# 混合app开发课程介绍

## 什么是混合app

application（应用）

用web前端的知识开发出来app的方式就叫做混合app开发

## 为什么要学习这部分课程那？

“钱”多

有移动开发经验的开发人员比普通的web前端开发人员工资要高出一大截。

## 混合开发课程的目标是什么？

我们用学过的前端知识可以开发出一个可以安装在手机里面的app。

顺便复习前边的基础

## 混合app开发部分课程要讲什么东西？

# 混合开发简介

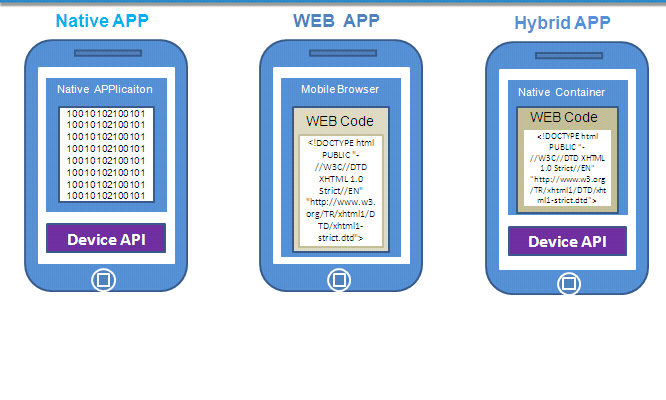
## 现在app开发方类型有几种？

WebApp:就是在浏览器中运行的web应用

NativeApp:用android和Object-C等原生语言开发的应用

HybridApp:就是外面是原生的壳，里面是webapp应用，兼具2者的优势

## 三种开发类型的原理和对比





## 企业如何选择用那种方式开发产品那？

企业要根据自身的特点进行app开发方式的选择，不要为了混合而混合。

混合app的开发方式是未来的主流趋势。

# 流行开发框架介绍

## 移动端开发有哪些框架那？

**混合App：**

Ionic

Html5+ Mui

AppCan

Framework7

移动APP：

Jquery Mobile

原生APP：

React Native

…….

## Html5+

### 首先要认识Html5+联盟是什么？

HTML5中国产业联盟，简称“HTML5+联盟”，是为了更好的推进HTML5的商用、更好的为HTML5开发者服务而由产 业链主流厂商共同组成的一个联盟。

联盟致力于整合产业链资源，建立围绕HTML5开发者的生态系统，通过产业链共同为HTMLl5开发者服务。

W3C中国是联盟的指导单位。

CSDN 和 DCloud 是联盟的秘书处单位。

### Dcloud官网

官网：<http://www.dcloud.io/>

Emmet语法:<http://docs.emmet.io/cheat-sheet/>

这个公司下面有四个主要的产品



HBuilder：是一款web开发工具。它能大幅提升开发效率，对程序员也设计了更人文关怀的UI，它包括最全面的语法库和浏览器兼容性数据。到现在 已经有两年的时间了，到现在大概有几十万的开发者在使用。

5+Runtime：是一个H5的增强引擎。这是一个可以去调用H5的能力和原生操作系统的产 品。

MUI框架：是高性能的前端框架，可用开发高性能App，也是目前最接近原生App效果的框架，可以有效解决HTML5原生开发中遇到的部分问题，同 时体积也小。

应用流：会让用户感觉一点开就可以用，这也是颠覆现有的应用方式的主要因素，给最终用户去使用应用有更好的体验。

### Html5+的优点

提供了一个自定制的IDE工具，支持非常好，轻量级，甚至不需要安装直接解压缩就可以使用

开发出来的app的性能很好，用户体验很高

丰富的api支持，号称有40万+

### 现在那些公司在用Html5+?

<http://www.dcloud.io/case/#group-1>



## Ionic介绍

### 官网地址

[**Ionic官网**](http://ionicframework.com/) http://ionicframework.com/

[**Ionic中文网**](http://www.ionic.wang/) http://www.ionic.wang/

### 首先要认识Ionic框架？

IONIC 是目前最有潜力的一款 HTML5 手机应用开发框架。通过 SASS 构建应用程序，它提供了很多 UI 组件来帮助开发者开发强大的应用。 它使用 JavaScript MVVM 框架和 AngularJS 来增强应用。提供数据的双向绑定，使用它成为 Web 和移动开发者的共同选择。Ionic是一个专注于用WEB开发技术，基于HTML5创建类似于手机平台原生应用的一个开发框架。Ionic框架的目的是从 web的角度开发手机应用，基于PhoneGap的编译平台，可以实现编译成各个平台的应用程序。

### Ionic框架构成

**CSS框架** - 提供原生 App 质感的 CSS 样式模拟。 ionic 这部分的实现使用了 ionicons 图标样式库。

**JavaScript框架** - 提供移动 Web 应用开发框架。ionic 基于 AngularJS 基础框架开发，遵循 AngularJS 的框架约束；此外 ionic 使用 AngularJS UI Router 实现前端路由。

**命令行/CLI** - 命令行工具集用来简化应用的开发、构造和仿真运行。ionic 命令行工具使用了 Cordova，依赖于平台 SDK（Android & iOS）实现将移动 Web 项目打包成原生 App。

### Ionic特点

基于Angular语法，简单易学。

是一个轻量级框架。性能优越，运行速度快。

完美的融合下一代移动框架，支持 Angularjs 的特性， MVC ，代码易维护。

提供了漂亮的设计，通过 SASS 构建应用程序，它提供了很多 UI 组件来帮助开发者开发强大的应用。

专注原生，让你看不出混合应用和原生的区别

提供了强大的命令行工具。

### Ionic开发注意点有哪些？

由于 ionic 使用了 HTML5 和 CSS3 的一些新规范，所以要求 iOS7+ / Android4.1+。

在低于这些版本的手机上使用 ionic 开发的应用，有时会发生莫名其妙的问题。

### 现在那些公司再用Ionic开发？

http://www.phonegap100.com/app.html



## 

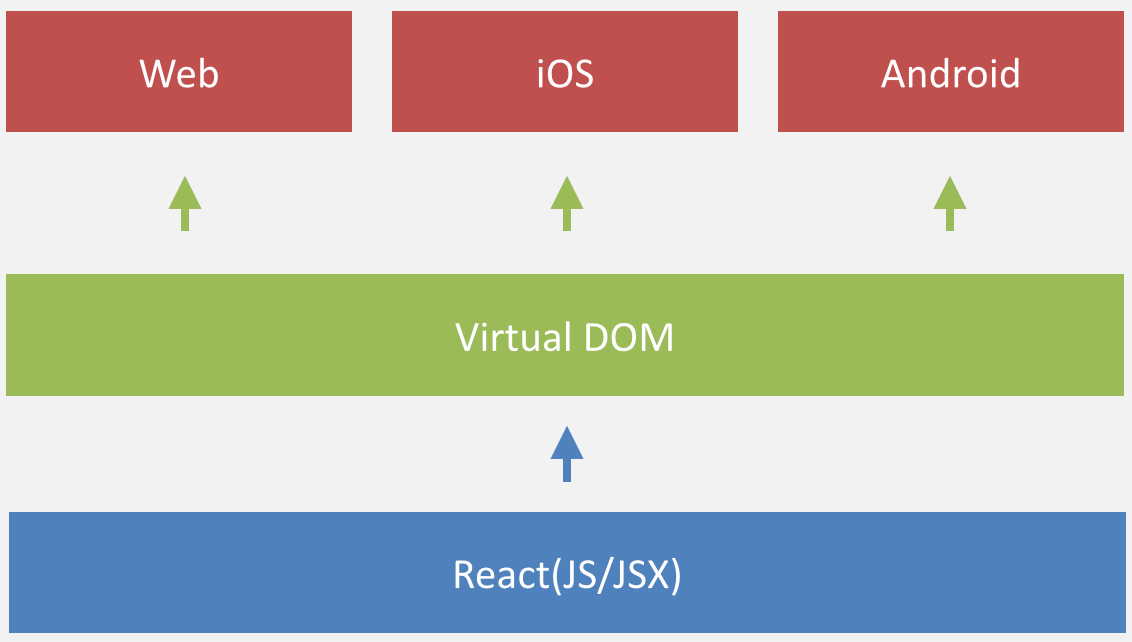
## React Native介绍

### 官网地址

[RN官网](https://facebook.github.io/react-native/) <https://facebook.github.io/react-native/>

[RN中文网](http://react-native.cn/) <http://react-native.cn/>

### React native原理图



### React Native核心实现是什么？

React Native里面**没有webview**，这货不是Hybrid app，里面执行JS是用的 JavascriptCore。

把Native code十来个最**基本核心的类**（RCTDeviceEventEmitter、RCTRenderingPerf等）、或组件（RCTView、RCTTextField、RCTTextView、RCTModalFullscreenView等）封装成二十来个**基本组件**（Popover、Listview等）

实现了一套类似css的子集，用来解决样式问题，相当复杂和强大，靠这个才能将Native的核心组件组成JS层的基本组件再组成业务端的业务组件

在React Native中，写JS的工程师解决的是「将基本组件拼装成可用的React组件」的问题，写Native Code的工程师解决的是「提供核心组件，提供足够的扩展性、灵活性和性能」的问题。

### React Native的优点有哪些？

不用Webview，彻底摆脱了Webview让人不爽的交互和性能问题。

有较强的扩展性，这是因为Native端提供的是基本控件，JS可以自由组合使用。

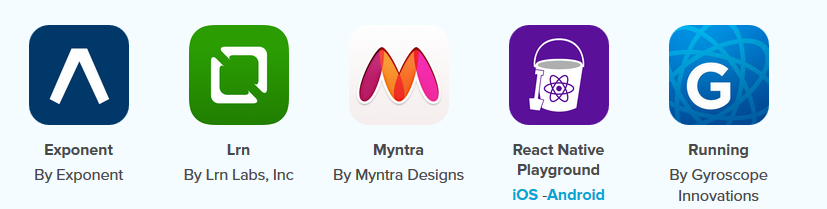
可以直接使用Native原生的「牛逼」动画。

可以通过更新远端JS，直接更新app。

### 现在那些公司在用React Native

<https://facebook.github.io/react-native/showcase.html>





淘宝部分功能也在用这个东西

## Jquery Mobile介绍

### 官网地址

[官网地址](http://jquerymobile.com/) http://jquerymobile.com/

[中文网地址](http://jquerymobile.weebly.com/jquerymobile3245220214.html) http://jquerymobile.weebly.com/jquerymobile3245220214.html

