](#_Toc492630359)

[ Android开发的工具的使用 2](#_Toc492630360)

[ 第一个Android工程 创建和运行 3](#_Toc492630361)

[ 第一个HybridApp工程 3](#_Toc492630362)

[ 在Android开发时，查看js通过控制台所输出的日志信息 5](#_Toc492630363)

[ Ionic 5](#_Toc492630364)

[ webStorm指定视口的快捷键：*meta:vp +tab* 5](#_Toc492630365)

[1. Ionic概述 6](#_Toc492630366)

[2. 使用Ionic的基本页面结构 6](#_Toc492630367)

[3. 常见的颜色的使用 7](#_Toc492630368)

[4. Button按钮 7](#_Toc492630369)

[5. List 列表项 8](#_Toc492630370)

[6. card 卡片 8](#_Toc492630371)

[7. form 9](#_Toc492630372)

[8. tabs 选项卡(标签页) 11](#_Toc492630373)

[9. 九、grid 11](#_Toc492630374)

[ Ionic基于angularJS所封装的指令和服务 13](#_Toc492630375)

[10. ionHeaderBar中标题默认情况下，在Android设备下是靠左居中，在iOS设备默认是居中，需要统一设置指定align-title 13](#_Toc492630376)

[11. ionRefresher 下拉刷新 13](#_Toc492630377)

[12. ionInfiniteScroll 上滑无限加载更多 14](#_Toc492630378)

[13. $ionicScrollDelegate 控制滚动位置 15](#_Toc492630379)

[14. SideMenus 侧边栏菜单 16](#_Toc492630380)

[15. ionTabs 标签页 18](#_Toc492630381)

[16. 窗口$ionicLoading 19](#_Toc492630382)

[17. 自定义弹窗$ionicModal 21](#_Toc492630383)

[18. 序列化对象为地址栏get参数(uname=yy&upwd=123) 22](#_Toc492630384)

## 移动端的3种主流的开发模式

1、WebApp： 网页应用程序，运行在手机浏览器中的，提供的用户体验非常接近原生app的体验的应用程序

（网页版的淘宝、京东）

2、NativeApp：原生应用程序：运行在手机的操作系统的，借助SDK所提供的原生组件或者服务进行编程而生成的app称之为NativeApp

SDK：Software Development Kit 软件开发包

Android（Google提供的SDK+java（kotlin））

iOS（Apple提供的SDK+oc(swift)）

WP（Microsoft提供的SDK..）

（可以下载安装的微信、淘宝）

3、HybridApp：混合编程：借助于前端开发技能和原生开发技巧，进行混合开发而生成的可以安装在手机上的app称之为HybridApp

（微信、美团。。）



## 搭建Android的开发环境

http://www.androiddevtools.cn/

①安装jdk

（Java Development Kit java的运行环境）

②安装adt

（Android Developer tools 安卓开发工具）

## Android开发的工具的使用

Eclipse：一款IDE（集成开发环境 intergrated Develop enverionment）,主要是以java语言为主的开发（由于Android主要开发语言就是java）

工具条的显示或者隐藏：window->show/hide toolbar

创建模拟器（虚拟设备）：

点击工具条中 Android Virtual Device Manager 对应的按钮，在弹窗中点击new新建一个虚拟设备，

①指定设备的名称为myPhone，

②选中设备：4英寸 分辨率480\*800;

③将backCamera设置为emulated

④sd卡指定512M

如何显示logCat？

window->show view->other->搜索logcat 选中第一个

## 第一个Android工程 创建和运行

**1、创建：**

①file->new->Android Application Project

②配置application name 以及 package name （com.tarena.demo01）

③configure launcher icon 启动图标

④一路next，最后finish

**2、运行：**

选中要运行的项目名字，点击右键，run as Android Application.

