前言

虽然现在javascript模块还没有官方规范，但目前通行的javascript模块规范有两种，即commonJs和AMD（asynchronous module define）。

Node.js就是参照commonJs规范的，从而可以将javascript语言用于服务器端，实现模块化编程。CommonJs中有个全局性方法“require()”,用于模块的加载。

在服务器端，因为模块存储在本地硬盘，所以模块加载的时间=硬盘读取时间。

但是在客户端，比如浏览器，要从服务器端加载模块时，为了缩短加载时间，常常需要异步加载。所以符合AMD规范的javascript库require.js就产生了，除此之外还有curl.js和符合cmd规范的seaJs。在此篇文档中，重点讲述requireJs，若对其他的javascript库感兴趣，请上网百度。

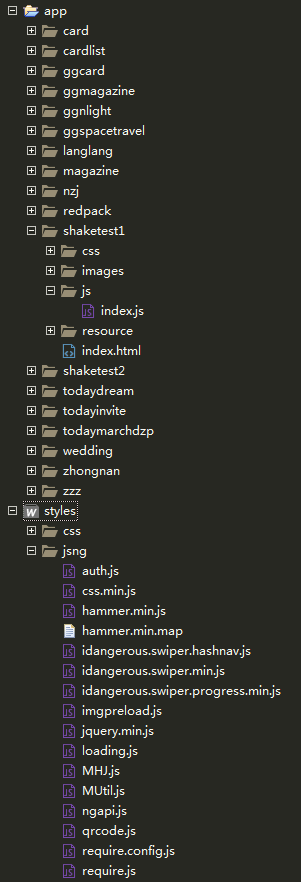
1. require.js的使用

RequireJs以一个相对于baseUrl的地址来加载所有代码。页面顶层<script>标签含有一个特殊的属性data-main，require.js使用它来启动脚本加载过程，而baseUrl一般设置到与该属性相一致的目录。

RequireJS默认假定所有的依赖资源都是js脚本，因此无需在module ID上再加".js"后缀，RequireJS在进行module ID到path的解析时会自动补上后缀。你可以通过[paths config](http://makingmobile.org/docs/tools/requirejs-api-zh/" \l "config-paths)设置一组脚本，这些有助于我们在使用脚本时码更少的字。

以国广摇一摇中的配置为例：

文件目录：

  
图1

由图1 可知，index.html的路径名pathname是“app/shaketest1/index.html”



index.html



图2

通过data-main属性，加载第一个应用模块index.js。

index.js

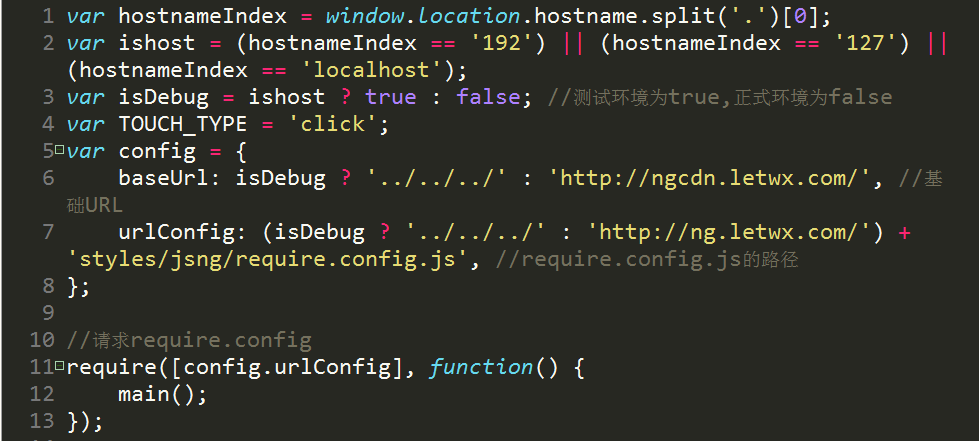


图3

Require.config是一个js对象，有baseUrl和paths等属性。

在图3中定义了config对象，并设置beseUrl和urlConfig的属性值。

如果是测试环境，baseUrl为本机的ip地址，我本机的ip地址和端口号为“192.168.1.4:2014” 

如果是正式环境，baseUrl为“http://ngcdn.letwx.com/”