[APi学习地址](http://www.w3school.com.cn/jquerymobile/index.asp) http://www.w3school.com.cn/jquerymobile/index.asp

### Jquery Mobile是做什么的？

jQuery Mobile 是一个为触控优化的web框架，用于创建移动 web 应用程序。

jQuery 适用于所有流行的智能手机和平板电脑

jQuery Mobile 构建于 jQuery 库之上，这使其更易学习，如果您通晓 jQuery 的话。

它使用 HTML5、CSS3、JavaScript 和 AJAX 通过尽可能少的代码来完成对页面的布局。

## AppCan介绍

### 官网地址

[官网地址](http://www.appcan.cn/) <http://www.appcan.cn/>

### 首先来认识AppCan是什么？

AppCan.cn开发平台是基于HTML5技术的跨平台移动应用快速开发一体化解决方案。开发者利用HTML5+CSS3+JavaScript技术可 以快速地开发与本地应用体验相媲美的移动应用。AppCan.cn平台提供了UI快速开发框架、本地功能调用API接口、应用打包系统、IDE集成开发环 境和本地应用调试模拟器，预置数百套界面模板和数十种应用插件，提供多套应用模板。完善的框架接口，人性化的开发环境，丰富的开发资源，强大的服务支持， 使开发者可以快速迈入移动开发领域。

### 谁在用AppCan?

[项目演示](http://www.appcan.cn/appshow/index.html) http://www.appcan.cn/appshow/index.html

# 学习方法介绍



## 思想态度

积极努力

没有人能平白无故的得到成功

利用一切时间

## 价值观

脚踏实地，厚积薄发

懂得取舍，莫扣细节

## 新技术学习的本质？

一门新技术无非就是一些新概念的提出，伴随着一套完整的规则约定。

## 如何学习？

1、官网看docs

2、官网看不懂找相应的中文网

3、根据快速开始的指引或者从github上找一些快速开始的kit学习，勤练习，至少要把大部分的api都搞过一次。

4、加qq群，微信群

5、搜博客和视频

## 每天都应该上的几个网站？

[OSChina](http://www.oschina.net/)：https://www.oschina.net/

[InfoQ](http://www.infoq.com/cn/)：http://www.infoq.com/cn/

[CSDN](http://www.csdn.net/)：http://www.csdn.net/

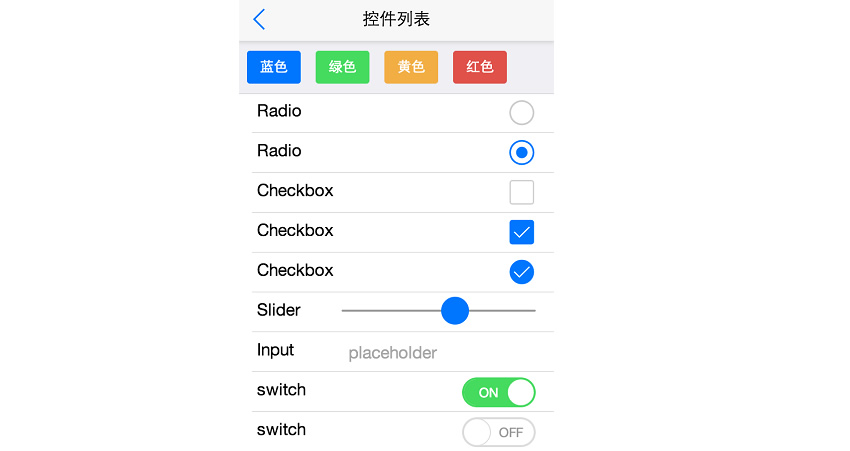
[Cnode社区](https://cnodejs.org/)：https://cnodejs.org/

# 第五章Mui

**MUI为何诞生**1.性能和体验的差距，一直是mobile app开发者放弃HTML5的首要原因。

2.浏览器天生的切页白屏、不忍直视的转页动画、浮动元素的抖动、无法流畅下拉刷新、侧滑抽屉卡顿等问题，这些都让HTML5开发者倍感挫败，尤其拿到Android低端机运行，摔手机的心都有；

另一方面，浏览器默认控件样式又少又丑，制作一个漂亮的控件非常麻烦，也有一些制作简单的ui框架但性能低下。mui框架有效的解决了这些问题，这是一个可以方便开发出高性能App的框架，也是目前最接近原生App效果的框架。

  
**MUI的定位是：最接近原生体验的移动App的UI框架**

基于mui的定位，产生了mui的几个特点，轻、小、只涉及UI、只为移动App而生、界面风格原生化。

所以请大家注意，mui有所为有所不为：  
 **1、mui不是jq，不封装dom操作**

与ui无关的mui不做，你愿意用jq或zepto就自己用，并不冲突。

但我们并不建议在移动App里引入jq或zepto这些框架，原因如下：

为了性能，层层封装的框架，尤其是遍历循环dom时，影响效率，尤其在低端Android手机上，我们费死劲了才把性能以毫秒为单位一点点提升，搞这个的dom框架进来就让很多努力又付诸东流。

**原生JS挺简单，为何需要jq？**

jq的成功当时是因为ie6、7、8、9、10、chrome、ff这些浏览器不兼容，让开发者崩溃，而且pc上浏览器性能好，跨平台兼容也不影响性能。但jq根本就不是为手机设计的。

手机上只有webkit浏览器（忽略wp，反正mui不支持wp），根本就不需要jq这种封装框架来操作dom。

而且HBuilder提供了代码块来简化开发，敲dg、dq，直接生成document.getElementById("")、document.querySelectorAll("")，非常快捷方便，而且执行性能非常高，而且没有浏览器兼容问题。

发现很多开发者只会jq，反正想继续在App里使用jq没有问题。但也建议大家多学学js本身。

**2、mui、HTML5+、5+Runtime的关系说明**

mui是一个前端框架，HTML5+是一套HTML5能力扩展规范，HTML5+ Runtime是实现HTML5+规范的强化浏览器引擎。

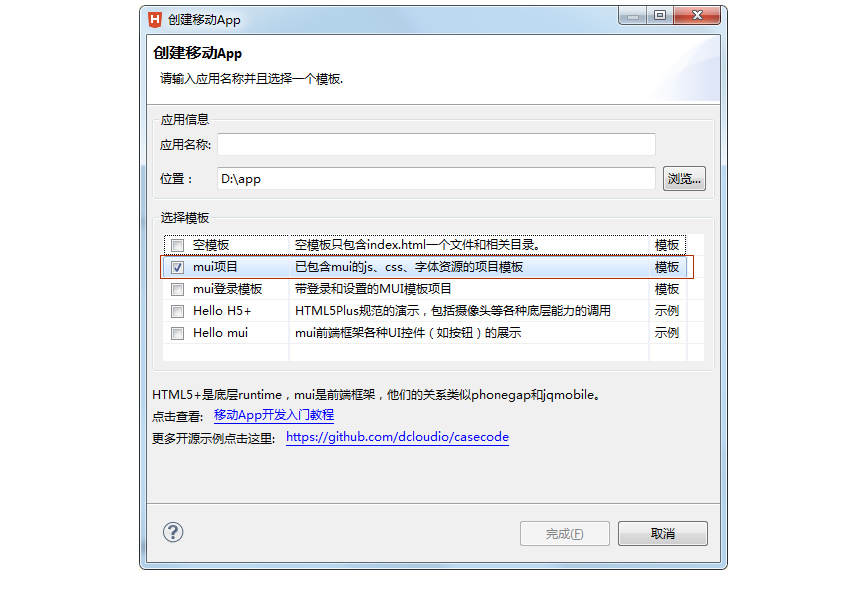
有点类似于bootstrap、w3c和chrome os的关系。

HTML5+规范隶属于http://www.html5plus.org，定义了HTML5规范中没有但开发者做App需要的扩展规范。

DCloud的5+ Runtime完整的实现了HTML5+规范。同时5+ Runtime还实现了Native.js，一种通过js调用几十万原生API的技术。

为了提升体验，mui势必会调用一些5+Rutime的增强能力，主要是plus.webview和plus.nativeUI。

但mui不是要替代HTML5Plus，以后也无计划替代把所有5+的webview api都包一层。

**开始体验MUI  
1、安装新版HBUilder**下载地址: http://www.dcloud.io/  
 **2、新建app项目  
  
3、真机调试及运行**

不管是iOS还是Android，不管是模拟器还是真机，都可以与HBuilder连接进行真机运行。

以往开发App，需要改一个界面，然后打包，然后安装到手机上，然后进入那个界面，看看改对了没。没有的话循环这套动作，非常低效。

有了真机运行，改了代码后保存，手机立即看到效果，如果在手机上运行时发生错误，那么日志和错误信息也都可以反馈到HBuilder控制台。

有些开发者喜欢alert大法，但事实上在HBuilder的js里敲clog，用打日志来调程序更好。  
  
**4、文件结构介绍**  
|\_ css 样式表文件夹  
|\_ fonts 字体文件夹  
|\_ js JavaScript脚本文件夹  
|\_ index.html 入口文件

**基础布局**

**一、头部**核心css mui-bar mui-bar-nav

<header class="mui-bar mui-bar-nav">

<a class="mui-action-back mui-icon mui-icon-left-nav mui-pull-left"></a>

<h1 class="mui-title">hello</h1>

</header>

**二、主体部分**核心css mui-content

<div class="mui-content">

主体部分....