**3、Android工程基本运行方式：**

在layout目录下的activity\_main.xml文件，完成app视图的配置。

通过java来编写业务逻辑代码的。（src进行java代码的编辑）

**4、一个Android工程的基本目录结构**

src ：存储的是编写业务逻辑的java代码

gen:存储的是整个Android工程自动生成的文件

assets:默认是空的，存储的hybrid开发时对应的前端的代码

bin:(binary) 存储是生成的可以安装在手机中的apk结尾的文件

libs：（libraries） 存储的是当前工程所依赖的库文件

res:(resource 资源) 存储的是图片、布局文件、值的配置文件

AndroidManifest.xml:这个配置文件非常重要，记录了当前Android程序的版本信息、权限的设置、启动哪个activity。。。。

**5、如何给AndroidApp添加某种权限？？**

①点击当前工程的AndroidManifest.xml

②点击permissions标签页

③点击add，在弹窗中选中最后一个uses permission

④从右侧的下拉菜单中找到需要用到的权限名字

⑤ctr+s 保存就完成权限的设置了。

## 第一个HybridApp工程

**方式1：将前端代码全部拷贝到assets**

混合是前端代码和原生的代码。

基本步骤：

①创建前端项目，实现功能，并测试保证功能是ok

②创建一个Android工程

③将前端项目中代码拷贝到Android工程的assets目录

④通过一个原生组件WebView载入指定的前端代码

//创建一个WebView类型的对象

WebView wv = new WebView(getApplicationContenxt());

//载入指定的url

wv

.loadUrl("file:///android\_asset/test.html");

//将配置过的wv作为内容视图去展示

setContentView(wv);

**知识点：**

webView默认为了保证系统的安全，是不允许执行的js。

解决方案：

//设置允许执行js

**wv.getSettings().setJavaScriptEnabled(true);**

**方案2: 将前端代码部署在服务器**

①完成前端工程的所有的功能

②创建一个Android工程

③将前端代码部署在服务器端，拿到服务器返回的地址

localhost-->真实的ip地址

（windows+R -> cmd ->ipconfig）

④编写java，通过webView载入指定的url

⑤添加网络权限，才可以实现与服务器端的交互

任务：将开饭了项目 打包生成一个hybridApp，在Android手机的操作系统中去运行。

①前端项目已经做好

②新建一个Android工程

③将前端项目中的代码 部署在服务器端

**http://localhost/chaptor4/ng/kfl/kaifanla.html#/kflStart**

**--》**

**http://172.163.100.93/chaptor4/ng/kfl/kaifanla.html#/kflStart**

④编写java代码

⑤添加能够访问网络的权限

在Android4.4系统版本之上，允许使用css3和es6的新特性，在之下是不允许。

## 在Android开发时，查看js通过控制台所输出的日志信息

**//可以查看控制台输出的日志信息**

wv.setWebChromeClient(new WebChromeClient(){

//onCon

@Override

public boolean onConsoleMessage(ConsoleMessage consoleMessage) {

// TODO Auto-generated method stub

return super.onConsoleMessage(consoleMessage);

}

});

**例子：**

**public** **class** MainActivity **extends** Activity {

@Override

**protected** **void** onCreate(Bundle savedInstanceState) {

**super**.onCreate(savedInstanceState);

setContentView(R.layout.*activity\_main*);

//1、创建Web view组件实例 alt + / ①帮助自动补全；②

WebView wv = **new** WebView(getApplicationContext());

//2、载入指定的页面(assets)

wv.loadUrl("file:///android\_asset/test.html");

//wv.loadUrl("http://176.116.55.229/ng/kfl\_project/kaifanla.h//tml#/kflStart");

//3、允许使用js脚本

wv.getSettings().setJavaScriptEnabled(**true**);

//4、将位置指定页面的wv设置为内容视图

setContentView(wv);

}

## Ionic

### webStorm指定视口的快捷键：*meta:vp +tab*

*<!--meta:vp-->*<**meta name="viewport"  
 content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0"**/>

