**公众号开发**

目录

[前言 2](#_Toc515722585)

[开发准备工作 2](#_Toc515722586)

[开发详述 2](#_Toc515722587)

[数据交互流程 2](#_Toc515722588)

[开发者模式接入校验 2](#_Toc515722589)

[公众号Token 2](#_Toc515722590)

[公众号菜单 2](#_Toc515722591)

[公众号消息 3](#_Toc515722592)

[普通消息 4](#_Toc515722593)

[事件消息 5](#_Toc515722594)

[公众号用户管理 6](#_Toc515722595)

[获取用户列表 6](#_Toc515722596)

[获取某用户基本信息 7](#_Toc515722597)

[用户分组管理（标签） 7](#_Toc515722598)

# 前言

1. 微信公众平台（<https://mp.weixin.qq.com/>）注册属于自己的公众号。注册还是很简单的，有问题请参看：<http://kf.qq.com/faq/120322fu63YV130422AJbaI3.html>
2. 个人只能注册订阅号，且不可认证，对于开发者而言，这样的公众号根本不能满足需求。不过..别急，微信方为开发者开放了测试号，可体验各种高级接口
3. 对于服务号和订阅号的区分，网上已经很多人探讨过了，这里不再做赘述，给个网址各位看官自行浏览： <https://www.zhihu.com/question/21289814>
4. 如果只是要在自己的公众号写些随笔、发发文章，那么微信公众平台网页就足够了。本文主要记录开发者模式下，程序调用微信接口以实现预定功能的过程。

# 开发准备工作

1. 进入微信公众平台，在左侧导航栏找到“开发者工具”，点击后找到“公众平台测试账号”进入。



1. 进入测试号后，你会看到测试号信息appID 和 appsecret，这两个非常重要，非常重要，非常重要！是你进行后续开发的必备信息。



接口配置信息是一定要填写的，这里的Token是要你自己随意填写的一串字符，后续代码中还会用到。填写的URL需要正确响应微信发送的Token验证。即你需要有自己的服务器资源，且要发布一个相应的网站地址。

1. 说到这里，就不得不再说说怎样才能拥有自己的服务器资源。

现在的云服务已经大众化，你可以任选一家租用服务器，一般服务器购买后会有配备教程教你怎么设置。

开通好一个实例后要怎样做呢？当然是在你的服务器上配置IIS啦，不然怎么发布网站呢，配置IIS的过程肯定也不是一帆风顺的=。= 这里有几篇博客可能会帮到你，也可以自去Google

<http://blog.csdn.net/luchengtao11/article/details/72283237>

（云服务器Windows Server2012 配置http服务器(又称Web服务器，IIS)）

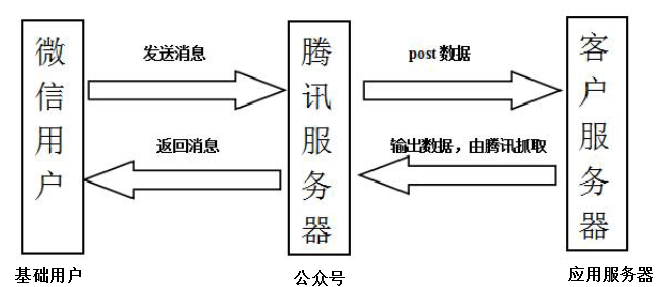
<http://www.cnblogs.com/fuhai/p/7078398.html>

（Windows Server 2012部署遇到的问题）

好了，现在服务器已经准备就绪，接下来就尽情的玩耍吧~

# 开发详述

## 数据交互流程



首先要理解数据交互。后台逻辑方法要与微信客户端交互，必须经由腾讯服务器进行中转，不能直接发送消息。

## 开发者模式接入校验

1. 设计用来接收微信消息和事件的接口URL，初次对接时用来验证身份和有效性。设计的接口需要能够验证签名（验证开发者身份，GET方式）和处理请求。
2. 登录公众号平台官网填写服务器配置，将发布后的URL、Token和EncodingAESKey填入配置中。其中Token由开发者任意填写，用作生成签名，配置中填写的Token会和接口URL中设计的Token进行对比，所以填写的Token值必须一致。EncodingAESKey可由系统随机生成。
3. 提交信息后，微信服务器会发送GET请求到填写的URL中，其中携带了四个参数：

|  |  |
| --- | --- |
| 参数 | 描述 |
| signature | 微信加密签名，signature结合了开发者填写的token参数和请求中的timestamp参数、nonce参数 |
| timestamp | 时间戳 |
| nonce | 随机数 |
| echostr | 随机字符串 |