同时还require了“require.config.js”这个模块，在require.config.js这个文件里面，配置了一些会经常用到的js模块（如图4）。

require.config.js

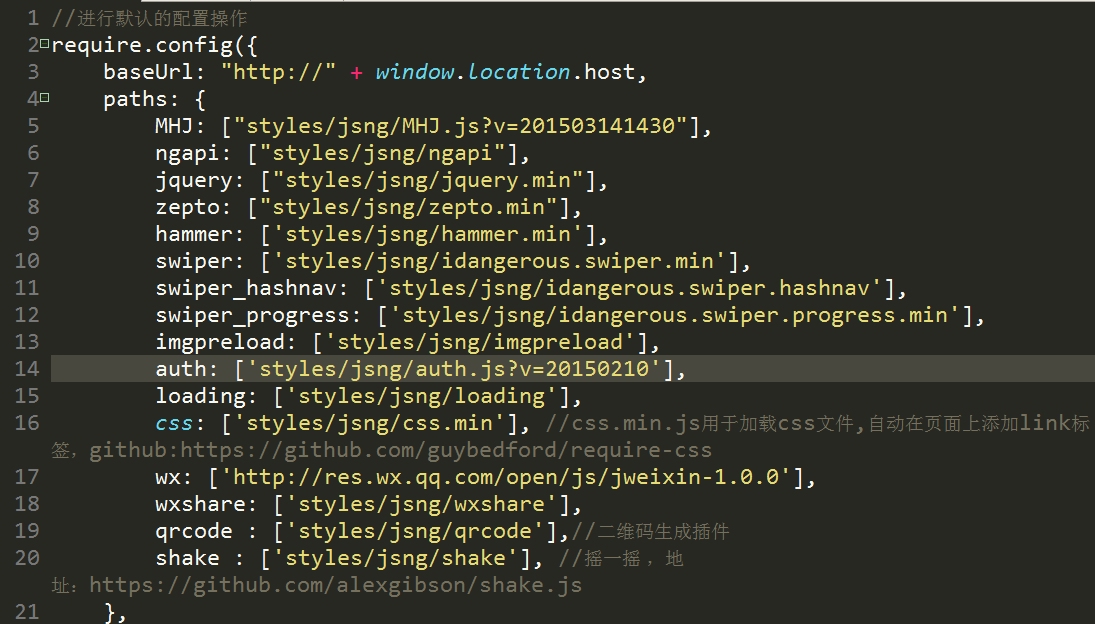


图4



图5

paths是基于baseUrl的，我们可以查看浏览器中加载的代码（如图6）

shim：为那些没有使用define()来声明依赖关系、设置模块的"浏览器全局变量注入"型脚本做依赖和导出配置。



图6

模板加载过程：

首先加载了require.min.js，然后通过data-main入口点加载index.js。index.js类似于c语言中的main函数，是个入口。通过index.js加载了require.config.js（异步加载）。又通过require.config.js中的配置异步加载其他的模块,在后面我会较为详细地说明各个模块的功能。