## Ionic概述

已经掌握了实现混合编程的基本技能，但是所实现的hybridapp ①与原生应用交互体验差太大 ②功能比较少（无法实现与设备底层的交互）

？？ ionic做一个移动端的框架，可以让界面美观，让功能更强大。

http://ionicframework.com/ 官网

http://ionicons.com/ 图标的官网

what？Ionic是一个移动端的开源的ui类框架，

ionic = ng(数据处理)+cordova(phonegap提供了js接口允许调用设备底层)+ui库(样式类、基于ng封装的指令和服务)+uiRouter(类似ngRoute的路由模块)

GUI 图形化

CLI 命令行

why？

Ionic完美的融合了ng、cordova（学习成本低）

Ionic提供了非常的漂亮的ui库（包括字体图标、丰富的样式类）

Ionic提供了基于CLI(command line interface)的操作方式

Ionic的性能比较好

where？

移动端的app（WebApp，配合WebView实现hybridApp）

how？

两种方式：

方式1

直接引入对应的js css fonts文件

方式2

安装nodejs

安装corodva和ionic

npm install -g cordova ionic

创建项目 ionic start myApp tabs

启动项目 cd myApp ;ionic serve

## 使用Ionic的基本页面结构

**页头**

<ion-header-bar> </ion-header-bar>

**正文**

<ion-content> </ion-content>

**页尾**

<ion-footer-bar> </ion-footer-bar>

前提：创建的自定义模块，必须指定依赖ionic这个模块，这样ionic所提供的指令或者服务，才可以在自定义模块的范围内去调用。

## 常见的颜色的使用

哪些？

light 浅灰色

stable 灰色

positive 天蓝色

calm 湖蓝色

balanced 绿色

enerygized 黄色

assertive 红色

royal 紫色

dark 黑色

用在哪些地方？

前景色、背景色、边框

## Button按钮

button 调整按钮的大小，加上了圆角效果

button-positive 天蓝色....

button-block 宽度

button-large/button-small 大小

button-outline 边框

button-clear 只显示文本

icon-left 图标在按钮文本的左边

ion-home 具体的图标

button-bar 将容器中的按钮进行宽度的均等分

<**ion-content**>  
 <**button class="button button-large button-positive"**>  
 大按钮  
 </**button**>  
 <**button class="button button-small button-assertive"**>  
 小按钮  
 </**button**>  
 <**br**/>  
 <**div class="button-bar"**>  
 <**button class="button icon-left ion-person"**>btn1</**button**>  
 <**button class="button icon-right ion-gear-a"**>btn2</**button**>  
 <**button class="button button-clear button-dark"**>btn3</**button**>  
 </**div**>  
</**ion-content**>

## List 列表项

list/item 让指定的容器和列表项目具有列表的外观显示效果

item-divider 用在item中，实现一个比较醒目的分隔效果，字体加粗，背景色变成灰色

list-inset 用在list中，实现一个嵌入的效果（距离上下左右都会margin）

item-icon-left/right 用在item中，指定一个图标显示在列表项的左边或者右边或者两边都显示

badge 计数气泡(徽章)，可以用在item中，通过一个span指定class='badge badge-assertive'

item-button-left/right,用在item中，给列表项指定一个可以点击的按钮

在ionic提供的样式类中，列表的实现有3种方式：

①图标列表 (图标比较小)

<li class='item item-icon-left'></li>

②联系人列表(图片要比图标大 比缩略图小)

<li class='item item-avatar'></li>

③缩略图列表

<li class='item item-thumbnail-left'></li>

<**ul class="list list-inset"**>  
 <**li class="item item-icon-left item-icon-right"**>  
 <**i class="icon ion-wifi"**></**i**>  
 Wi-Fi  
 <**span class="badge "**>未连接</**span**>  
 <**i class="icon ion-ios-arrow-right"**></**i**>  
 </**li**>  
 <**li class="item item-icon-left item-icon-right"**>  
 <**i class="icon ion-gear-a"**></**i**>  
 系统更新

**//徽章** <**span class="badge badge-assertive"**>  
 有新版本  
 </**span**>  
 <**i class="icon ion-ios-arrow-right"**></**i**>  
 </**li**>  
</**ul**>

## card 卡片

提供的样式类 主要是用来进行数据的展示

<div class="card">

<div class="item item-divider">

header

</div>

<div class='item'>

data...

