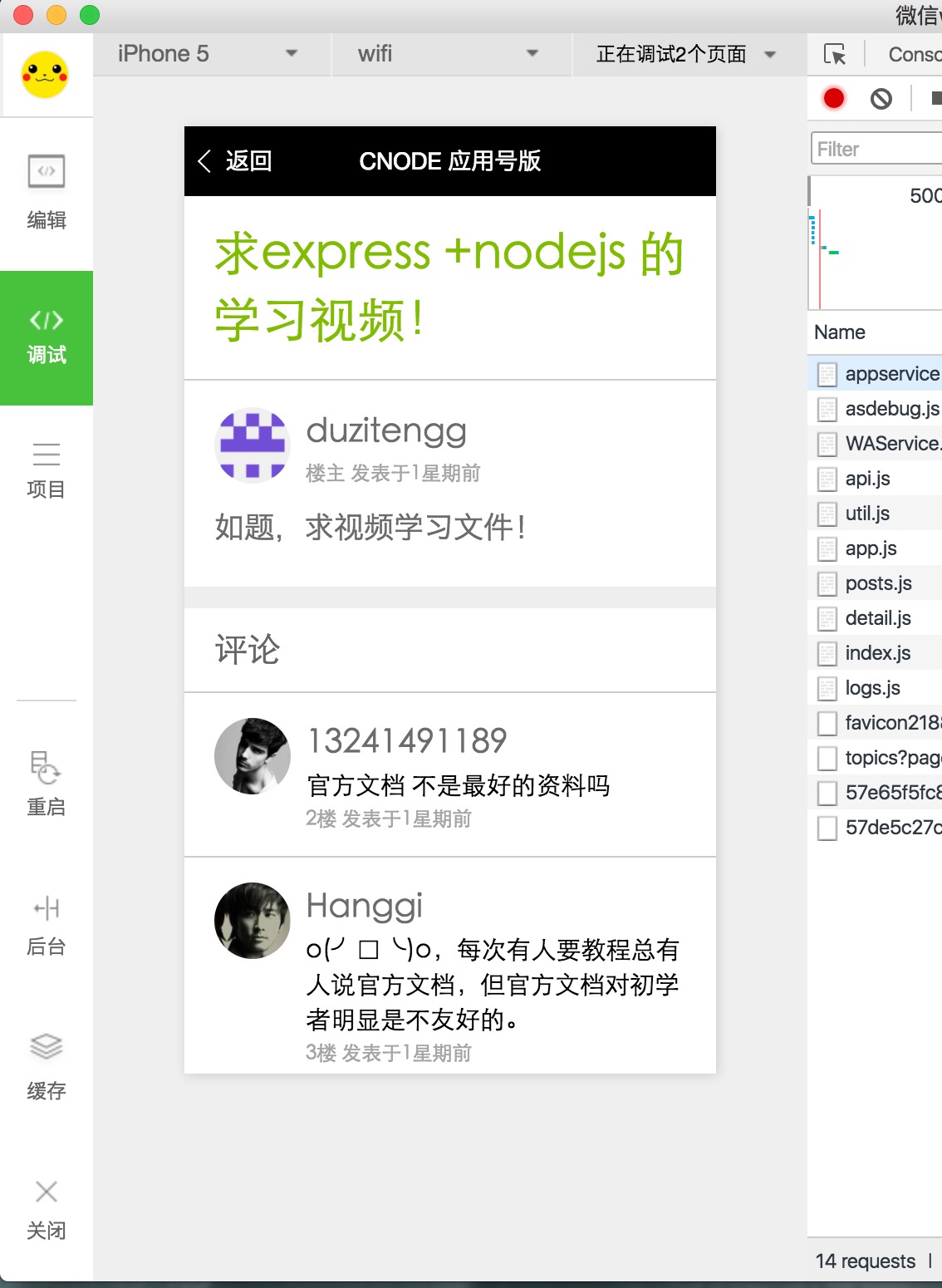
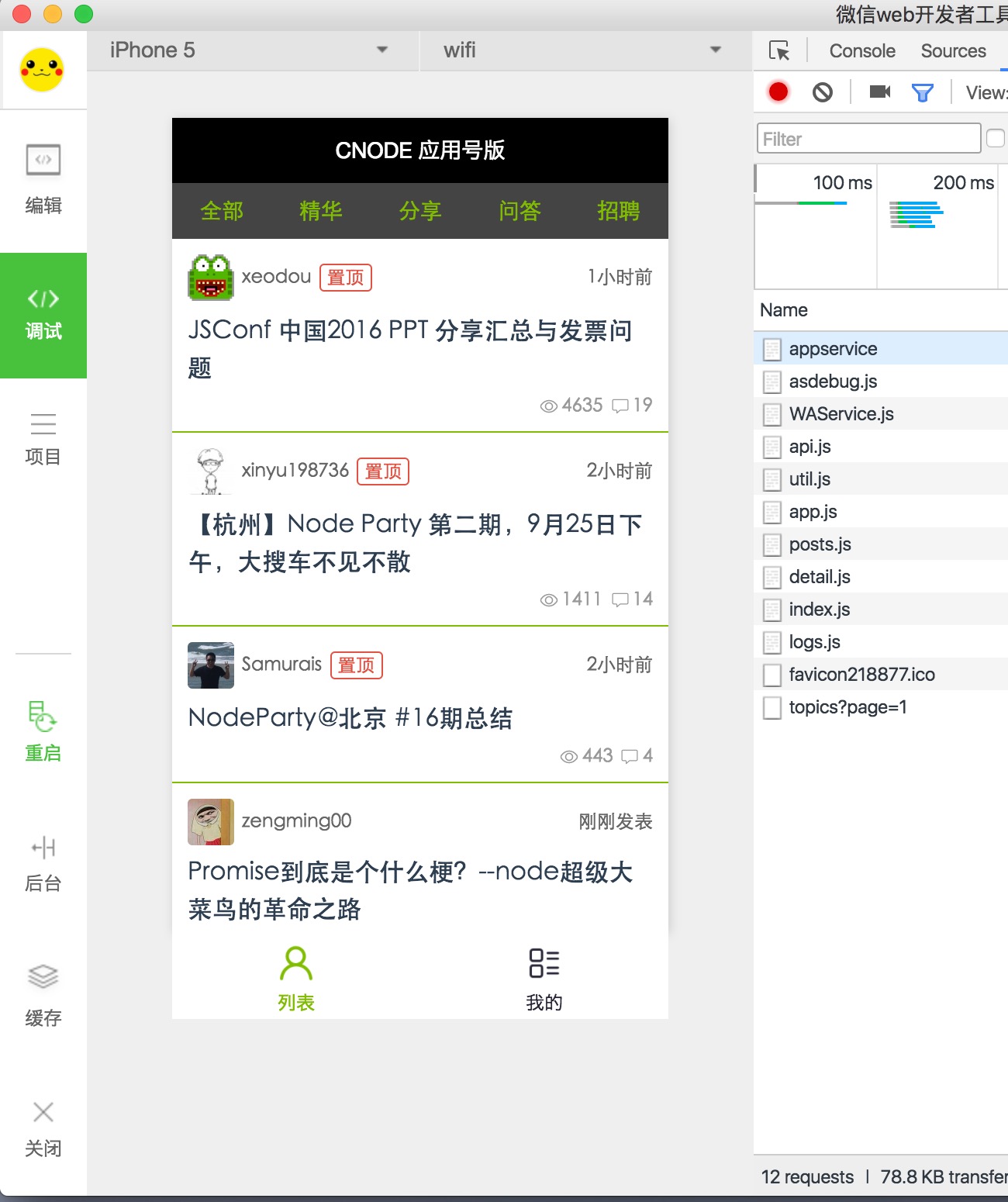
**微信应用号入门实践之 cnode 社区版**