1. 整体流程
2. 引入服务config.js

在上面的实例中已经详细讲述如何引入及配置，在此我就不赘述了。

1. 检测用户身份进行auth认证

Index.js

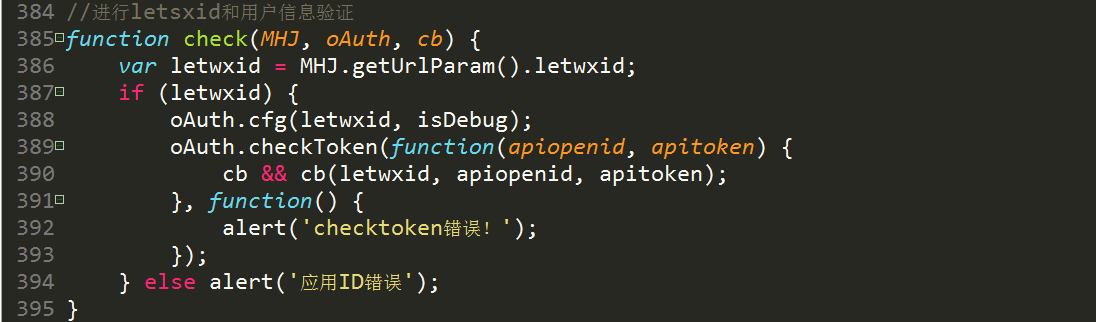


图7

定义check()函数 ，用于用户认证。

注意：在浏览器中进行测试时，在pathname后面加上“？letwxid=XX”，否则会报错“应用ID错误”。