</div>

<div class="item item-divider">

footer

</div>

</div>

## form

表单中放置label的四种方式：

①placeholder labels

<label class="item item-input">

<input type='text' placeholder="输入用户名'/>

</label>

②inline labels

<label class="item item-input">

<span class='input-label'>请输入用户名</span>

<input type='text' placeholder="输入用户名'/>

</label>

③stacked labels

<label class="item item-input item-stacked-label">

<span class='input-label'>请输入用户名</span>

<input type='text' placeholder="输入用户名'/>

</label>

④floating labels

<label class="item item-input item-floating-label">

<span class='input-label'>请输入用户名</span>

<input type='text' placeholder="输入用户名'/>

</label>

* item-select
* item-toggle
* ionList ionToggle ionCheckbox ionRadio

<**div class="list list-inset"**>  
 *<!-- 姓名-->* <**label class="item item-input"**>  
 <**span class="input-label"**>姓名</**span**>  
 <**input type="text" placeholder="请输入用户名"**/>  
 </**label**>  
 *<!-- 搜索-->* <**label class="item item-input"**>  
 <**span class="input-label"**>要查找的内容</**span**>  
 <**i class="icon ion-search placeholder-icon"**></**i**>  
 <**input type="text"  
 placeholder="请输入要查找的内容"**/>  
 </**label**>  
 *<!-- 日期 -->* <**label class="item item-input"**>  
 <**span class="input-label"**>日期</**span**>  
 <**input type="date"**/>  
 </**label**>  
 *<!-- 颜色 select-->* <**label class="item item-select"**>  
 <**span class="input-label"**>选择喜欢的颜色</**span**>  
 <**select** >  
 <**option value="red"**>红色</**option**>  
 <**option value="green"**>绿色</**option**>  
 <**option value="blue"**>蓝色</**option**>  
 </**select**>  
 </**label**>  
 <**div class="item item-toggle"**>  
 HTML5  
 <**label class="toggle toggle-assertive"**>  
 <**input type="checkbox"**>  
 <**div class="track"**>  
 <**div class="handle"**></**div**>  
 </**div**>  
 </**label**>  
 </**div**>  
  
 <**ion-toggle**> 开关</**ion-toggle**>  
  
 <**div class="item"**>  
 选择喜欢的电影  
 *<!-- 复选框-->* <**ion-list**>  
 <**ion-checkbox** >movie1</**ion-checkbox**>  
 <**ion-checkbox** >movie2</**ion-checkbox**>  
 <**ion-checkbox** >movie3</**ion-checkbox**>  
 </**ion-list**>  
 </**div**>  
  
 <**div class="item"**>  
 选择性别  
 *<!-- 单选-->* <**ion-list**>  
 <**ion-radio value="1"**>男</**ion-radio**>  
 <**ion-radio value="2"**>女</**ion-radio**>  
 </**ion-list**>  
  
 </**div**>  
  
</**div**>

## tabs 选项卡(标签页)

tabs/tabs-icon-left/tabs-icon-top.../

tab-item

<**div class="tabs tabs-icon-left"**>  
 <**a class="tab-item"**>  
 <**i class="icon ion-home"**></**i**>  
 主页  
 </**a**>  
 <**a class="tab-item"**>  
 <**i class="icon ion-person"**></**i**>  
 个人中心  
 </**a**>  
 <**a class="tab-item"**>  
 <**i class="icon ion-gear-a"**></**i**>  
 设置  
 </**a**>  
</**div**>