</div>

完整代码演示

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width,initial-scale=1,minimum-scale=1,maximum-scale=1,user-scalable=no" />

<title></title>

<script src="js/mui.min.js"></script>

<link href="css/mui.min.css" rel="stylesheet"/>

<script type="text/javascript" charset="utf-8">

mui.init();

</script>

</head>

<body>

<div>

<header class="mui-bar mui-bar-nav">

<a class="mui-action-back mui-icon mui-icon-left-nav mui-pull-left"></a>

<h1 class="mui-title">hello</h1>

</header>

<div class="mui-content">

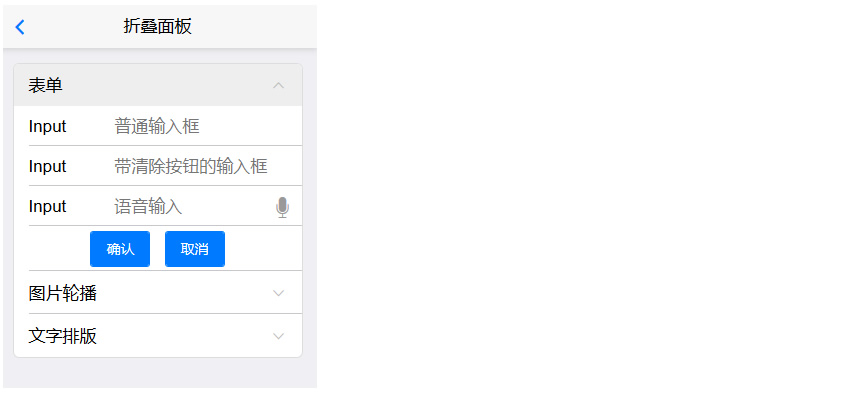
内容部分....

</div>

</body>

</html>

MUI - accordion（折叠面板）、button（按钮）

**1、折叠面板**  


折叠面板从二级列表中演化而来，dom结构和二级列表类似，如下：

<ul class="mui-table-view">

<li class="mui-table-view-cell mui-collapse">

<a class="mui-navigate-right" href="#">面板1</a>

<div class="mui-collapse-content">

<p>面板1子内容</p>

</div>

</li>

</ul>

可以在折叠面板中放置任何内容；折叠面板默认收缩，若希望某个面板默认展开，只需要在包含.mui-collapse类的li节点上，增加.mui-active类即可；mui官网中的方法说明，使用的就是折叠面板控件。  
  
**注意事项：**  
1、折叠面板布局必须在 mui-content下  
2、外层使用 mui-card 包裹产生边缘

3、与mlist结合使用

**完整代码**

<header class="mui-bar mui-bar-nav">

<a class="mui-action-back mui-icon mui-icon-left-nav mui-pull-left"></a>

<h1 class="mui-title">hello</h1>

</header>

<div class="mui-content">

<div class="mui-card">

<ul class="mui-table-view">

<li class="mui-table-view-cell mui-collapse">

<a class="mui-navigate-right" href="#">面板1</a>

<div class="mui-collapse-content">

<p>面板1子内容</p>

</div>

</li>

</ul>

</div>

</div>

**2、按钮  
**

mui默认按钮为灰色，另外还提供了蓝色（blue）、绿色(green)、黄色(yellow)、红色(red)、紫色(purple)五种色系 的按钮，五种色系对应五种场景，分别为primary、success、warning、danger、royal；使用.mui-btn类即可生成一个 默认按钮，继续添加.mui-btn-颜色值或.mui-btn-场景可生成对应色系的按钮，例如：通过.mui-btn-blue或.mui-btn- primary均可生成蓝色按钮；

普通按钮

在button节点上增加.mui-btn类，即可生成默认按钮；若需要其他颜色的按钮，则继续增加对应class即可，比如.mui-btn-blue即可变成蓝色按钮

<button type="button" class="mui-btn">默认</button>

<button type="button" class="mui-btn mui-btn-primary">蓝色</button>

<button type="button" class="mui-btn mui-btn-success">绿色</button>

<button type="button" class="mui-btn mui-btn-warning">黄色</button>

<button type="button" class="mui-btn mui-btn-danger">红色</button>

<button type="button" class="mui-btn mui-btn-royal">紫色</button>

若希望无底色、有边框的按钮，仅需增加.mui-btn-outlined类即可，代码如下：

<button type="button" class="mui-btn mui-btn-outlined">默认</button>

<button type="button" class="mui-btn mui-btn-primary mui-btn-outlined">蓝色</button>

带图标的按钮：<button class="mui-btn mui-icon mui-icon-home mui-right ">蓝色</button>

块级按钮：<button class="mui-btn mui-btn-block mui-btn-royal">紫色</button>

MUI - actionsheet（操作表）、badge（数字角标）

**1、操作表**  
  
actionsheet一般从底部弹出，显示一系列可供用户选择的操作按钮； actionsheet是从popover控件基础上演变而来，实际上就是一个固定从底部弹出的popover，故DOM结构和popove类似，只是需要在含.mui-popover类的节点上增加.mui-popover-bottom、.mui-popover-action类；

<div id="sheet1" class="mui-popover mui-popover-bottom mui-popover-action ">

<!-- 可选择菜单 -->

<ul class="mui-table-view">

<li class="mui-table-view-cell">

<a href="#">菜单1</a>

</li>

<li class="mui-table-view-cell">

<a href="#">菜单2</a>

</li>

</ul>

<!-- 取消菜单 -->

<ul class="mui-table-view">

<li class="mui-table-view-cell">

<a href="#sheet1"><b>取消</b></a>

</li>

</ul>

</div>

推荐使用锚点方式显示、隐藏actionsheet；若要使用js代码动态显示、隐藏actionsheet，同样在popover插件的构造方法中传入"toggle"参数即可，如下：

//传入toggle参数，用户无需关心当前是显示还是隐藏状态，mui会自动识别处理；

mui('#sheet1').popover('toggle');

**完整代码**

<header class="mui-bar mui-bar-nav">

<a class="mui-action-back mui-icon mui-icon-left-nav mui-pull-left"></a>

<h1 class="mui-title">hello</h1>

</header>

<div class="mui-content">

<div style="padding:20px;">

<input type="button" class="mui-btn-blue" value="点击这里弹出操作菜单" onclick="showPop();" />

</div>

</div>

<div id="sheet1" class="mui-popover mui-popover-bottom mui-popover-action ">

<!-- 可选择菜单 -->

<ul class="mui-table-view">

<li class="mui-table-view-cell">

<a href="#">菜单1</a>

</li>

<li class="mui-table-view-cell">

<a href="#">菜单2</a>

</li>

</ul>

<!-- 取消菜单 -->

<ul class="mui-table-view">

<li class="mui-table-view-cell">

<a href="#sheet1"><b>取消</b></a>

</li>

</ul>

</div>

<script type="text/javascript">

function showPop(){

mui('#sheet1').popover('toggle');

}

</script>

**2、数字角标**

数 字角标一般和其它控件（列表、9宫格、选项卡等）配合使用，用于进行数量提示。 角标的核心类是.mui-badge，默认为实心灰色背景；同时，mui还内置了蓝色（blue）、绿色(green)、黄色(yellow)、红色 (red)、紫色(purple)五种色系的数字角标，如下：

<span class="mui-badge mui-badge-inverted">1</span>

<span class="mui-badge mui-badge-primary mui-badge-inverted">2</span>

<span class="mui-badge mui-badge-success mui-badge-inverted">3</span>

<span class="mui-badge mui-badge-warning mui-badge-inverted">4</span>

<span class="mui-badge mui-badge-danger mui-badge-inverted">5</span>

<span class="mui-badge mui-badge-royal mui-badge-inverted">6</span>

若无需底色，则增加.mui-badge-inverted类即可，如下：

<span class="mui-badge mui-badge-inverted">1</span>

<span class="mui-badge mui-badge-primary mui-badge-inverted">2</span>

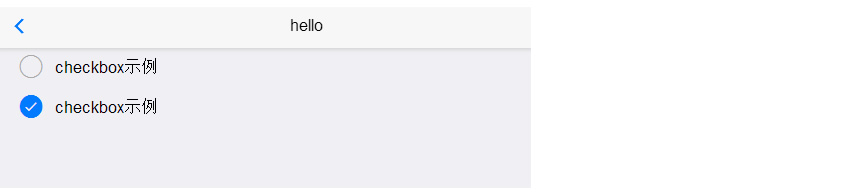
<span class="mui-badge mui-badge-success mui-badge-inverted">3</span>

<span class="mui-badge mui-badge-warning mui-badge-inverted">4</span>

<span class="mui-badge mui-badge-danger mui-badge-inverted">5</span>

<span class="mui-badge mui-badge-royal mui-badge-inverted">6</span>

MUI - 复选框、单选框、使用js获取选择值

**1、复选框**  
  
checkbox常用于多选的情况，比如批量删除、添加等；

**DOM结构**

<div class="mui-input-row mui-checkbox">

<label>checkbox示例</label>

<input name="checkbox1" value="Item 1" type="checkbox" checked>

</div>

默认checkbox在右侧显示，若希望在左侧显示，只需增加.mui-left类即可，如下：

<div class="mui-input-row mui-checkbox mui-left">

<label>checkbox左侧显示示例</label>

<input name="checkbox1" value="Item 1" type="checkbox">

</div>

若要禁用checkbox，只需在checkbox上增加disabled属性即可；

**2、单选框**  
radio用于单选的情况  
DOM结构

<div class="mui-input-row mui-radio">

<label>radio</label>

<input name="radio1" type="radio">

</div>

默认radio在右侧显示，若希望在左侧显示，只需增加.mui-left类即可，如下：

<div class="mui-input-row mui-radio mui-left">

<label>radio</label>

<input name="radio1" type="radio">

</div>

若要禁用radio，只需在radio上增加disabled属性即可；  
mui基于列表控件，提供了列表式单选实现；在列表根节点上增加.mui-table-view-radio类即可，若要默认选中某项，只需要在对应li节点上增加.mui-selected类即可，dom结构如下：