调用了MHJ模块中的getUrlParam()方法，auth模块中的cfg()方法和checkToken()方法。

MHJ.js

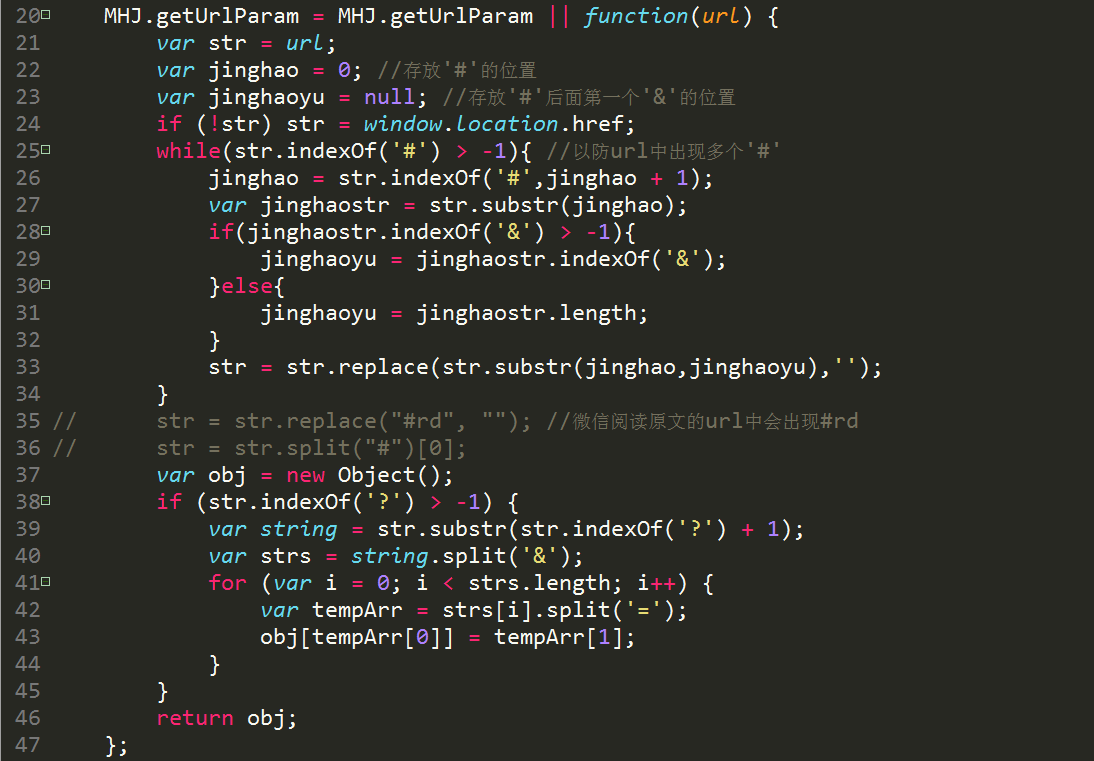


图8

图8中，getUrlParam()方法返回参数对象。可直接通过MHJ.getUrlParam().letwxid获得letwxid的值。

Auth.js