首先感谢 cnode 社区提供的 api ，本次实现了简单的 cnode 社区应用号制作。 实现了数据的读取、展示， 实现了简单的布局， 实现了下一页功能。



下面就说说我做这个的过程，不足之处，请多多指教，只愿为进步。

## 1.创建项目

首先，在官网下载工具，[下载地址](https://mp.weixin.qq.com/debug/wxadoc/dev/devtools/download.html?t=1474887501214) 我的是选择 mac 版本 0.9.092300 。

然后跟着官方的[简版教程](https://mp.weixin.qq.com/debug/wxadoc/dev/?t=1474887501301) 创建一个项目。

#### 注：现在官方的工具支持无 appid 创建项目。

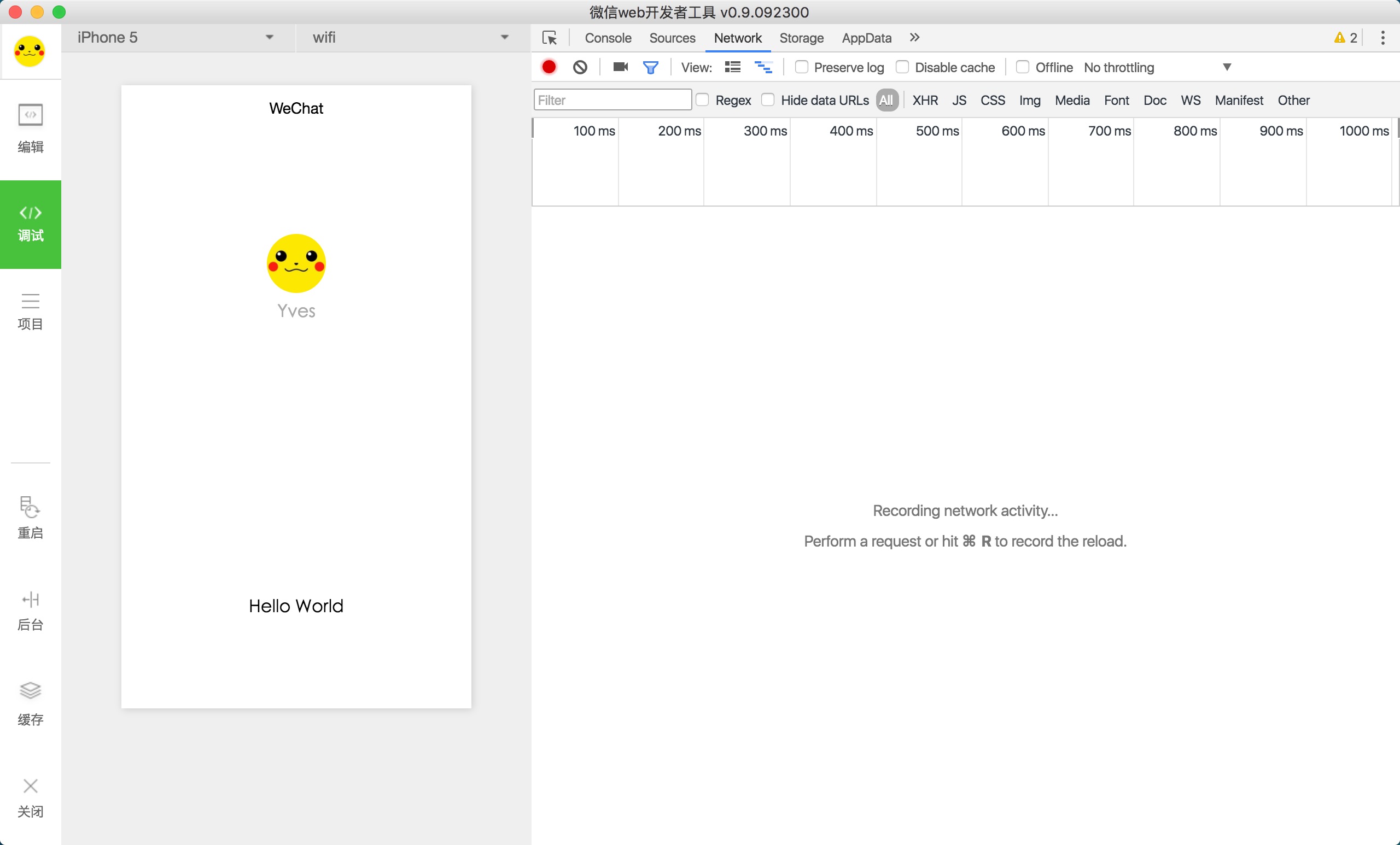
##### 1.打开开发者工具，选择“添加项目”