<ul class="mui-table-view mui-table-view-radio">

<li class="mui-table-view-cell">

<a class="mui-navigate-right">Item 1</a>

</li>

<li class="mui-table-view-cell mui-selected">

<a class="mui-navigate-right">Item 2</a>

</li>

<li class="mui-table-view-cell">

<a class="mui-navigate-right">Item 3</a>

</li>

</ul>

列表式单选在切换选中项时会触发selected事件，在事件参数（e.detail.el）中可以获得当前选中的dom节点，如下代码打印当前选中项的innerHTML：

var list = document.querySelector('.mui-table-view.mui-table-view-radio');

list.addEventListener('selected',function(e){

console.log("当前选中的为："+e.detail.el.innerText);

});

**3、js获取单选按钮的值**

function getVals(){

var res = getRadioRes('rds');

if(res == null){mui.toast('请选择'); return;}

mui.toast(res);

}

function getRadioRes(className){

var rdsObj = document.getElementsByClassName(className);

var checkVal = null;

for(i = 0; i < rdsObj.length; i++){

if(rdsObj[i].checked){checkVal = rdsObj[i].value;}

}

return checkVal;

}

</script>

**4、js获取复选框的值**

function getVals(){

var res = getCheckBoxRes('rds');

if(res.length < 1){

mui.toast('请选择');

return;

}

mui.toast(res);

}

function getCheckBoxRes(className){

var rdsObj = document.getElementsByClassName(className);

var checkVal = new Array();

var k = 0;

for(i = 0; i < rdsObj.length; i++){

if(rdsObj[i].checked){

checkVal[k] = rdsObj[i].value;

k++;

}

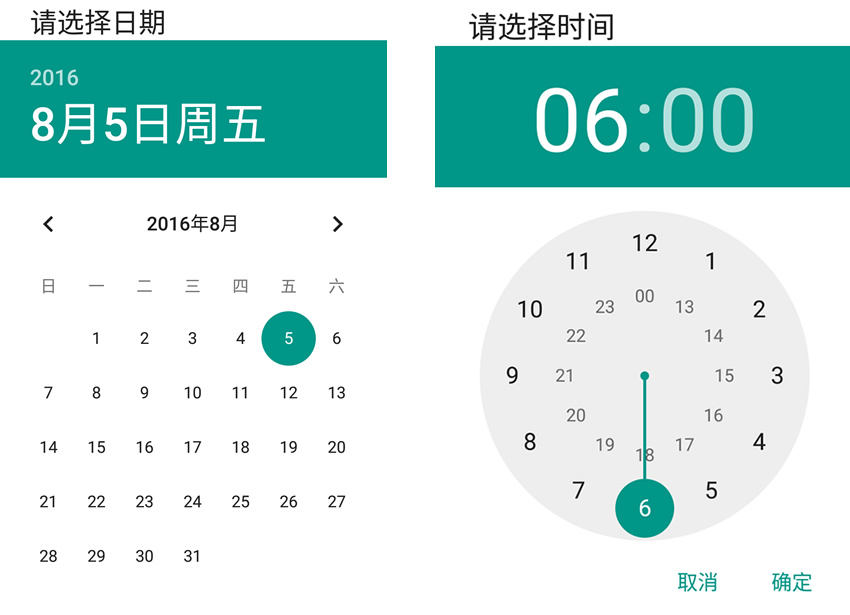
}

return checkVal;

}

MUI - datepicker（时间选择器）

教程内容

  
**完整代码：**

<header class="mui-bar mui-bar-nav">

<a class="mui-action-back mui-icon mui-icon-left-nav mui-pull-left"></a>

<h1 class="mui-title">hello</h1>

</header>

<div class="mui-content">

<div style="padding:15px;">

<button id='pickDateBtn' type="button" class="mui-btn">选择日期</button>

</div>

<div style="padding:15px;">

<button id='pickTimeBtn' type="button" class="mui-btn">选择时间</button>

</div>

</div>

<script type="text/javascript">

document.getElementById("pickDateBtn").addEventListener('tap', function() {

var dDate = new Date();

//设置当前日期（不设置默认当前日期）

dDate.setFullYear(2016, 7, 16);

var minDate = new Date();

//最小时间

minDate.setFullYear(2010, 0, 1);

var maxDate = new Date();

//最大时间

maxDate.setFullYear(2016, 11, 31);

plus.nativeUI.pickDate(function(e) {

var d = e.date;

mui.toast('您选择的日期是:' + d.getFullYear() + "-" + (d.getMonth() + 1) + "-" + d.getDate());

}, function(e) {

mui.toast("您没有选择日期");

}, {

title: '请选择日期',

date: dDate,

minDate: minDate,

maxDate: maxDate

});

});

document.getElementById("pickTimeBtn").addEventListener('tap', function() {

var dTime = new Date();

//设置默认时间

dTime.setHours(6, 0);

plus.nativeUI.pickTime(function(e) {

var d = e.date;

mui.toast("您选择的时间是：" + d.getHours() + ":" + d.getMinutes());

}, function(e) {

mui.toast("您没有选择时间");

}, {

title: "请选择时间",

is24Hour: true,

time: dTime

});

});

**js获取当前时间**

var myDate = new Date();

myDate.getYear(); //获取当前年份(2位)

myDate.getFullYear(); //获取完整的年份(4位,1970-????)

myDate.getMonth(); //获取当前月份(0-11,0代表1月)

myDate.getDate(); //获取当前日(1-31)

myDate.getDay(); //获取当前星期X(0-6,0代表星期天)

myDate.getTime(); //获取当前时间(从1970.1.1开始的毫秒数)

myDate.getHours(); //获取当前小时数(0-23)

myDate.getMinutes(); //获取当前分钟数(0-59)

myDate.getSeconds(); //获取当前秒数(0-59)

myDate.getMilliseconds(); //获取当前毫秒数(0-999)

myDate.toLocaleDateString(); //获取当前日期

var mytime=myDate.toLocaleTimeString(); //获取当前时间

myDate.toLocaleString( ); //获取日期与时间

MUI - dialog对话框、ipnut (表单)

教程内容

  
**1、mui.alert() 普通提醒**  
参数

1、message Type: String 提示对话框上显示的内容

2、title Type: String 提示对话框上显示的标题

3、btnValue Type: String 提示对话框上按钮显示的内容

4、callback Type: Function 提示对话框上关闭后的回调函数

5、type Value: 'div'是否使用h5绘制的对话框

演示代码

mui.alert('欢迎使用Hello MUI', 'Hello MUI', function() {

mui.toast('你刚关闭了警告框');

});

**2、mui.confirm() 确定消息框**  
参数

1、message Type: String 提示对话框上显示的内容

2、title Type: String提示对话框上显示的标题

3、btnValue Type: Array 提示对话框上按钮显示的内容

4、callback Type: Function 提示对话框上关闭后的回调函数

5、type Value: 'div' 是否使用h5绘制的对话框

演示代码

var btnArray = ['取消', '确定'];

mui.confirm('真的要删除吗？','Hi...',new Array('否','是'),function(e){

if(e.index == 1){mui.toast('是');}else{mui.toast('否');}

});

**3、mui.prompt() 输入框**  
参数

message Type: String 提示对话框上显示的内容

placeholder Type: String 编辑框显示的提示文字

title Type: String 提示对话框上显示的标题

btnValue Type: Array 提示对话框上按钮显示的内容

callback Type: Function提示对话框上关闭后的回调函数

type Value: 'div' 是否使用h5绘制的对话框

演示代码

var btnArray = ['取消', '确定'];

mui.prompt('请输入您的姓名？','Hi...',new Array('取消','确定'),function(e){

if(e.index == 1){

mui.toast(e.value);

}else{

mui.toast('您取消了输入');

}

});

**4、mui.toast() 自动消失的消息框**

mui.toast('hi...');

**5、表单元素**

基本说明:  
所有包裹在.mui-input-row 类中的 input、textarea等元素都将被默认设置宽度属性为width: 100%; 。 将 label 元素和上述控件控件包裹在.mui-input-group中可以获得最好的排列。

<form class="mui-input-group">

<div class="mui-input-row">

<label>用户名</label>

<input type="text" class="mui-input-clear" placeholder="请输入用户名">

</div>

<div class="mui-input-row">

<label>密码</label>

<input type="password" class="mui-input-password" placeholder="请输入密码">

</div>

<div class="mui-button-row">

<button type="button" class="mui-btn mui-btn-primary" >确认</button>

<button type="button" class="mui-btn mui-btn-danger" >取消</button>

</div>

</form>

**输入增强**  
mui目前提供的输入增强包括：快速删除、语音输入\*5+ only和密码框显示隐藏密码。  
1、快速删除：只需要在input控件上添加.mui-input-clear类，当input 控件中有内容时，右侧会有一个删除图标，点击会清空当前input的内容；

<form class="mui-input-group">

<div class="mui-input-row">

<label>快速删除</label>

<input type="text" class="mui-input-clear" placeholder="请输入内容">

</div>

</form>

2、搜索框：在.mui-input-row同级添加.mui-input-search 类，就可以使用search控件

<form class="mui-input-group">

<div class="mui-input-row mui-search">

<input type="search" class="mui-input-clear" placeholder="hcoder">

</div>

</form>

3、密码框：给input元素添加.mui-input-password类即可使用

<form class="mui-input-group">

<div class="mui-input-row">

<label>密码框</label>

<input type="password" class="mui-input-password" placeholder="请输入密码">

</div>

</form>

**初始化**  
mui在mui.init()中会自动初始化基本控件,但是 动态添加的元素需要重新进行初始化

mui('.mui-input-row input').input();

二．轮播(slide)

**轮播**组件是mui提供的一个核心组件，在该核心组件基础上，衍生出了图片轮播、可拖动式图文表格、可拖动式选项卡、左右滑动9宫格等组件，这些组件有较多共同点。  
**Dom构造：**

<div class="mui-slider">

<div class="mui-slider-group">

<!--第一个内容区容器-->

<div class="mui-slider-item">

<!-- 具体内容 -->

</div>

<!--第二个内容区-->

<div class="mui-slider-item">

<!-- 具体内容 -->

</div>

</div>

</div>

**js部分**

<script type="text/javascript">

mui.plusReady(function(){

//获得slider插件对象

var gallery = mui('.mui-slider');

gallery.slider({

interval:5000//自动轮播周期，若为0则不自动播放，默认为0；

});