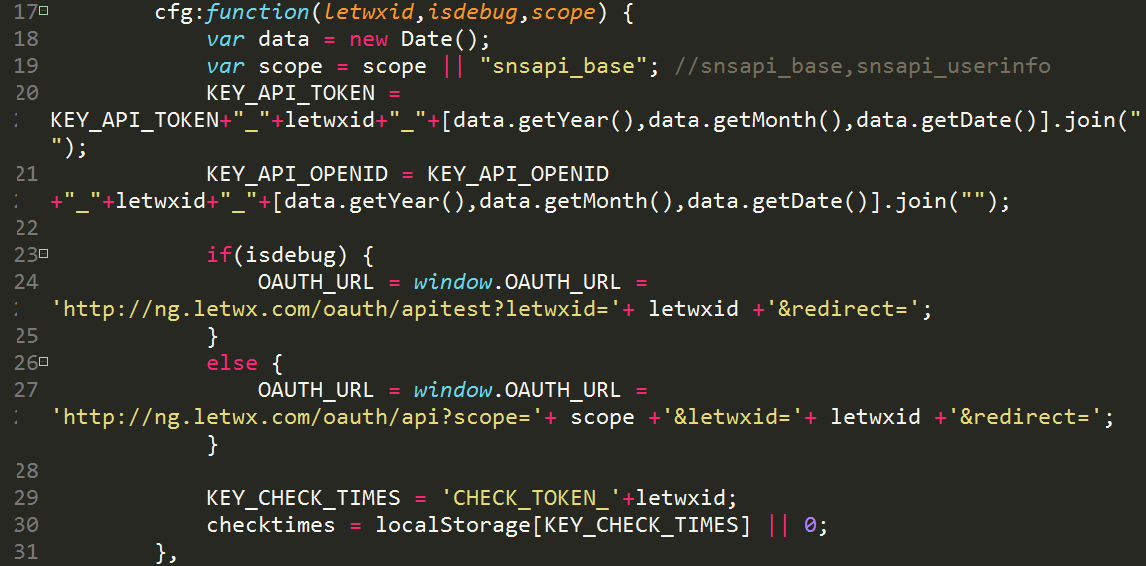


图9

oAuth.cfg()方法用来设置OAUTH\_URL用户地址和一些其他的变量设置，比如KEY\_API\_TOKEN。

Auth.js

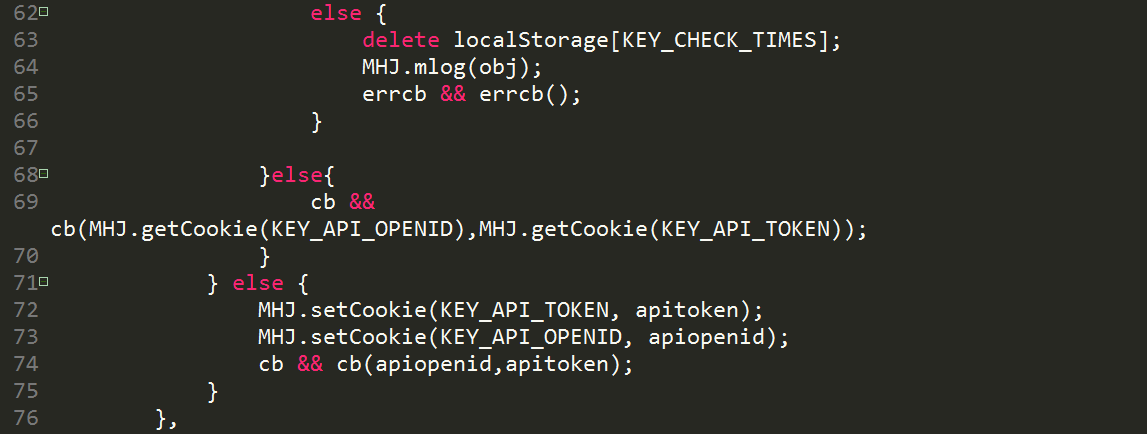
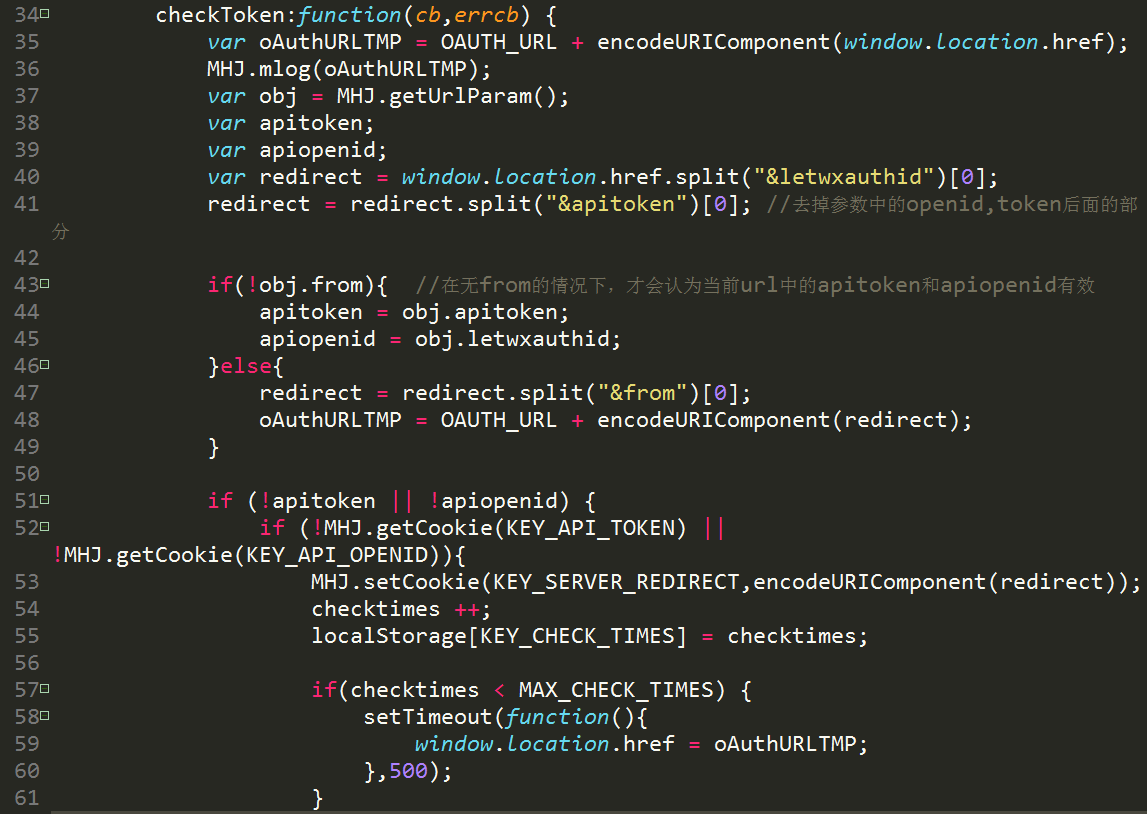