## 九、grid

row/col 指定是一个行或者一个列

指定具体的列的宽度：

.col-10 10%

.col-20 20%

.col-25 25%

.col-33 33.3333%

.col-50 50%

.col-67 66.6666%

.col-75 75%

.col-80 80%

.col-90 90%

设置距离左边的这一列的偏移量

.col-offset-10 10%

.col-offset-20 20%

.col-offset-25 25%

.col-offset-33 33.3333%

.col-offset-50 50%

.col-offset-67 66.6666%

.col-offset-75 75%

.col-offset-80 80%

.col-offset-90 90%

列之间垂直对齐的方式：

默认col-top

col-center col-bottom

通过行进行列对齐的统一的设置：

row-top/row-center/row-bottom

<**div class="row"**>  
 <**div class="col col-bottom col-10"**>  
 <**button class="button button-block button-positive"**>  
 第一列  
 </**button**>  
 </**div**>  
 <**div class="col col-center col-offset-20"**>  
 <**button class="button  
 button-block button-assertive"**>  
 第二列  
 </**button**>  
 </**div**>  
 <**div class="col col-10"**>  
 <**button class="button button-block button-positive"**>  
 第三列  
 </**button**>  
 <**br**/>  
 <**button class="button button-block button-positive"**>  
 第三列  
 </**button**>  
 <**br**/>  
 <**button class="button button-block button-positive"**>  
 第三列  
 </**button**>  
 </**div**>  
</**div**>

## Ionic基于angularJS所封装的指令和服务

## ionHeaderBar中标题默认情况下，在Android设备下是靠左居中，在iOS设备默认是居中，需要统一设置指定align-title

例子：

<ion-header-bar align-title='center'>

<h1 class='title'>myTitle</h1>

</ion-header-bar>

## ionRefresher 下拉刷新

是ionic基于ng封装的自定义指令，可以实现pull-to-refresh(下拉刷新)的效果

**使用步骤：**

1. 将ionRefresher指令用在ionContent的第一个子元素位置

<**ion-content**>  
 *<!--①将ionRefresher放在ionContent的第一个子元素位置-->* <**ion-refresher on-refresh="loadData()"**></**ion-refresher**>

<**ul class="list"**><**li class="item"**></**li**></**ul**>  
</**ion-content**>

②通过on-refresh去指定下拉刷新时 要执行的操作

③结束掉刷新动作

$scope.$broadcast('scroll.refreshComplete');

**var *app*** = ***angular***.module(**'myApp'**, [**'ionic'**]);  
***app***.**controller**(**'myCtrl'**,  
 [**'$scope'**,**'$timeout'**, **function** ($scope,$timeout) {  
 *//当下拉刷新执行该方法* $scope.loadData = **function** () {  
 ***console***.log(**'下拉刷新执行方法'**);  
 *//模拟2s的延迟：在2s之后，  
 //通过广播事件来结束刷新动作* $timeout(**function** () {  
 *//广播* $scope.$broadcast(**'scroll.refreshComplete'**);  
 }, 2000)  
 }  
 }]  
);

## ionInfiniteScroll 上滑无限加载更多

停用无限加载更多:ngIf

阻止首次加载时检查是否还能加载更多：immediate-check=false;

使用步骤：

①将ionInfiniteScroll放到ionContent的最后一个子元素

②on-infinite去绑定加载更多时要执行的方法

1. 通过广播结束加载更多的操作

$scope.$broadcast('scroll.infiniteScrollComplete');

**Html:**

*<!-- 无限滚动的自定义指令-->  
 <!-- 去掉加载更多的功能，只需要指定ngIf指令-->* <**ion-infinite-scroll  
 on-infinite="loadMore()"**>  
 </**ion-infinite-scroll**>  
</**ion-content**>