});

</script>

**显示圆点**

<div class="mui-slider-indicator">

<div class="mui-indicator mui-active"></div>

<div class="mui-indicator"></div>

</div>

**轮播循环**  
若要支持循环，则需要在.mui-slider-group节点上增加.mui-slider-loop类，同时需要重复增加2张图片，图片顺序变为：4、1、2、3、4、1，代码示例如下：

<div class="mui-slider">

<div class="mui-slider-group mui-slider-loop">

<!--支持循环，需要重复图片节点-->

<div class="mui-slider-item mui-slider-item-duplicate"><a href="#"><img src="4.jpg" /></a></div>

<div class="mui-slider-item"><a href="#"><img src="1.jpg" /></a></div>

<div class="mui-slider-item"><a href="#"><img src="2.jpg" /></a></div>

<div class="mui-slider-item"><a href="#"><img src="3.jpg" /></a></div>

<div class="mui-slider-item"><a href="#"><img src="4.jpg" /></a></div>

<!--支持循环，需要重复图片节点-->

<div class="mui-slider-item mui-slider-item-duplicate"><a href="#"><img src="1.jpg" /></a></div>

</div>

</div>

**轮播跳转**  
若要跳转到第x张图片，则可以使用图片轮播插件的gotoItem方法，例如：

var gallery = mui('.mui-slider');

gallery.slider().gotoItem(index);//跳转到第index张图片，index从0开始；

**轮播事件**  
当拖动切换显示内容时，会触发slide事件，通过该事件的detail.slideNumber参数可以获得当前显示项的索引（第一项索引为0，第二项为1，以此类推），利用该事件，可在显示内容切换时，动态处理一些业务逻辑。  
  
如下为一个可拖动式选项卡示例，为提高页面加载速度，页面加载时，仅显示第一个选项卡的内容，第二、第三选项卡内容为空。  
当切换到第二、第三个选项卡时，再动态获取相应内容进行显示：

var item2Show = false,item3Show = false;//子选项卡是否显示标志

document.querySelector('.mui-slider').addEventListener('slide', function(event) {

if (event.detail.slideNumber === 1&&!item2Show) {

//切换到第二个选项卡

//根据具体业务，动态获得第二个选项卡内容；

var content = ....

//显示内容

document.getElementById("item2").innerHTML = content;

//改变标志位，下次直接显示

item2Show = true;

} else if (event.detail.slideNumber === 2&&!item3Show) {

//切换到第三个选项卡

//根据具体业务，动态获得第三个选项卡内容；

var content = ....

//显示内容

document.getElementById("item3").innerHTML = content;

//改变标志位，下次直接显示

item3Show = true;

}

});

三．MUI - list（列表/图文列表）

**1、普通列表**  
列表是常用的UI控件，mui封装的列表组件比较简单，只需要在ul节点上添加.mui-table-view类、在li节点上添加.mui-table-view-cell类即可，如下为示例代码

<ul class="mui-table-view">

<li class="mui-table-view-cell">Item 1</li>

<li class="mui-table-view-cell">Item 2</li>

<li class="mui-table-view-cell">Item 3</li>

</ul>

若右侧需要增加导航箭头，变成导航链接，则只需在li节点下增加a子节点，并为该a节点增加.mui-navigate-right类即可，如下：

<ul class="mui-table-view">

<li class="mui-table-view-cell"><a class="mui-navigate-right">Item 1</a></li>

<li class="mui-table-view-cell"><a class="mui-navigate-right">Item 2</a></li>

<li class="mui-table-view-cell"><a class="mui-navigate-right">Item 3</a></li>

</ul>

mui支持将数字角标、按钮、开关等控件放在列表中；mui默认将数字角标放在列表右侧显示，代码如下：

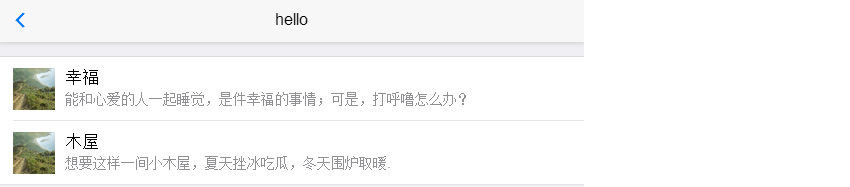
<ul class="mui-table-view">

<li class="mui-table-view-cell">Item 1<span class="mui-badge mui-badge-primary">11</span></li>

<li class="mui-table-view-cell">Item 2<span class="mui-badge mui-badge-success">22</span></li>

<li class="mui-table-view-cell">Item 3<span class="mui-badge">33</span></li>

</ul>

**2、图文列表  
**  
图文列表继承自列表组件，主要添加了.mui-media、.mui-media-object、.mui-media-body几个类，如下为示例代码：

<ul class="mui-table-view">

<li class="mui-table-view-cell mui-media">

<a href="javascript:;">

<img class="mui-media-object mui-pull-left" src="imgs/1.jpg">

<div class="mui-media-body">

幸福

<p class='mui-ellipsis'>能和心爱的人一起睡觉，是件幸福的事情；可是，打呼噜怎么办？</p>

</div>

</a>

</li>

<li class="mui-table-view-cell mui-media">

<a href="javascript:;">

<img class="mui-media-object mui-pull-left" src="imgs/1.jpg">

<div class="mui-media-body">

木屋

<p class='mui-ellipsis'>想要这样一间小木屋，夏天挫冰吃瓜，冬天围炉取暖.</p>

</div>

</a>

</li>

</ul>

**四MUI - progressbar（进度条）、滑块及switch开关**

**1、有准确值的进度条  
**  
有准确值的进度条会显示当前进度正处于整体进度的占比位置，用户可以更直观的预期完成时间；  
使用进度条控件，需要一个进度条控件容器，mui会自动识别该容器下是否有进度条控件，若没有，则自动创建。  
进度条控件DOM结构：

<div id="demo1" class="mui-progressbar">

<span></span>

</div>

初始化:

mui(container).progressbar({progress:20}).show();

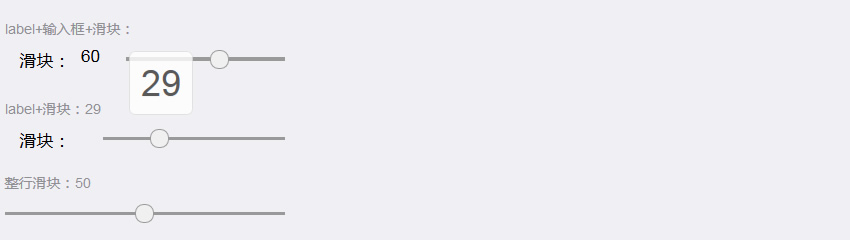
progressbar初始化逻辑：  
检查当前容器(container控件)自身是否包含.mui-progressbar类：  
当前容器包含.mui-progressbar类，则以当前容器为目标控件，直接显示进度；  
否则，检查当前容器的直接孩子节点是否包含.mui-progressbar类，若存在，则以遍历到的第一个含有.mui-progressbar类的孩子节点为目标控件，显示进度；  
若当前容器的直接孩子节点，均不含.mui-progressbar类,则自动创建进度条控件；  
更改显示进度条:

mui(container).progressbar().setProgress(50);

关闭进度条:

mui(container).progressbar().hide();

备注：关闭进度条一般用于动态创建（DOM中预先未定义）的进度条，调用hide方法后，会直接删掉对应的DOM节点；若已提前创建好DOM节点的进度条，调用hide方法无效；

**2、滑块**  


<div class="mui-input-row mui-input-range">

<label>Range</label>

<input type="range" min="0" max="100">

</div>

获取滑块数值的样例代码

<div class="mui-content">

<div class="mui-input-row mui-input-range">

<label>Range</label>

<input type="range" min="0" max="100" id="range1">

</div>

<div style="padding:20px;">

<input type="button" class="mui-btn" value="获取滑块值" onclick="getVal();" />

</div>

</div>

<script>

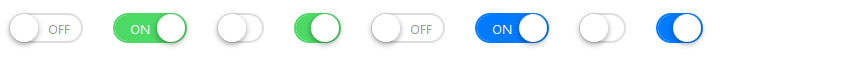
function getVal(){

var rangeObj = mui('#range1');

mui.toast(rangeObj[0].value);

}

</script>

**3、switch 开关**  
mui提供了开关控件，点击滑动两种手势都可以对开关控件进行操作,UI如下：  
  
默认开关控件,带on/off文字提示，打开时为绿色背景，基本class类为.mui-switch、.mui-switch-handle，DOM结构如下：

<div class="mui-switch">

<div class="mui-switch-handle"></div>

</div>

若希望开关默认为打开状态，只需要在.mui-switch节点上增加.mui-active类即可，如下：

<!-- 开关打开状态，多了一个.mui-active类 -->

<div class="mui-switch mui-active">

<div class="mui-switch-handle"></div>

</div>

若希望隐藏on/off文字提示，变成简洁模式，需要在.mui-switch节点上增加.mui-switch-mini类，如下：

<!-- 简洁模式开关关闭状态 -->

<div class="mui-switch mui-switch-mini">

<div class="mui-switch-handle"></div>

</div>

<!-- 简洁模式开关打开状态 -->

<div class="mui-switch mui-switch-mini mui-active">

<div class="mui-switch-handle"></div>

</div>

mui默认还提供了蓝色开关控件，只需在.mui-switch节点上增加.mui-switch-blue类即可，如下：

<!-- 蓝色开关关闭状态 -->

<div class="mui-switch mui-switch-blue">

<div class="mui-switch-handle"></div>

</div>

<!-- 蓝色开关打开状态 -->

<div class="mui-switch mui-switch-blue mui-active">

<div class="mui-switch-handle"></div>

</div>

蓝色开关上增加.mui-switch-mini即可变成无文字的简洁模式  
  
若要获得当前开关状态，可通过判断当前开关控件是否包含.mui-active类来实现，若包含，则为打开状态，否则即为关闭状态；如下为代码示例：

var isActive = document.getElementById("mySwitch").classList.contains("mui-active");

if(isActive){

console.log("打开状态");