图10

图10中是checkToken方法，进行apiopenid和apitoken验证。

MHJ.js

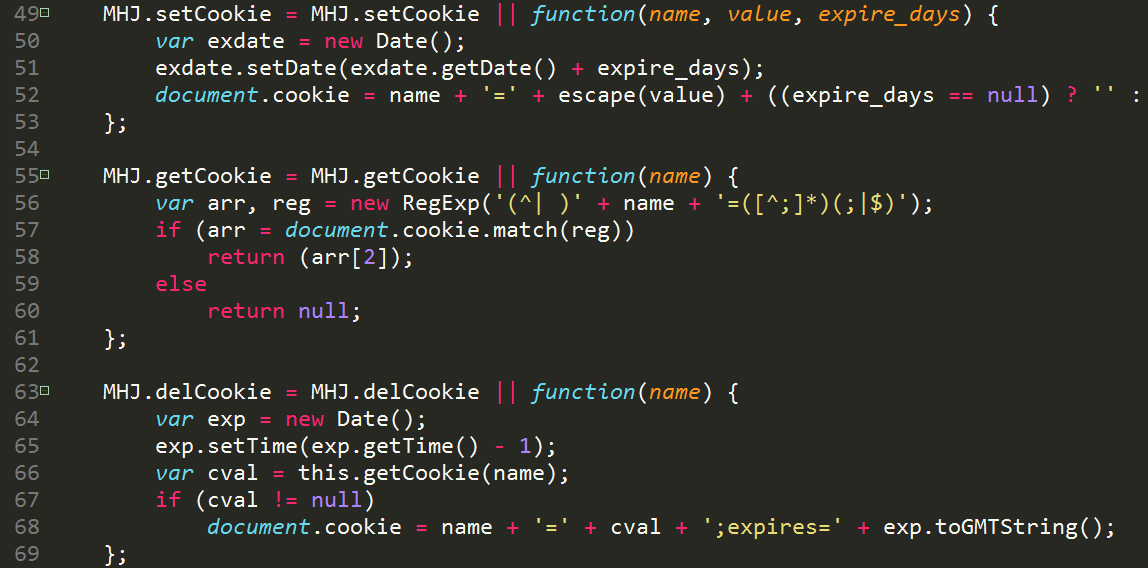


图11

setCookie()：存入cookie的名称、值和有效天数。

getCookie()：通过名称找到对应的值。

delCookie()：通过设置失效时间来清除cookie。

（若编写的页面只是一般用于浏览的页面，不需要输入用户信息，就无需进行用户验证，此步骤可以省略。）

1. 业务逻辑

完成以上两个步骤之后，就可以真正地来写页面的业务逻辑了。

从定义图3中的main()函数开始，

Index.js

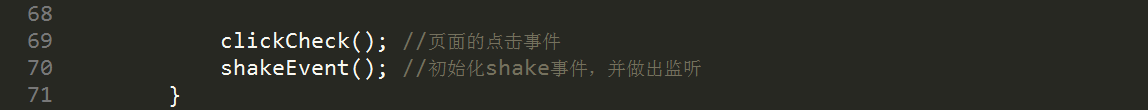
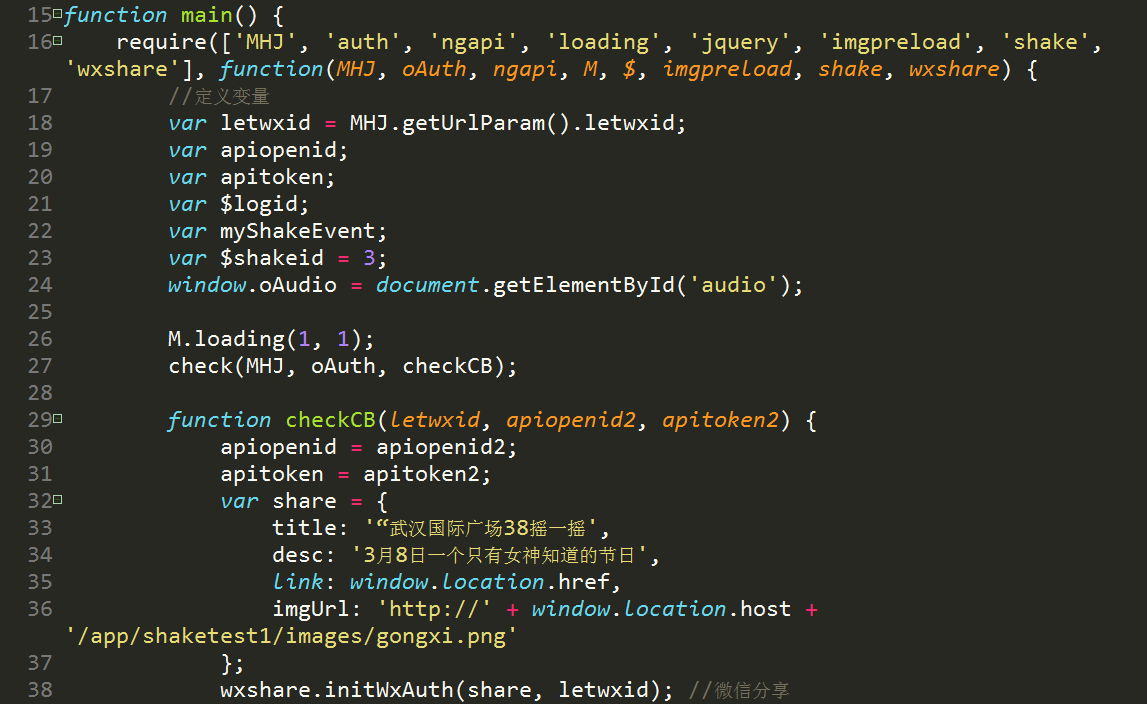


图12

在图12中，require异步加载需要用到的模块，MHJ和auth主要用于用户认证，ngapi用于调用后台数据，loading是页面加载时的动画，jquery是js库，imgpreload用于图片加载，shake用于手机摇一摇，wxshare用于微信分享。

Require中function的参数顺序要和模块数组中的顺序一致，参数只是模块的一种表示符号，可以随意取值，但要注意简单易懂，大众化。

首先页面加载动画M.loading()，接着调用图7中的check()方法，定义回调函数checkCB()，在checkCB()中进行微信分享wxshare.initWxAuth()和图片预加载imgpreload.load()。当图片预加载完了之后，隐藏页面加载动画，显示主页面，然后通过ngapi调用后台数据和编写页面上的其他业务逻辑，比如touch事件、shake事件等等。