**Js:**

**var *app*** = ***angular***.module(**'myApp'**, [**'ionic'**]);  
***app***.**controller**(**'myCtrl'**,  
 [**'$scope'**, **function** ($scope) {  
 $scope.**myList** = [];  
 *//给myList初始化一些数据* **for**(**var** i=0;i<50;i++) {  
 $scope.**myList**.push(  
 Math.floor(Math.random()\*100)  
 );  
 }  
 *//定义一个加载更多的方法* $scope.loadMore = **function** () {  
 ***console***.log(**'准备加载更多数据'**);  
 *//当数据处理完毕，需要通过广播指定的事件  
 // 结束加载更多的动作* $scope.$broadcast(**'scroll.infiniteScrollComplete'**);  
 }  
 }]  
);

## $ionicScrollDelegate 控制滚动位置

$ionicScrollDelegate是由ionic基于ng自定义的一个服务，控制滚动相关的操作

$ionicScrollDelegate.scrollTop() 滚动到顶部

$ionicScrollDelegate.scrollBottom() 滚动到底部

$ionicScrollDelegate.scrollTo()滚动到指定的位置

$ionicScrollDelegate.getScrollPosition()得到当前的滚动位置

<**body ng-controller="myCtrl"**>  
<**ion-header-bar**>  
 <**h1 class="title"**>页头</**h1**>  
</**ion-header-bar**>  
<**ion-content**>  
 <**ul class="list list-inset"**>  
 <**li class="item" ng-repeat="tmp in myList"**>  
 {{**tmp**}}  
 </**li**>  
 </**ul**>  
</**ion-content**>  
<**ion-footer-bar**>  
 <**button ng-click="goToTop()" class="button button-clear button-dark"**>  
 top  
 </**button**>  
 <**button ng-click="savePosition()" class="button button-positive"**>  
 save  
 </**button**>  
 <**button ng-click="restorePosition()" class="button button-balanced"**>  
 restore  
 </**button**>  
 <**h1 class="title"**>页尾</**h1**>  
</**ion-footer-bar**>  
<**script**>  
 **var *app*** = ***angular***.module(**'myApp'**, [**'ionic'**]);  
 *//使用服务的基本步骤？①注入到对应的ng对象 ②使用服务所提供的方法* ***app***.**controller**(**'myCtrl'**,  
 [**'$scope'**, **'$ionicScrollDelegate'**,  
 **function** ($scope, $ionicScrollDelegate) {  
 $scope.**myList** = [];  
 **for**(**var** i=0;i<100;i++) {  
 $scope.**myList**.push(i);  
 }  
 $scope.goToTop = **function** () {  
 *//通过ionic封装好的自定义服务来讲列表滚动到顶部  
 //scrollTop方法有一个可选的参数，在滚动到顶部时是否要有动画效果* $ionicScrollDelegate.scrollTop(**true**);  
 };  
 *//保存当前的位置* $scope.savePosition = **function** () {  
 **var** position = $ionicScrollDelegate.getScrollPosition();  
 ***console***.log(position);  
 $scope.**topValue** = position.**top**;  
 };  
 *//恢复之前保存的位置* $scope.restorePosition = **function** () {  
 $ionicScrollDelegate  
 .scrollTo(0,$scope.**topValue**,**true**)  
 }  
 }  
 ]  
 )  
</**script**>  
</**body**>

## SideMenus 侧边栏菜单

**①使用侧边栏菜单的基本结构：**

<ion-side-menus>

<ion-side-menu-content>

</ion-side-menu-content>

<ion-side-menu side='left/right' width='200'>

</ion-side-menu>

</ion-side-menus>

**②操作(开、关)侧边栏菜单的方式**

打开侧边栏菜单：

menu-toggle

例如：

<button menu-toggle='left'>

toggle

</button>

关闭侧边栏菜单：

menu-close

例如：

<a menu-close></a>

js的方式（借助于服务 $ionicSideMenuDelegate）

$ionicSideMenuDelegate.toggleLeft();