}else{

console.log("关闭状态");

}

若使用js打开、关闭开关控件，可使用switch插件的toggle()方法，如下为示例代码：

mui("#mySwitch").switch().toggle();

**事件**  
开关控件在打开/关闭两种状态之间进行切换时，会触发toggle事件,通过事件的detail.isActive属性可以判断当前开关状态。可通过监听toggle事件，可以在开关切换时执行特定业务逻辑。如下为使用示例：

document.getElementById("mySwitch").addEventListener("toggle",function(event){

if(event.detail.isActive){

console.log("你启动了开关");

}else{

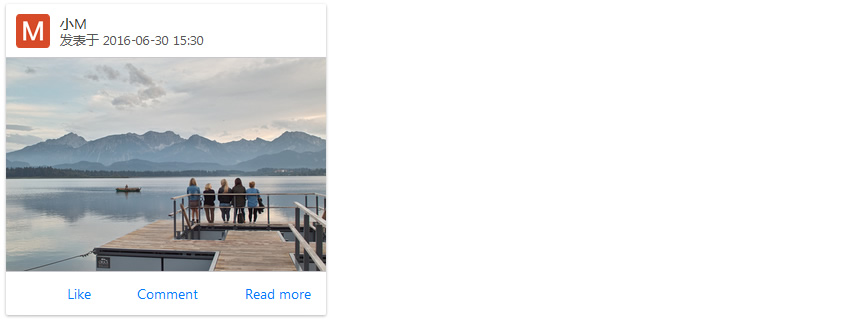
console.log("你关闭了开关");

}

})

五．MUI - cardview（卡片视图）、mask（遮罩蒙版）

**1、cardview（卡片视图）**



使用mui-card类即可生成一个卡片容器，卡片视图主要有页眉、内容区、页脚三部分组成，结构如下：

<div class="mui-card">

<!--页眉，放置标题-->

<div class="mui-card-header">页眉</div>

<!--内容区-->

<div class="mui-card-content">内容区</div>

<!--页脚，放置补充信息或支持的操作-->

<div class="mui-card-footer">页脚</div>

</div>

卡片页眉及内容区，均支持放置图片； 页眉放置图片的话，需要在.mui-card-header节点上增加.mui-card-media类，然后设置一张图片做背景图即可，代码如下：

<div class="mui-card-header mui-card-media" style="height:40px;background-image:url(../images/cbd.jpg)"></div>

若希望在页眉放置更丰富的信息，比如头像、主标题、副标题，则需使用.mui-media-body类，示例代码如下：

<div class="mui-card-header mui-card-media">

<img src="../images/logo.png" />

<div class="mui-media-body">

小M

<p>发表于 2016-06-30 15:30</p>

</div>

</div>

若希望在页脚放置更丰富的信息, 则需使用.mui-media-body类，示例代码如下：

<div class="mui-card-footer">

<div class="mui-media-body">

<h4>美图</h4>

<p>发表于2018-02-09 11:23:23</p>

</div>

</div>

**小技巧**图片出现默认的空白解决方案：  
方式1.给包裹图片的div设置：line-height:0px;

方式2.给图片添加属性：style="display: block;"  
 **2、mask（遮罩蒙版）**

在popover、侧滑菜单等界面，经常会用到蒙版遮罩；比如popover弹出后，除popover控件外的其它区域都会遮罩一层蒙版，用户点击蒙版不会触发蒙版下方的逻辑，而会关闭popover同时关闭蒙版；再比如侧滑菜单界面，菜单划出后，除侧滑菜单之外的其它区域都会遮罩一层蒙版，用户点击蒙版会关闭侧滑菜单同时关闭蒙版。

遮罩蒙版常用的操作包括：创建、显示、关闭，如下代码：

var mask = mui.createMask(callback);//callback为用户点击蒙版时自动执行的回调；

mask.show();//显示遮罩

mask.close();//关闭遮罩

注意：关闭遮罩仅会关闭，不会销毁；关闭之后可以再次调用mask.show();打开遮罩；

mui默认的蒙版遮罩使用.mui-backdrop类定义（如下代码），若需自定义遮罩效果，只需覆盖定义.mui-backdrop即可；

.mui-backdrop {position: fixed;top: 0;right: 0;bottom: 0;left: 0;z-index: 998;background-color: rgba(0,0,0,.3);}

六．MUI - 窗口管理及窗口之间的数据传递

**1、mui插件初始化**  
使用 mui.init(); 进行mui插件初始化。  
  
**2、页面初始化**

在app开发中，若要使用HTML5+扩展api，必须等plusready事件发生后才能正常使用，mui将该事件封装成了mui.plusReady()方法，涉及到HTML5+的api，建议都写在mui.plusReady方法中。如下为打印当前页面URL的示例：

mui.plusReady(function(){

console.log("当前页面URL："+plus.webview.currentWebview().getURL());

});

**3、创建子页面**  
在移动app开发过程中，经常遇到卡头卡尾的页面，此时若使用局部滚动，在android手机上会出现滚动不流畅的问题； mui的解决思路是：将需要滚动的区域通过单独的webview实现，完全使用原生滚动。具体做法则是：将目标页面分解为主页面和内容页面，主页面显示卡头卡尾区域，比如顶部导航、底部选项卡等；内容页面显示具体需要滚动的内容，然后在主页面中调用mui.init方法初始化内容页面。

mui.init({

subpages:[{

url:your-subpage-url,//子页面HTML地址，支持本地地址和网络地址

id:your-subpage-id,//子页面标志

styles:{

top:subpage-top-position,//子页面顶部位置

bottom:subpage-bottom-position,//子页面底部位置

width:subpage-width,//子页面宽度，默认为100%

height:subpage-height,//子页面高度，默认为100%

},

extras:{}//额外扩展参数

}]

});

参数说明：styles表示窗口属性，参考5+规范中的WebviewStyle；特别注意，height和width两个属性,即使不设置，也默认按100%计算；因此若设置了top值为非"0px"的情况，建议同时设置bottom值，否则5+ runtime根据高度100%计算，可能会造成页面真实底部位置超出屏幕范围的情况；left、right同理。  
 **代码样例**  
主页面index.html

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width,initial-scale=1,minimum-scale=1,maximum-scale=1,user-scalable=no" />

<title></title>

<script src="js/mui.min.js"></script>

<link href="css/mui.min.css" rel="stylesheet"/>

</head>

<body>

<header class="mui-bar mui-bar-nav">

<a class="mui-action-back mui-icon mui-icon-left-nav mui-pull-left"></a>

<h1 class="mui-title">hello</h1>

</header>

</body>

<script type="text/javascript">

mui.init({

subpages:[{

url :'sub.html',

id : 'sub.html',

styles:{

top : '45px',//子页面顶部位置

bottom : '0px',//子页面底部位置

width : '100%',

height : '100%'

},

}]

});

</script>

</html>

子页面 sub.html

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width,initial-scale=1,minimum-scale=1,maximum-scale=1,user-scalable=no" />

<title></title>

<script src="js/mui.min.js"></script>

<link href="css/mui.min.css" rel="stylesheet"/>

<script type="text/javascript" charset="utf-8">

mui.init();

</script>

<div class="mui-content">

<div class="mui-card">

<!--页眉，放置标题-->

<div class="mui-card-header mui-card-media">

<img src="imgs/logo.png" />

<div class="mui-media-body">

小M

<p>发表于 2016-06-30 15:30</p>

</div>

</div>

<!--内容区-->

<div class="mui-card-content" style="overflow:hidden; padding:0px; line-height:0px;">

<img src="imgs/1.jpg" width="100%" />

</div>

<!--页脚，放置补充信息或支持的操作-->

<div class="mui-card-footer">页脚</div>

</div>

<div class="mui-card">

<!--页眉，放置标题-->

<div class="mui-card-header mui-card-media">

<img src="imgs/logo.png" />

<div class="mui-media-body">

小M

<p>发表于 2016-06-30 15:30</p>

</div>

</div>

<!--内容区-->

<div class="mui-card-content" style="overflow:hidden; padding:0px;">

<img src="imgs/1.jpg" width="100%" />

</div>

<!--页脚，放置补充信息或支持的操作-->

<div class="mui-card-footer">页脚</div>

</div>

</div>

</body>

</html>

**4、打开新页面**  
做web app，一个无法避开的问题就是转场动画；web是基于链接构建的，从一个页面点击链接跳转到另一个页面，如果通过有刷新的打开方式，用户要面对一个空白的页面等待；如果通过无刷新的方式，用Javascript移入DOM节点（常见的SPA解决方案），会碰到很高的性能挑战：DOM节点繁多，页面太大，转场动画不流畅甚至导致浏览器崩溃； mui的解决思路是：单webview只承载单个页面的dom，减少dom层级及页面大小；页面切换使用原生动画，将最耗性能的部分交给原生实现.

mui.openWindow({

url:new-page-url,

id:new-page-id,

styles:{

top:newpage-top-position,//新页面顶部位置

bottom:newage-bottom-position,//新页面底部位置

width:newpage-width,//新页面宽度，默认为100%

height:newpage-height,//新页面高度，默认为100%

......

},

extras:{

.....//自定义扩展参数，可以用来处理页面间传值

},

createNew:false,//是否重复创建同样id的webview，默认为false:不重复创建，直接显示

show:{

autoShow:true,//页面loaded事件发生后自动显示，默认为true

aniShow:animationType,//页面显示动画，默认为”slide-in-right“；

duration:animationTime//页面动画持续时间，Android平台默认100毫秒，iOS平台默认200毫秒；

},

waiting:{

autoShow:true,//自动显示等待框，默认为true

title:'正在加载...',//等待对话框上显示的提示内容

options:{

width:waiting-dialog-widht,//等待框背景区域宽度，默认根据内容自动计算合适宽度

height:waiting-dialog-height,//等待框背景区域高度，默认根据内容自动计算合适高度

......