##### 2.选择无 appid ，填写地址，创建项目



##### 3.创建成功，看到默认的 Demo 项目页面



## 2.配置

默认的项目里已经没有关于 tabBar 的配置信息，所以为了学习，我把这个配置进行了修改。

首先关于配置的说明同样来自于[官方](https://mp.weixin.qq.com/debug/wxadoc/dev/framework/config.html?t=1474887501556)

注意：官方的代码有些字段是不一样的，小心被坑。

{

"pages":[

"pages/index/index",

"pages/logs/logs"

],

"window":{

"backgroundTextStyle":"light",

"navigationBarBackgroundColor": "#fff",

"navigationBarTitleText": "WeChat",

"navigationBarTextStyle":"black"

},

"tabBar":{

"list": [{

"pagePath": "pages/index/index",

"text": "首页"

}, {

"pagePath": "pages/logs/logs",

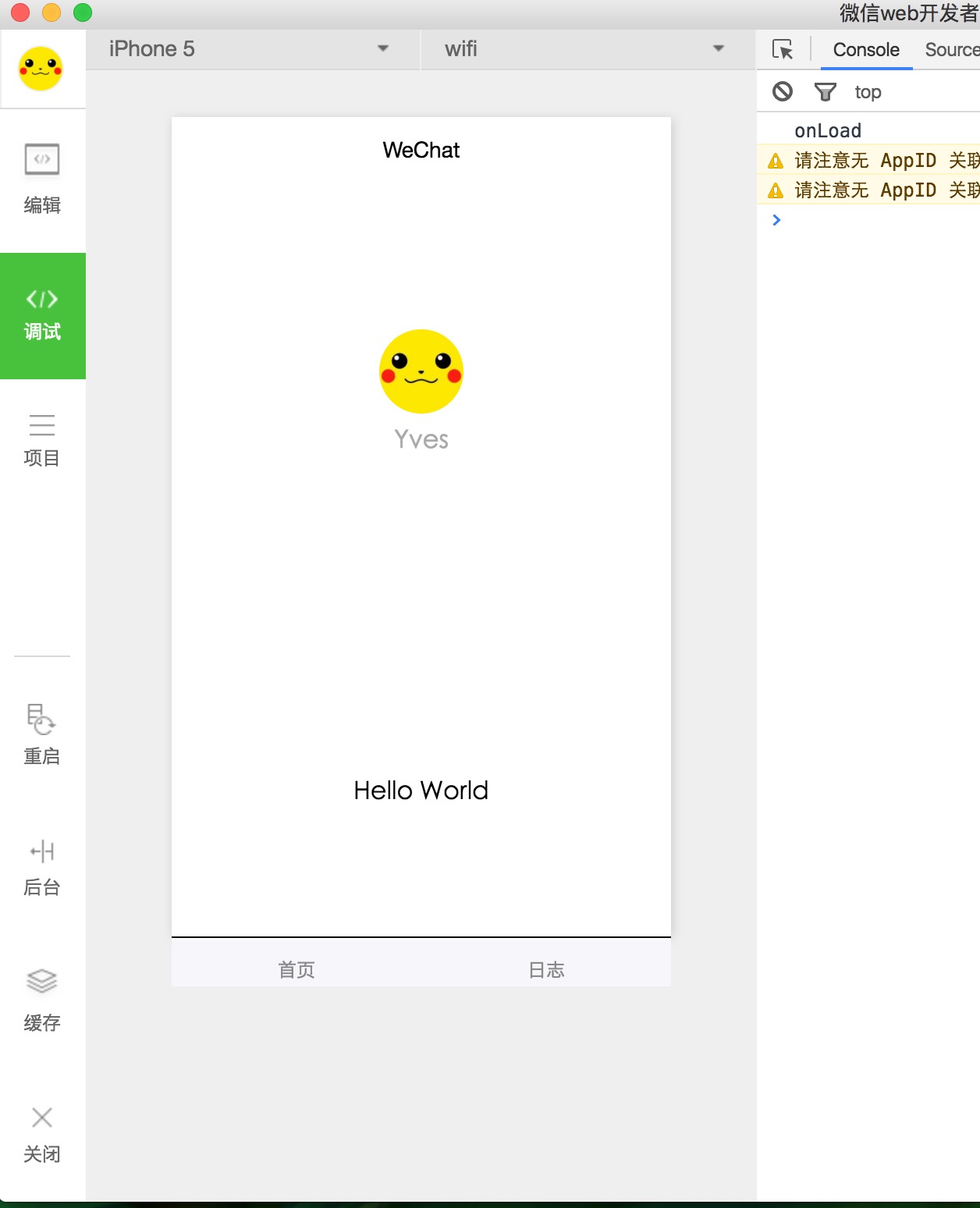
"text": "日志"

}]

}

}

增加了 tabBar, 查看调试

看界面是如此的简陋，为此针对 tabBar 参考官方说明进行了简单的美化。

"tabBar":{

"color":"#272636",

"selectedColor":"#80bd01",

"backgroundColor":"#fff",

"borderStyle":"white",

"list":[{

"pagePath":"pages/index/index",

"text":"首页",

"iconPath":"images/tabBar/my.png",

"selectedIconPath":"images/tabBar/my\_hl.png"