$ionicSideMenuDelegate.toggleRight();

<**body ng-controller="myCtrl"**>  
<**ion-side-menus**>  
 <**ion-side-menu-content**>  
 *<!-- 显示的页头、页尾、正文-->* <**ion-header-bar**>  
 <**h1 class="title"**>页头</**h1**>  
 </**ion-header-bar**>  
 <**ion-content**>  
 <**button menu-toggle="left"  
 class="button button-balanced"**>  
 操作左侧侧边栏菜单  
 </**button**>  
 *<!-- 通过js的方式来完成侧边栏菜单的打开和关闭操作-->* <**button  
 ng-click="openRightMenu()"  
 class="button button-positive"**>  
 打开右边的侧边栏菜单  
 </**button**>  
 </**ion-content**>  
 <**ion-footer-bar**>  
 <**h1 class="title"**>页尾</**h1**>  
 </**ion-footer-bar**>  
 </**ion-side-menu-content**>  
  
 <**ion-side-menu side="right"**>  
 *<!-- 显示的侧边栏菜单-->* <**ul class="list list-inset"**>  
 <**li class="item"**>  
 个人中心  
 </**li**>  
 <**li class="item"**>  
 设置  
 </**li**>  
 <**li class="item"**>  
 关于  
 </**li**>  
 </**ul**>  
 </**ion-side-menu**>  
  
 <**ion-side-menu**>  
 <**a menu-close**>关闭菜单</**a**>  
 <**ul class="list list-inset"**>  
 <**li class="item item-icon-left"**>  
 <**i class="icon ion-home"**></**i**>  
 主页  
 </**li**>  
 <**li class="item item-icon-left"**>  
 <**i class="icon ion-person"**></**i**>  
 个人订单  
 </**li**>  
 </**ul**>  
 </**ion-side-menu**>  
</**ion-side-menus**>  
<**script**>  
 **var *app*** = ***angular***.module(**'myApp'**, [**'ionic'**]);  
 ***app***.**controller**(**'myCtrl'**,  
 [**'$scope'**, **'$ionicSideMenuDelegate'**,  
 **function** ($scope, $ionicSideMenuDelegate) {  
 *//打开右边的侧边栏菜单* $scope.openRightMenu = **function** () {  
 $ionicSideMenuDelegate.toggleRight(**true**);  
 }  
 }  
 ])  
</**script**>

## ionTabs 标签页

基于ng封装的自定义指令，主要是来实现标签页（选项卡）:页面导航

5.1 基本结构

<ion-tabs>

<ion-tab> </ion-tab>

<ion-tab> </ion-tab>

<ion-tab> </ion-tab>

</ion-tabs>

5.2 ion-tab

<ion-tab icon-on="" icon-off="" on-select=""// 当选中某一个标签执行的方法，默认情况下当页面首次加载时会执行第一个标签选中时所指定的方法 title="myTitle">

</ion-tab>

<**body ng-controller="myCtrl"**>  
<**ion-header-bar**>  
 <**h1 class="title"**>header</**h1**>  
</**ion-header-bar**>  
<**ion-tabs class="tabs-icon-top"**>  
 <**ion-tab  
 on-select="handleSelect()"  
 title="主页"  
 icon-off="ion-ios-home-outline"  
 icon-on="ion-ios-home"**>  
 </**ion-tab**>  
 <**ion-tab ng-click="handleClick()"  
 title="个人中心"  
 icon="ion-person"**>  
 </**ion-tab**>  
 <**ion-tab on-select=""  
 title="设置"  
 icon="ion-gear-a"**>  
  