}

}

})

**代码样例**  
主页面index.html

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width,initial-scale=1,minimum-scale=1,maximum-scale=1,user-scalable=no" />

<title></title>

<script src="js/mui.min.js"></script>

<link href="css/mui.min.css" rel="stylesheet"/>

</head>

<body>

<header class="mui-bar mui-bar-nav">

<a class="mui-action-back mui-icon mui-icon-left-nav mui-pull-left"></a>

<h1 class="mui-title">hello</h1>

</header>

<div class="mui-content">

<button type="button" class="mui-btn" onclick="openWindow();">点击这里打开新窗口</button>

</div>

</body>

<script type="text/javascript">

function openWindow(){

mui.openWindow({

url : 'test.html',

extras : {'name':'小明', 'age':18}

});

}

</script>

</html>

子页面test.html（接收附加参数）

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width,initial-scale=1,minimum-scale=1,maximum-scale=1,user-scalable=no" />

<title></title>

<script src="js/mui.min.js"></script>

<link href="css/mui.min.css" rel="stylesheet"/>

</head>

<body>

<header class="mui-bar mui-bar-nav">

<a class="mui-action-back mui-icon mui-icon-left-nav mui-pull-left"></a>

<h1 class="mui-title">test</h1>

</header>

<div class="mui-content">

<div>姓名 : <span id="name"></span></div>

<div>年龄 : <span id="age"></span></div>

</div>

</body>

<script type="text/javascript">

mui.init();

mui.plusReady(function(){

var sData = plus.webview.currentWebview();

var nameObj = mui('#name');

nameObj[0].innerHTML = sData.name;

var ageObj = mui('#age');

ageObj[0].innerHTML = sData.age;

});

</script>

</html>

**5、预加载**

所谓的预加载技术就是在用户尚未触发页面跳转时，提前创建目标页面，这样当用户跳转时，就可以立即进行页面切换，节省创建新页面的时间，提升app使用体验。mui提供两种方式实现页面预加载。

方式一：通过mui.init方法中的preloadPages参数进行配置.

mui.init({

preloadPages:[{

url:prelaod-page-url,

id:preload-page-id,

styles:{},//窗口参数

extras:{},//自定义扩展参数

subpages:[{},{}]//预加载页面的子页面

}],

preloadLimit:5//预加载窗口数量限制(一旦超出,先进先出)默认不限制

});

该种方案使用简单、可预加载多个页面，但不会返回预加载每个页面的引用，若要获得对应webview引用，还需要通过plus.webview.getWebviewById方式获得；另外，因为mui.init是异步执行，执行完mui.init方法后立即获得对应webview引用，可能会失败，例如如下代码：

mui.init({

preloadPages:[{

url:'list.html',

id:'list'

}]

});

var list = plus.webview.getWebviewByid('list');//这里可能返回空；

方式二：通过mui.preload方法预加载.

var page = mui.preload({

url:new-page-url,

id:new-page-id,//默认使用当前页面的url作为id

styles:{},//窗口参数

extras:{}//自定义扩展参数

});

通过mui.preload()方法预加载，可立即返回对应webview的引用，但一次仅能预加载一个页面；若需加载多个webview，则需多次调用mui.preload()方法；

如上两种方案，各有优劣，需根据具体业务场景灵活选择；

七．MUI - 事件管理及自定义事件详解

**1、事件绑定**可以使用addEventListener()方法监听某个特定元素上的事件外， 也可以使用.on()方法实现**批量元素**的事件绑定。  
addEventListener示例（单个元素事件绑定）：

mui.plusReady(function(){

document.getElementById('but1').addEventListener('tap',function(){

alert(1);

});

});

on()示例，适用于批量元素绑定。演示代码：

<div class="mui-content">

<ul id="list1">

<li>小明</li>

<li>小花</li>

<li>小萌</li>

</ul>

</div>

</body>

<script type="text/javascript">

mui.plusReady(function(){

mui('#list1').on('tap', 'li', function(){

var thisHtml = this.innerHTML;

alert(thisHtml);

});

});

</script>

**2、事件取消**  
使用on()方法绑定事件后，若希望取消绑定，则可以使用off()方法。

mui('#list1').off('tap', 'li');

**3、事件触发**  
使用mui.trigger()方法可以动态触发特定DOM元素上的事件。

var btn = document.getElementById("submit");

//监听点击事件

btn.addEventListener("tap",function () {console.log("tap event trigger");});

//触发submit按钮的点击事件

mui.trigger(btn,'tap');

**4、手势事件**  
在开发移动端的应用时，会用到很多的手势操作，比如滑动、长按等，为了方便开放者快速集成这些手势，mui内置了常用的手势事件，目前支持的手势事件见如下列表：

点击

tap 单击屏幕

doubletap 双击屏幕

长按

longtap 长按屏幕

hold 按住屏幕

release 离开屏幕

滑动

swipeleft 向左滑动

swiperight 向右滑动

swipeup 向上滑动

swipedown 向下滑动

拖动

dragstart 开始拖动

drag 拖动中

dragend 拖动结束

**5、自定义事件**  
在App开发中，经常会遇到页面间传值的需求，比如从新闻列表页进入详情页，需要将新闻id传递过去； Html5Plus规范设计了evalJS方法来解决该问题； 但evalJS方法仅接收字符串参数，涉及多个参数时，需要开发人员手动拼字符串； 为简化开发，mui框架在evalJS方法的基础上，封装了自定义事件，通过自定义事件，用户可以轻松实现多webview间数据传递。  
添加自定义事件监听操作和标准js事件监听类似，可直接通过window对象添加，如下：

window.addEventListener('customEvent',function(event){mui.toast('ok');});

**6、触发自定义事件**  
通过mui.fire()方法可触发目标窗口的自定义事件：  
参数

target

Type: WebviewObject

需传值的目标webview

event

Type: String

自定义事件名称

data

Type: JSON

json格式的数据

**自定义事件完整例子**主页面:

<div class="mui-content">

<ul id="list1">

<li id="1">新闻1</li>

<li id="2">新闻2</li>

<li id="3">新闻3</li>

</ul>

</div>

</body>

<script type="text/javascript">

mui.init();

mui.plusReady(function(){

var detailPage = mui.preload({url:'sub.html',id:'sub.html'});

mui('#list1').on('tap', 'li', function(e){

mui.fire(detailPage,'newsId',{id:this.getAttribute('id')});

mui.openWindow({id:'sub.html'});

});

});

</script>

子页面

mui.plusReady(function(){

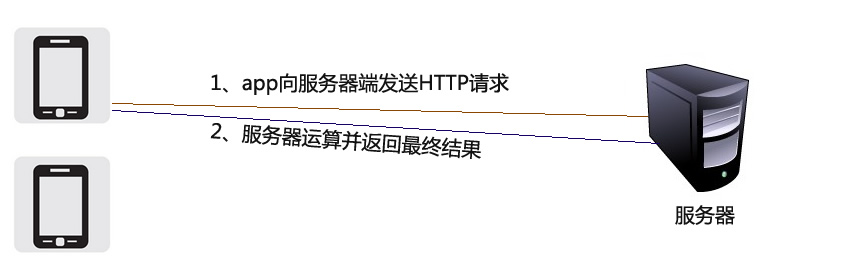
window.addEventListener('newsId',function(event){

console.log(event.detail.id);

});

});

八．APP与服务器之间的交互原理、MUI Ajax使用

**1、APP与服务器之间的交互原理**  
  
app端（客户端）与服务端的交互其实理解起来和容易，客户端想服务器端发送请求，服务器端进行数据运算后返回最终结果。结果可以是多种格式：

1、text 文本格式

2、xml 格式

3、json 格式

**2、什么是JSON？**  
JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式。

JSON 语法规则:

数据在键值对中

数据由逗号分隔

花括号保存对象

方括号保存数组  
简单的例子：

{name:'小明', age:'18'}

**3、使用MUI提供的Ajax工具完成与服务器端的交互**  
mui框架基于htm5plus的XMLHttpRequest，封装了常用的Ajax函数，支持GET、POST请求方式，支持返回json、xml、html、text、script数据类型；   
本着极简的设计原则，mui提供了mui.ajax方法，并在mui.ajax方法基础上，进一步简化出最常用的mui.get()、mui.getJSON()、mui.post()三个方法。  
  
**4、mui.ajax()**  
参数：

url Type: String 请求发送的目标地址

async Type: Boolean 发送同步请求

data {xx:xx,xxx:xxx} 发送到服务器的业务数据

dataType "xml": 返回XML文档,"html": 返回纯文本HTML信息；,"script": 返回纯文本JavaScript代码,"json": 返回JSON数据,"text": 返回纯文本字符串

error 请求失败时触发的回调函数，该函数接收三个参数： (xhr：xhr实例对象,type：错误描述，可取值："timeout", "error", "abort", "parsererror"、"null",errorThrown：可捕获的异常对象)

success 请求成功时触发的回调函数，该函数接收三个参数：(data：服务器返回的响应数据，类型可以是json对象、xml对象、字符串等；,textStatus：状态描述，默认值为'success', xhr：xhr实例对象)

timeout 请求超时时间（毫秒），默认值为0，表示永不超时；若超过设置的超时时间(非0的情况)，依然未收到服务器响应，则触发error回调

type 请求方式，目前仅支持'GET'和'POST'，默认为'GET'方式

headers 指定HTTP请求的Header

一个完整示例

<script type="text/javascript">

mui.init();

mui.plusReady(function(){

document.getElementById('btn1').addEventListener('tap',function(){

mui.ajax({

url : 'http://demo.hcoder.net/index.php',

type : 'GET',

success : function(data){

mui.toast(data);

},

error : function(xhr,type,errorThrown){

mui.toast(type);

}

});

});

});

</script>

返回JSON形式的数据请观看本视频！  
  
**5、Ajax简化模式**

mui.post()方法是对mui.ajax()的一个简化方法，直接使用POST请求方式向服务器发送数据、且不处理timeout和异常（若需处理异常及超时，请使用mui.ajax()方法），使用方法： mui.post(url[,data][,success][,dataType])，如上登录鉴权代码换成mui.post()后，代码更为简洁，如下：

mui.post('http://server-name/login.php',{

username:'username',

password:'password'

},function(data){

//服务器返回响应，根据响应结果，分析是否登录成功；

...