},{

"pagePath":"pages/index/index",

"text":"我的",

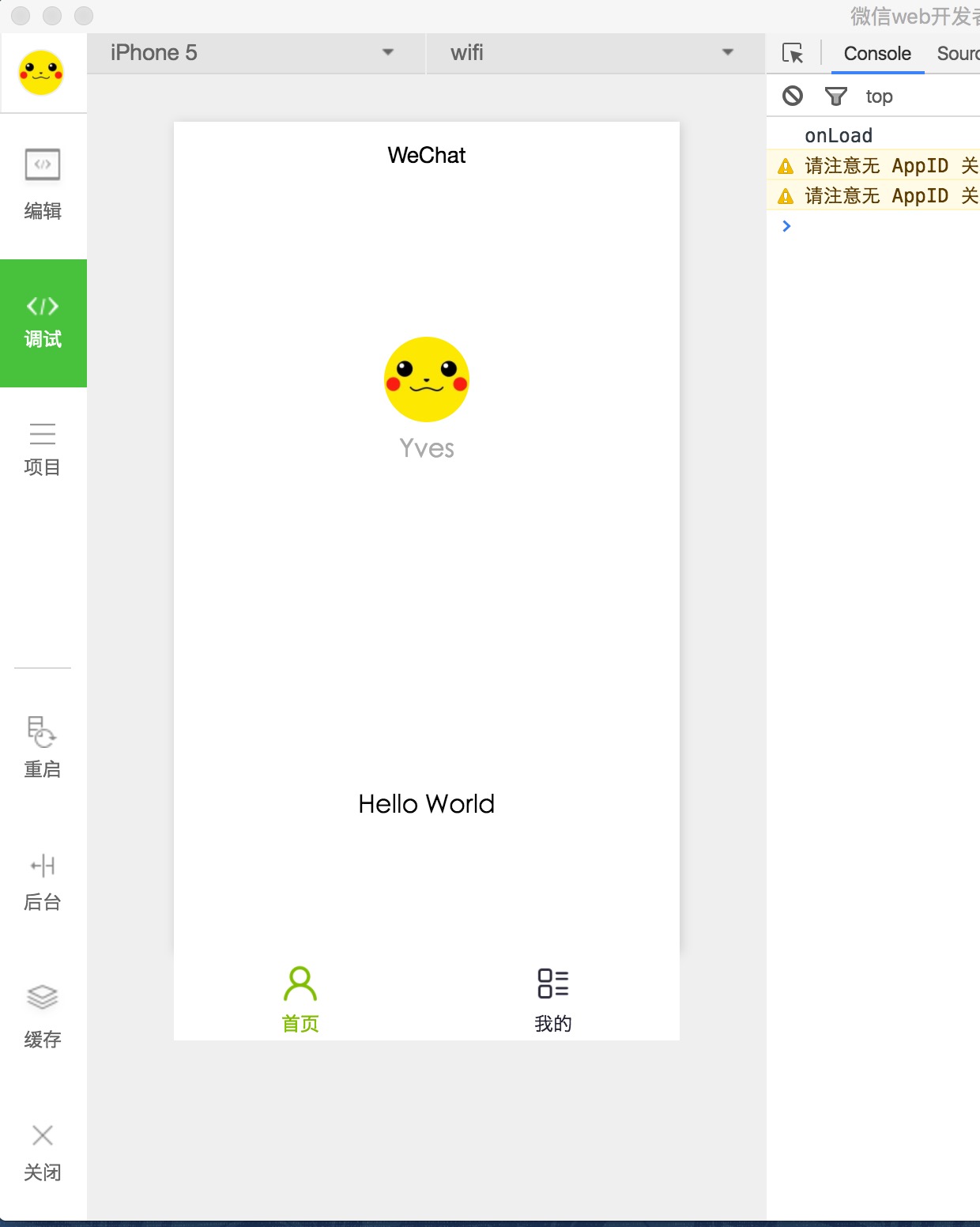
"iconPath":"images/tabBar/list.png",

"selectedIconPath":"images/tabBar/list\_hl.png"

}]

}

效果如图

最后根据文档，对默认页面的窗口表现进行了修改

"window":{

"backgroundTextStyle":"black",

"backgroundColor":"#fff",

"navigationBarBackgroundColor":"#000",

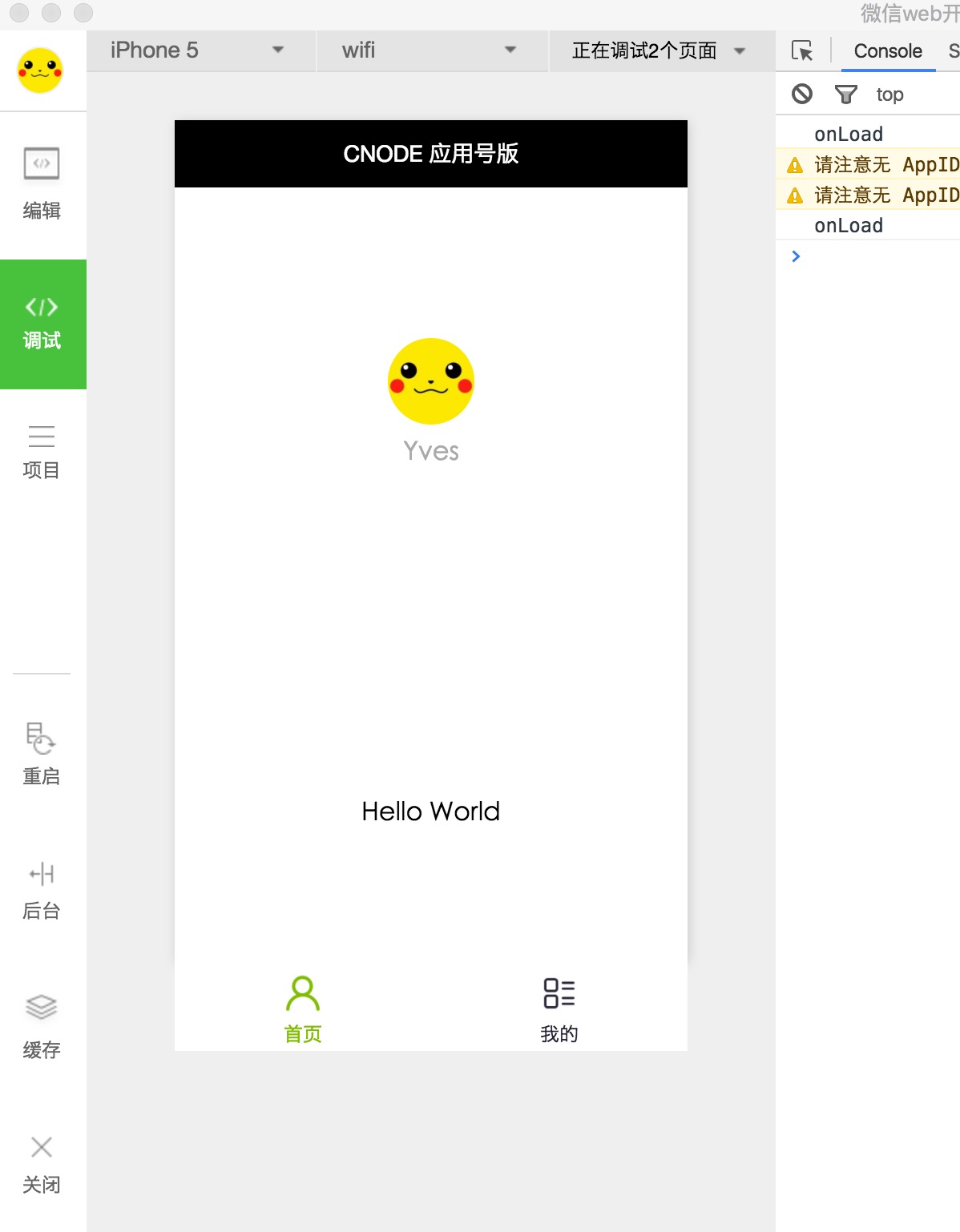
"navigationBarTitleText":"CNODE 应用号版",

"navigationBarTextStyle":"white",

"enablePullDownRefresh":"true"