 </**ion-tab**>  
</**ion-tabs**>  
<**script**>  
 **var *app*** = ***angular***.module(**'myApp'**, [**'ionic'**]);  
 ***app***.**controller**(**'myCtrl'**,  
 [**'$scope'**, **function** ($scope) {  
 *//定义当选中tab时要执行的方法* $scope.handleSelect = **function** () {  
 ***console***.log(**'tab被选中了'**);  
 }  
 *//定义当点击第二个tab要执行的方法* $scope.handleClick = **function** () {  
 ***console***.log(**'第二个标签被选中了'**);  
 }  
 }]  
 )  
</**script**>

## 刷新窗口$ionicLoading

**6.1 加载中窗口**

$ionicLoading.show({template:'',duration:2000})

$ionicLoading.hide()

**var *app*** = ***angular***.module(**'myApp'**, [**'ionic'**]);  
***app***.**controller**(**'myCtrl'**,  
 [**'$scope'**, **'$ionicLoading'**,  
 **function** ($scope, $ionicLoading) {  
 *//定义一个点击处理函数，  
 // 在处理函数中显示一个‘正在刷新’的窗口* $scope.showLoading = **function** () {  
 $ionicLoading.**show**({  
 **template**:**'正在刷新...'**,  
 **duration**:2000  
 });  
 }  
 }  
 ])

**6.2 actionSheet**

//show方法的返回值是一个方法，调用返回的方法可以actionSheet关掉。

$ionicActionSheet.show({

buttons:[{text:''}],

buttonClicked:function(index){},

cancelText:'',

destructiveText:'',

destructiveButtonClicked:function(){}

})

6.3 $ionicPopup

$ionicPopup.alert({title:'',template:''})//警告窗口

$ionicPopup.confirm()//确认窗口

$ionicPopup.prompt()// 输入提示框

<**ion-content**>  
 <**p**>正文</**p**>  
 <**button  
 ng-click="handleClick()"  
 class="button button-assertive"**>  
 edit  
 </**button**>  
</**ion-content**>

<**script**>  
 **var *app*** = ***angular***.module(**'myApp'**, [**'ionic'**]);  
 ***app***.**controller**(**'myCtrl'**,  
 [**'$scope'**, **'$ionicActionSheet'**, **'$timeout'**,  
 **function** ($scope, $ionicActionSheet, $timeout) {  
 $scope.handleClick = **function** () {  
 *//显示一个actionSheet* **var** hideActionSheet =  
 $ionicActionSheet.show(  
 {  
 **titleText**: **'编辑'**,  
 **cancelText**: **'取消编辑'**,  
 **buttons**: [  
 {**text**: **'分享到朋友圈'**},  
 {**text**: **'分享到新浪微博'**}  
 ],  
 buttonClicked: **function** (index) {  
 ***console***.log(  
 **"第"** + index + **'个自定义操作的按钮被点击了'**);  
 *//在按钮点击之后，将actionSheet给隐藏掉* **return true**;  
 },  
 **destructiveText**: **'删除'**,  
 destructiveButtonClicked: **function** () {  
 ***console***.log(**'删除按钮被点击了'**);  
 **return true**;  
 }  
 }  
 );  
 ***console***.log(hideActionSheet);  
 *//2s之内没有任何操作就将actionSheet隐藏起来* $timeout(**function** () {  
 hideActionSheet();  
 }, 2000)  
 }  
 }]  
 )  
</**script**>

## 自定义弹窗$ionicModal

$ionicModal

步骤：

①创建

$ionicModal

.fromTemplateUrl('tpl/myModal.html')

.then(function(modal){

$scope.customModal = modal;

})

②调用

$scope.customModal.show()/hide()

高级用法：

在自定义弹窗中去调用所在页面中控制器中数据和方法，解决方案：

$ionicModal

.fromTempalteUrl('url',{scope:$scope})

## 序列化对象为地址栏get参数(uname=yy&upwd=123)

Var obj={addr:’dd’,did:3,phone:123,uname:’yys’}

**$HttpParamSerializerJQlike(); //addr=dd&did=3&phone=123&sex=1&uname=yys**