Loading.js

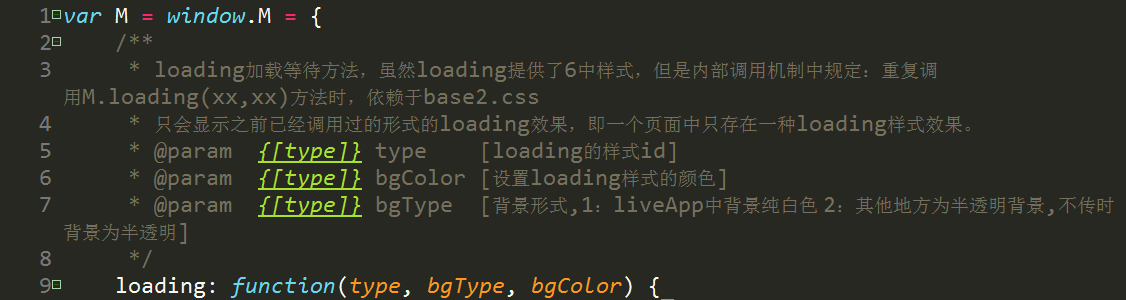


图13

Loading中的参数解释：

type：loading的样式id，不同的id值代表不同的加载动画，如表1。

bgType：背景类型，取值见表2。

bgColor：loading样式的颜色，可以取‘#0f0’或‘rgab(0,0,0,1)’。

表1

|  |  |
| --- | --- |
| type | animation |
| 1 | 翻转 |
| 2 | 竖条伸缩 |
| 3 | 两个圈相互切换 |
| 4 | 两个正方形相互切换 |
| 5 | 三个点相互切换 |
| 6 | 圆圈依次加载 |
| default | 元素旋转 |

表2

|  |  |
| --- | --- |
| bgType | color |
| 1 | 白色（white） |
| 2 | 半透明（rgba(0,0,0,0.6)） |

ngapi.js

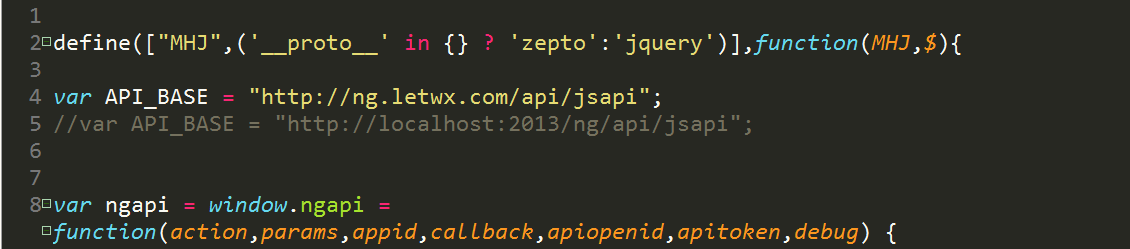


图14

ngapi中参数说明：

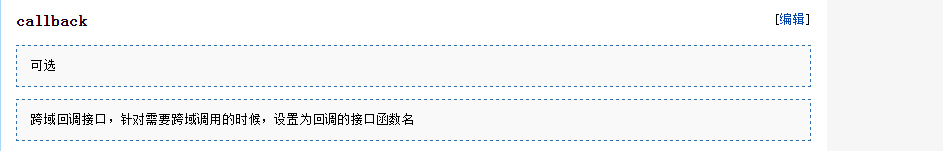


图15

具体的详见项目的接口文档。

例如摇一摇中的获取摇一摇活动信息（如图16）：

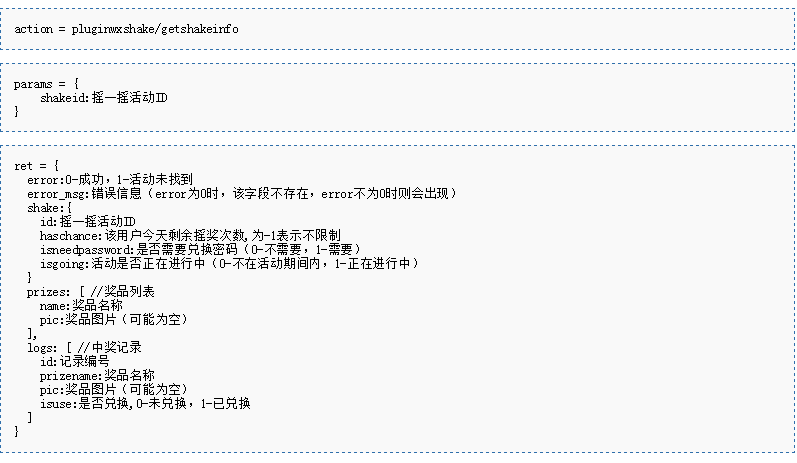


图16

具体使用请看图12.