},

效果如图

整体配置文件为

{

"pages":[

"pages/index/index",

"pages/logs/logs"

],

"window":{

"backgroundTextStyle":"black",

"backgroundColor":"#fff",

"navigationBarBackgroundColor":"#000",

"navigationBarTitleText":"CNODE 应用号版",

"navigationBarTextStyle":"white",

"enablePullDownRefresh":"true"

},

"tabBar":{

"color":"#272636",

"selectedColor":"#80bd01",

"backgroundColor":"#fff",

"borderStyle":"white",

"list":[{

"pagePath":"pages/index/index",

"text":"首页",

"iconPath":"images/tabBar/my.png",

"selectedIconPath":"images/tabBar/my\_hl.png"

},{

"pagePath":"pages/index/index",

"text":"我的",

"iconPath":"images/tabBar/list.png",

"selectedIconPath":"images/tabBar/list\_hl.png"

}]

}

}

## 3.制作首页列表

根据官方要求，我在 pages 文件夹内创建了 topics 文件夹，并创建了对应了 topics.js 、 topics.wxml 、 topics.wxss 三个文件。

### 1.注册页面

首先在配置文件里注册这个 topics,

"pages":[

"pages/topics/topics",

"pages/index/index",

"pages/logs/logs"

],

并且制定 tabBar 点击跳到对应的 topics 页面

"tabBar":{

"color":"#272636",

"selectedColor":"#80bd01",

"backgroundColor":"#fff",

"borderStyle":"white",

"list":[{

"pagePath":"pages/topics/topics",

"text":"首页",

"iconPath":"images/tabBar/my.png",

"selectedIconPath":"images/tabBar/my\_hl.png"

},{

"pagePath":"pages/index/index",

"text":"我的",

"iconPath":"images/tabBar/list.png",

"selectedIconPath":"images/tabBar/list\_hl.png"

}]

}

##### 注意：我发现注册页面的顺序会影响到默认显示 tabBar 的顺序，所以我把"pages/topics/topics"放到了"pages/index/index"的前面

然后编写 topics.js

Page({

data: {

title: '首页列表'

},

onLoad: function () {

console.log('onLoad by topics');

}

});

以及 topics.wxml 文件

<view class="topics-main">

测试首页列表界面</view>

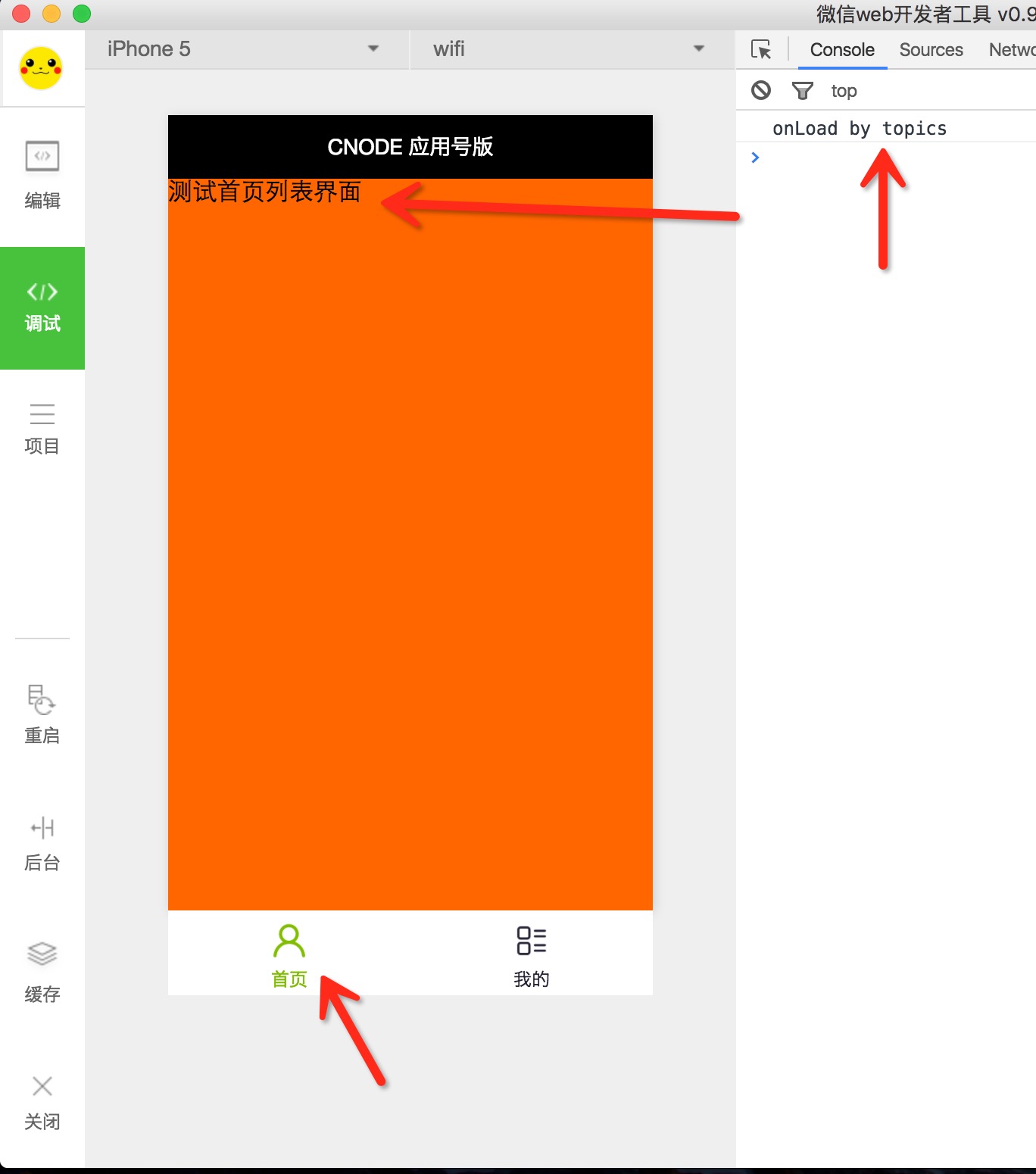
和 topics.wxss 文件

.topics-main {

background: #f60;

height: 100%;