},'json'

);

mui.get()方法和mui.post()方法类似，只不过是直接使用GET请求方式向服务器发送数据、且不处理timeout和异常（若需处理异常及超时，请使用mui.ajax()方法），使用方法： mui.get(url[,data][,success][,dataType])，如下为获得某服务器新闻列表的代码片段，服务器以json格式返回数据列表

mui.get('http://server-name/list.php',{category:'news'},function(data){

//获得服务器响应

...

},'json'

);

mui.getJSON()方法是在mui.get()方法基础上的更进一步简化，限定返回json格式的数据，其它参数和mui.get()方法一致，使用方法： mui.get(url[,data][,success])，如上获得新闻列表的代码换成mui.getJSON()方法后，更为简洁，如下：

mui.getJSON('http://server-name/list.php',{category:'news'},function(data){

//获得服务器响应

...

}

);

**页面点击切换**

第一步：引入h.js

<script src="js/h.js" type="text/javascript" charset="utf-8"></script>

第二步：初始化分页面

<script type="text/javascript" charset="utf-8">

mui.init({

subpages:[

{

url:"sub1.html",

id:"sub1.html",

styles:{

top:"44px",

bottom:"50px",

width:"100%"

}

},{

url:"sub2.html",

id:"sub2.html",

styles:{

top:"44px",

bottom:"50px",

width:"100%",

left:"100%"

}

},{

url:"sub3.html",

id:"sub3.html",

styles:{

top:"44px",

bottom:"50px",

width:"100%",

left:"200%"

}

},{

url:"sub4.html",

id:"sub4.html",

styles:{

top:"44px",

bottom:"50px",

width:"100%",

left:"300%"

}

}

]

});

//第三部：实现页面切换：给每个底部导航a添加id，

h('#nav1').tap(function(){changeSub(1);});

h('#nav2').tap(function(){changeSub(2);});

h('#nav3').tap(function(){changeSub(3);});

h('#nav4').tap(function(){changeSub(4);});

function changeSub(index){

var sub1 = plus.webview.getWebviewById('sub1.html');

var sub2 = plus.webview.getWebviewById('sub2.html');

var sub3 = plus.webview.getWebviewById('sub3.html');

var sub4 = plus.webview.getWebviewById('sub4.html');

// 1 0 100 200 300

// 2 -100 0 100 200

sub1.setStyle({left:(1-index)\*100+'%'});

sub2.setStyle({left:(2-index)\*100+'%'});

sub3.setStyle({left:(3-index)\*100+'%'});

sub4.setStyle({left:(4-index)\*100+'%'});

}

</script>

**页面滑动功能**:在主页面底部导航nav添加id=”navFooter”,然后在每个sub页面添加如下

var \_self,\_next,\_index,\_prev;

mui.plusReady(function () {

\_self = plus.webview.currentWebview();

\_next = plus.webview.getWebviewById('sub3.html');

\_index = plus.webview.getLaunchWebview();

\_prev = plus.webview.getWebviewById('sub1.html');

//向左滑动

\_self.drag({

direction:'left',

moveMode:"followFinger"

},{

view:\_next,

moveMode:"follow"

},function(e){

console.log(JSON.stringify(e));

if(e.type=="end" && e.result){

\_index.evalJS('h("#navFooter").find("a").removeClass("mui-active");h("#nav3").addClass("mui-active")');

}

});

//向右滑动

\_self.drag({

direction:'right',

moveMode:"followFinger"

},{

view:\_prev,

moveMode:"follow"

},function(e){

console.log(JSON.stringify(e));

if(e.type=="end" && e.result){

\_index.evalJS('h("#navFooter").find("a").removeClass("mui-active");h("#nav1").addClass("mui-active")');

}

});

})

打开页面详情：

mui('.mui-table-view-cell').on('tap','a',function(){

var title = this.querySelector("span").innerText;

var id = this.getAttribute("did");

// console.log(title);

mui.openWindow({

url:'info.html',

id:'info.html',

extras:{

id:id,

title:title

}

});

}) ;

子页面获取数据并发起ajax请求

var title;

mui.plusReady(function () {

var \_self = plus.webview.currentWebview();

title = \_self.title;

console.log(\_self.title);

mui('.mui-title')[0].innerHTML = title;

mui.ajax('json/home.json',{

dataType:'json',//服务器返回json格式数据

type:'post',//HTTP请求类型

timeout:10000,//超时时间设置为10秒；

success:function(data){

// console.log(typeof(data));

var str = "";

for(var i=0;i<data.length;i++){

// mui.toast(data[i].logo);

str+= '<div class="mui-card"><div class="mui-card-header mui-card-media"><img src="'+data[i].logo+'" /><div class="mui-media-body">'+data[i].media+'<p>发表于'+data[i].publish+'</p></div></div><div class="mui-card-content" style="background-color: red;line-height:0px;"><img src="'+data[i].photo+'" width="100%"/></div><div class="mui-card-footer"><div class="mui-media-body"><p>'+data[i].desc+'</p></div></div></div>';

}

mui('.mui-content')[0].innerHTML = str;

},

error:function(xhr,type,errorThrown){

mui.toast(type);

}

});

})

分享功能

document.getElementById('share').addEventListener('tap',function () {

//})

//h('#share').tap(function(){

var actionbuttons = [{title : "微信好友"}, {title : "朋友圈"}];

var actionstyle = {title : "请选择分享类型", cancel : "取消", buttons : actionbuttons};

plus.nativeUI.actionSheet(actionstyle, function(e){

if(e.index){shareNow(e.index);}

});

});

var shareWx = null;

function shareNow(type){

plus.share.getServices(function(services){

for(var k in services){

if(services[k].id == 'weixin'){

shareWx = services[k];

}

}

if(shareWx == null){mui.toast('您没有安装微信'); return;}

var msg = {

content : title,

href : "http://www.52ocean.com"

};

if(type == 1){

// 发送的目标场景，WXSceneSession表示发送到会话

msg.extra = {scene:"WXSceneSession"}

}else{

// 发送的目标场景，WXSceneTimeline表示发送朋友圈

msg.extra = {scene:"WXSceneTimeline"}

}

shareWx.send(msg, function(){

mui.toast('分享成功');

}, function(){

mui.toast('您取消了分享');

});

}, function(){

mui.toast('获取分享服务失败');

});

}

尝试不安装微信查看提示、详情页面数据接口

第三方账号登录：sub4.html作为登录子页面，只有登录才能进入

1. 在sub3.html删除左滑动
2. 在index.html页面，

document.getElementById('nav4').addEventListener('tap',function () {

//通过本地存储获取用户id

var uid = plus.storage.getItem('uid');

if(uid == null){

//判断是否登录

wxLogin("changeSub(4);");

return false;

}

changeSub(4);

});

1. 在mui.min.js中，书写wxLogin（）函数

var wxLoginObj;

function wxLogin(goto){

//获取微信登录服务

plus.oauth.getServices(function(services){

// console.log(JSON.stringify(services));

for(var k in services){

if(services[k].id == "weixin"){

//获取微信登录对象

wxLoginObj = services[k];

}

};

if(wxLoginObj == null){

mui.toast('获取微信登录服务失败');

return false;

}

//进行微信登录

wxLoginObj.login(function(res){

// console.log(JSON.stringify(res));

//获取微信提供的用户信息

wxLoginObj.getUserInfo(function(res){

// console.log(JSON.stringify(res));

//获取微信提供的unionid

var unionid = res.target.authResult.unionid;

// console.log(unionid);

var nickname = res.target.userInfo.nickname;

var face = res.target.userInfo.headimgurl;

// console.log(nickname+"----"+faceurl);

mui.post('http://www.52ocean.com/wx.php',{

unionid:unionid,

nickname:nickname

},function(data){

// console.log(data);

// console.log(JSON.stringify(data));

//本地存储用户id、用户昵称、用户头像、随机数

plus.storage.setItem('uid',data.id+"");//注意这里要求id为string格式

plus.storage.setItem('uname',data.nickname);

plus.storage.setItem('uface',face);

plus.storage.setItem('urand',data.randnumber);

//提示用户登录成功

mui.toast('登录成功');

//跳转到登录子页面

if(typeof(goto) != 'undefined'){

eval(goto);

}

},'json'

);

},function(){

mui.toast('获取用户信息失败');

});

},function(){

mui.toast('微信登录失败');

return false;

});

},function(){

mui.toast('获取微信登录服务失败');

});

}

**注意事项**

1、开启auths 登录权限

2、设置微信登录所需的appid 和 AppSecret

3、微信开发者认证地址: <https://open.weixin.qq.com>

流程见《开通微信相关权限文件夹》

4、注意应用的包名一定要与开发者内申请的应用所填写的包名对应

5、注意openid 和 unionid的区别

设置用户头像和昵称

//设置用户头像和昵称：先加载h.js

h("#face").attr("src",plus.storage.getItem('uface'));

h('#name').html(plus.storage.getItem('uname'));

退出登录

//退出登录

function loginOut(){

//获取微信登录服务

plus.oauth.getServices(function(services){

// console.log(JSON.stringify(services));

for(var k in services){

if(services[k].id == "weixin"){

//获取微信登录对象

wxLoginObj = services[k];

}

};

if(wxLoginObj == null){

mui.toast('获取微信登录服务失败');

return false;

}

wxLoginObj.logout(function(){

plus.storage.removeItem('uid');//注意这里要求id为string格式

plus.storage.removeItem('uname');

plus.storage.removeItem('uface');

plus.storage.removeItem('urand');

mui.toast('成功退出');

var \_index = plus.webview.getLaunchWebview();

\_index.evalJS('changeSub(1);h("#navFooter").find("a").removeClass("mui-active");h("#nav1").addClass("mui-active");');

},function(){});

},function(){}

);

}