1. 注意事项
2. 查错时可能会用到JSON.stringify(data)方法。Data是ngapi中返回的数据，是一个对象，所以直接打印出来的话是一个object，要想查看里面的内容，需要将其转换成字符串。
3. 在需要改变页面结构时，最好使用MHJ.tmpl()方法。使用这种方法的好处在于，页面结构看起来清晰有条理，修改方便，灵活性强。

使用示例：

Js部分：



图17

图17中语句的涵义：将data.logs中的数据绑定在id=”pListTMP”的textarea中，并将绑定后的textarea中的内容放置到id=”pList”的模块中。取模板的id名称时，最好在后面加上“TMP”，即id=”xxTMP”。

Html部分：



图18

如图18所示，在“<%%>”中可以进行逻辑处理，“<%=%>”是赋值的形式。这段代码的意思是循环输出li，并设置标签元素中的属性值，此时图18中的data相当于图17中的data.logs。data中数据的涵义可以详见图16，data就是摇一摇活动信息返回的json对象。

因为使用tmpl()方法时，模板中的内容是异步加载的，所以在模板内容中进行的逻辑操作和tmpl()的调用在一个function里面，否则写的逻辑会无效。

1. 为保证js的可读性和灵活性，最好用多个function来编写，即模块化，每个模块之间链式相连。
2. 在编写html页面时，body的firstchild要是“<section id=”content”></section>”然后把页面结构都放在section里面。这样写是和loading.js有关，在调用loading()时，创建了一个class名为“m-pop”的div，weight和height均为100%，z-index为999这样就遮住了原本的页面。等到图片加载完成后，“M.loadingHide();$(#content).show();”隐藏loading动画，显示content。

## 不要使用If...else if...else 语句，最好使用switch语句。

## Define的问题。

## 因为jQuery从1.7后开始支持AMD规范，即如果jQuery作为一个AMD模块运行时，它的模块名是“jquery”。

## jQuery中的支持AMD代码如下

## 

## 图19

## jQuery最终向外暴露的是全局的jQuery和 $。如下

## 

## 图20

但有些js模块是没有支持AMD规范的，这就需要我们为其define。

例如

auth.Js

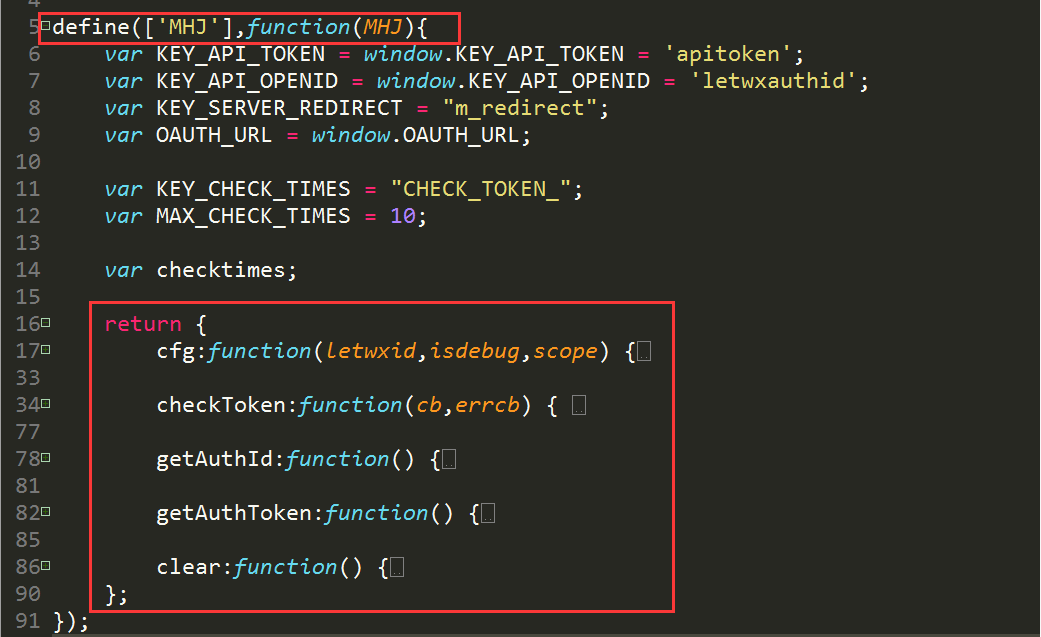


图21

依赖MHJ模块，返回了5个function。

MHJ.js



图22

没有依赖任何模块，直接返回M这个对象。