}

最后效果如图 

### 2.创建请求

根据文档[请求数据](https://mp.weixin.qq.com/debug/wxadoc/dev/api/network-request.html?t=1474887499445)，在 util 文件夹内创建一个 api.js 文件，专门进行数据请求处理。

'use strict';var HOST\_URI = 'https://cnodejs.org/api/v1';

var GET\_TOPICS = '/topics';var GET\_TOPIC\_BY\_ID = '/topic/';

function obj2uri (obj) {

return Object.keys(obj).map(function (k) {

return encodeURIComponent(k) + '=' + encodeURIComponent(obj[k]);

}).join('&');

}

module.exports = {

// 获取列表数据

getTopics: function (obj) {

return HOST\_URI + GET\_TOPICS + '?' + obj2uri(obj);

},

// 获取内容页数据

getTopicByID: function (id, obj) {

return HOST\_URI + GET\_TOPIC\_BY\_ID + id + '?' + obj2uri(obj);

}

};

修改 topics.js

var Api = require('../../utils/api.js');

Page({

data: {

title: '首页列表'

},

onLoad: function () {

console.log('onLoad by topics');

this.fetchData();// 获取数据

},

fetchData: function (data) {

// 处理参数

if (!data) data = {};

if (!data.page) data.page = 1;

wx.request({

url: Api.getTopics(data),

success: function (res) {

console.log(res);

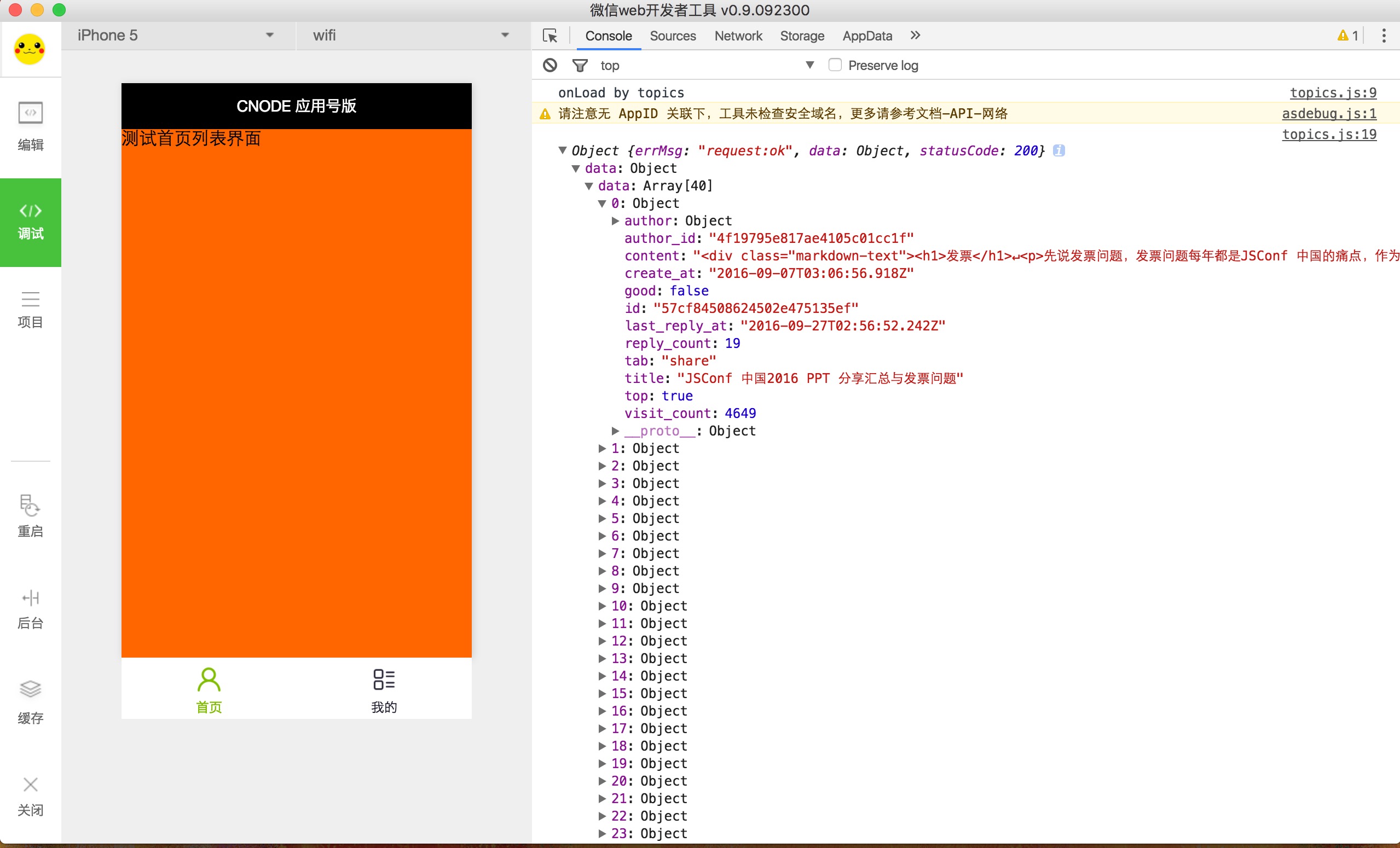
}

});

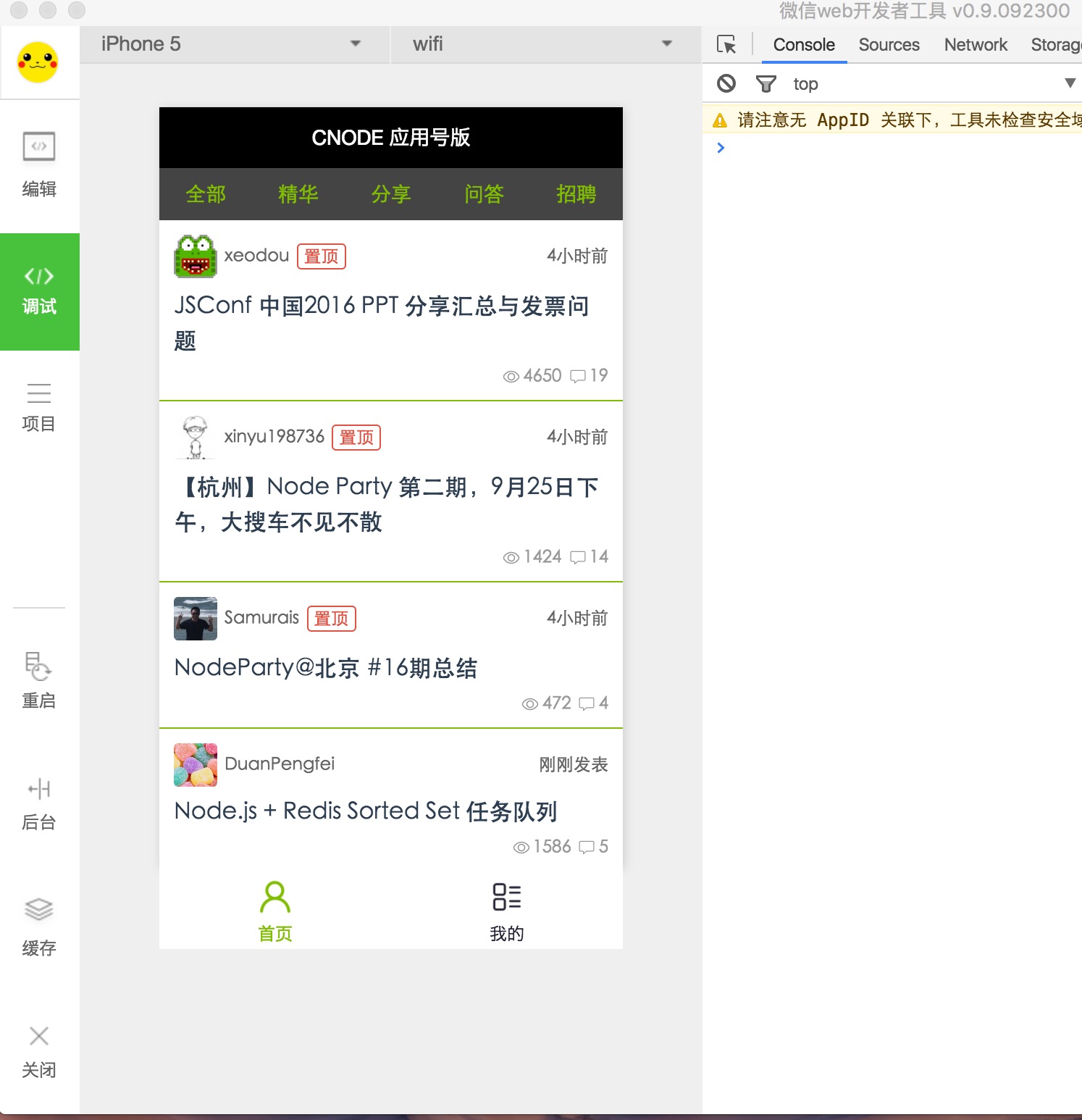
}

});

效果如图

成功拿到了数据。

### 3.完善首页列表

拿到了数据，也能修改界面，那么就直接完善这个首页把代码就不放了，直接上图 我认为这里值得说的大概只有 loading 、循环、传参、下一页和页面跳转了。

#### 1.loading

<loading hidden="{{hidden}}">

加载中...

</loading>

在 topics.wxml 中写官方提供的 loading 组件，根据在 topics.js 中对 hidden 值的修改，来触发 loading 。

#### 2.循环数据，展示列表

文档提供了[列表渲染](https://mp.weixin.qq.com/debug/wxadoc/dev/framework/view/wxml/list.html?t=1474887501556)

通过wx:for实现列表的渲染。

##### 注意: 默认数组的当前项的下标变量名默认为 index ，数组当前项的变量名默认为 item 。

<block wx:for="{{postsList}}">

<view class="posts-item" index="{{index}}" id="{{item.id}}" catchtap="redictDetail">

<view class="author">

<image class="author-avatar" src="{{item.author.avatar\_url}}"></image>

<view class="author-name">{{item.author.loginname}}</view>

<view class="posts-tag hot" wx:if="{{item.top === true}}">置顶</view>

<view class="posts-tag" wx:if="{{item.good === true}}">精华</view>

<view class="posts-last-reply">{{item.last\_reply\_at}}</view>

</view>

<view class="posts-title">{{item.title}}</view>

<view class="bar-info">

<view class="bar-info-item">

<image class="bar-info-item-icon" src="/images/icon/reply.png"></image>

<view class="bar-info-item-number">{{item.reply\_count}}</view>

</view>

<view class="bar-info-item">

<image class="bar-info-item-icon" src="/images/icon/visit.png"></image>

<view class="bar-info-item-number">{{item.visit\_count}}</view>

</view>

</view>

</view></block>

附上一个没有样式的列表展现 

#### 3.传参，实现 tab 切换

根据 cnode 的 api 可以知道通过 tab 不同的值，获得到不同标签下的内容列表。

所以 在页面的最上面 tab 栏中

<view class="top-bar">

<view class="top-bar-item" id="all" catchtap="onTapTag">全部</view>

<view class="top-bar-item" id="good" catchtap="onTapTag">精华</view>

<view class="top-bar-item" id="share" catchtap="onTapTag">分享</view>

<view class="top-bar-item" id="ask" catchtap="onTapTag">问答</view>

<view class="top-bar-item" id="job" catchtap="onTapTag">招聘</view>

</view>

将 id 进行定义，通过获取 id 拿到对应的 tab 类型。

其中catchtap是事件绑定。

bind 事件绑定不会阻止冒泡事件向上冒泡， catch 事件绑定可以阻止冒泡事件向上冒泡。

在 topics.js 获取

onTapTag: function (e) {

var self = this;

var tab = e.currentTarget.id;

// 这里就能获取到不同的 tab 值了

self.setData({

tab: tab

});

if (tab !== 'all') {

this.fetchData({tab: tab});

} else {

this.fetchData();

}

},

#### 4.下一页的实现

根据文档，组件的视图容器中有[scroll-view](https://mp.weixin.qq.com/debug/wxadoc/dev/component/scroll-view.html?t=1474887496374)这个可滚动视图区域。

##### 注意：使用竖向滚动时，需要给<scroll-view/>一个固定高度。

<scroll-view class="posts-list" style="height:100%" scroll-y="true" bindscrolltolower="lower">

<block wx:for="{{postsList}}">

...

</block></scroll-view>

topics.js 文件

lower: function (e) {

var self = this;

// 修改当前页码

self.setData({

page: self.data.page + 1

});

// 判断当前页的 tab 值 进行请求数据

if (self.data.tab !== 'all') {

this.fetchData({tab: self.data.tab, page: self.data.page});

} else {

this.fetchData({page: self.data.page});

}

}

#### 5.跳页的实现

在posts-item中已经进行了事件绑定。利用wx.navigateTo实现页面的跳转。

##### 注意：一个应用同时只能打开 5 个页面，当已经打开了 5 个页面之后， wx.navigateTo 不能正常打开新页面。

redictDetail: function (e) {

console.log('我要看详情');

var id = e.currentTarget.id,

url = '../detail/detail?id=' + id;

// 这里的 detail 是需要创建对应的文件，以及页面注册的

wx.navigateTo({

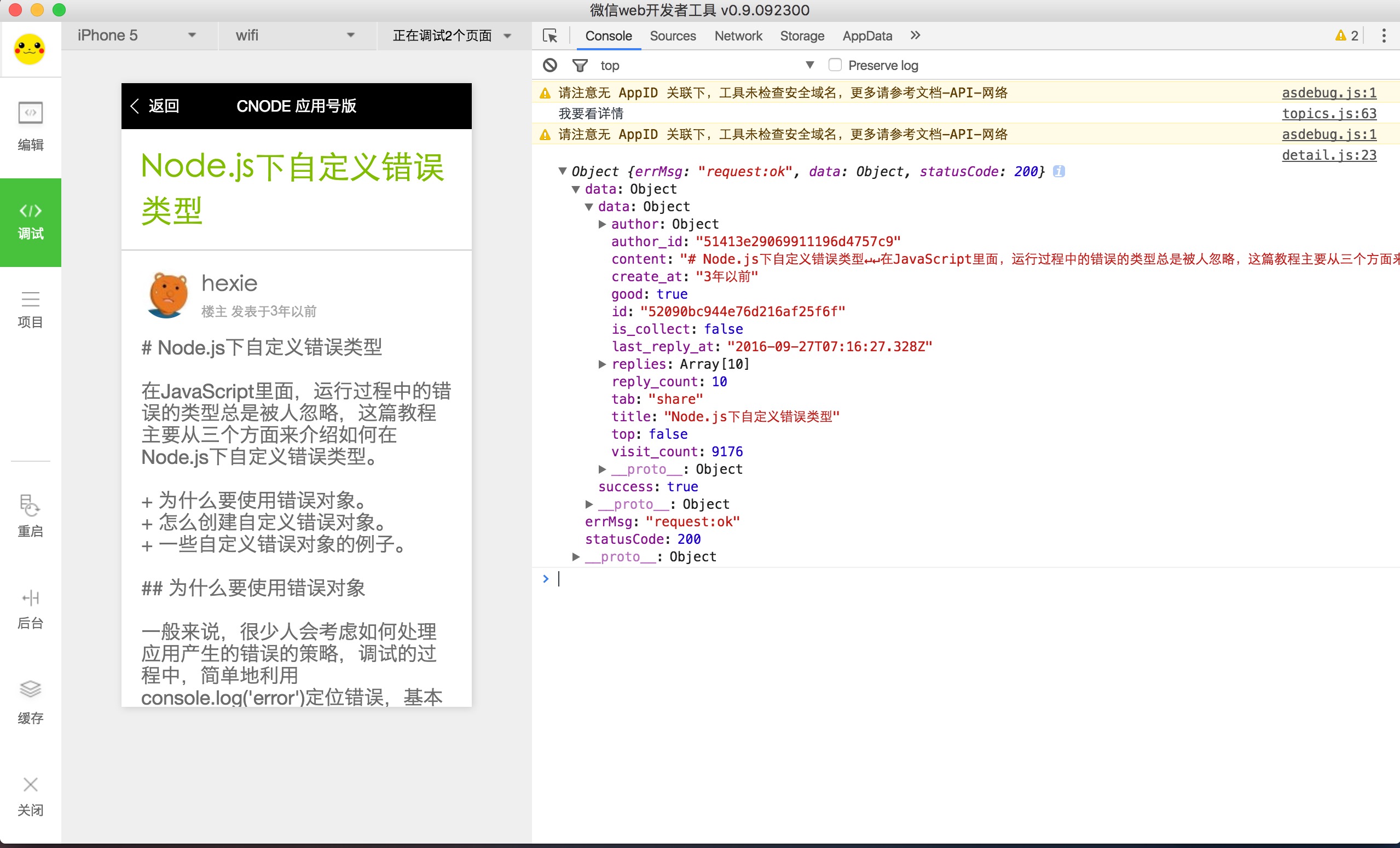
url: url

})

},

### 4.实现详情页

同样的原理，创建 detail 文件，并注册，获取数据，并美化页面。



### 5.总结

* 微信小应用页面的脚本逻辑在是在 JsCore 中运行， JsCore 是一个没有窗口对象的环境，所以不能再脚本中使用 window ，也无法在脚本中操作组件
* 同样不能用 jquery
* 也不能操作 dom
* 部分标签不支持，比如 h1-h6 会编译报错。
* 暂时没找到解决富文本详情页显示的办法。
* 整体下来，感觉开发简单，限制很多。
* 写过 react 的看这个确实比较简单。