1. 加密/校验流程：
2. 将token、timestamp、nonce三个参数进行字典序排序
3. 将三个参数字符串拼接成一个字符串进行sha1加密
4. 开发者获得加密后的字符串可与signature对比，标识该请求来源于微信，若相等则验证成功。成功后即接入生效成为开发者。

## 公众号Token

微信公众平台有两个Token，名字一样，但是是两个概念。一个是腾讯微信公众平台服务器和我们自己的服务器交互的密钥，是通过我们自己在服务器上设置的，注意在程序中首次校验签名时的token值要与服务器上设置的一致；另外一个是通过appID和appsecret 获取操作菜单、发送客服消息等所需的一个凭据，注意不要混淆！



## 公众号菜单

微信公众号的菜单最多允许有三个一级菜单，每个一级菜单下最多允许有五个二级菜单。一级菜单最多4个汉字，二级菜单最多7个汉字，多出来的部分将会以“…”代替。

菜单接口可实现多种类型的按钮，分别是click（点击推事件）、view（跳转URL）、scancode\_push（扫码推事件）、scancode\_waitmsg（扫码推事件且弹出“消息接收中”提示框）、pic\_sysphoto（弹出系统拍照发图）、 pic\_photo\_or\_album（弹出拍照或者相册发图）、 pic\_weixin（弹出微信相册发图器）、 location\_select（弹出地理位置选择器）。每个菜单有对应的name和key值。

接口调用请求说明：

（1）创建菜单：http请求方式：POST（请使用https协议） https://api.weixin.qq.com/cgi-bin/menu/create?access\_token=ACCESS\_TOKEN

（2）查询菜单：http请求方式：GET

https://api.weixin.qq.com/cgi-bin/menu/get?access\_token=ACCESS\_TOKEN

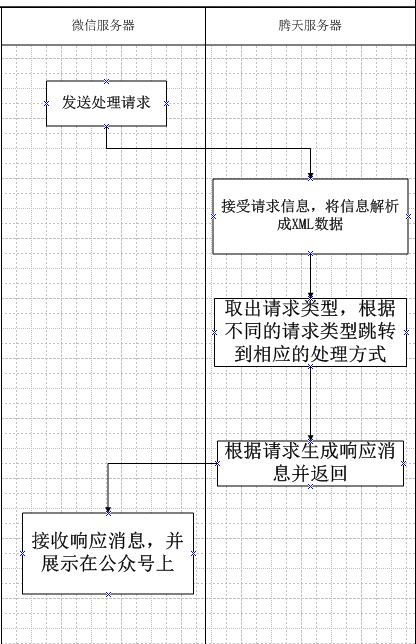
（3）删除菜单：http请求方式：GET

https://api.weixin.qq.com/cgi-bin/menu/delete?access\_token=ACCESS\_TOKEN

【**注意**】提供的接口不支持对菜单进行局部修改，若想编辑菜单项，需要删除整个菜单后重新创建。

## 公众号消息

公众号请求处理流程：



公众号请求类型可以分为事件（Event）和普通消息（文本Text，图片Picture，语音Voice，视频Video，地理位置Location和链接Link）。

当微信用户向公众账号发普通消息时，微信服务器将Post消息的XML数据包到开发者填写的URL上，根据不同类型的普通消息会有不同的XML数据包结构。

当微信用户向公众账号发起事件消息时，微信会把事件推送给开发者，一般情况下为用户选择自定义菜单后触发。事件消息有不同种事件类型（EventKey），分为:

(1)点击菜单拉取消息时的事件推送（Click）

(2)点击菜单跳转链接时的事件推送（View）

(3)扫码推事件的事件推送（scancode\_push）、扫码推事件并弹出“消息接收中”提示框的事件退送（scancode\_waitmsg）

(4)弹出系统拍照发图的事件推送（pic\_sysphoto）、弹出拍照或者相册发图的事件推送（pic\_photo\_or\_album）、弹出微信相册发图器的事件推送（pic\_weixin）

(5)弹出地理位置选择器的事件推送（location\_select）

(6)关注/取消关注事件（subscribe/unsubscribe）

根据不同的事件类型也会有不同的XML数据包结构。

### 普通消息

#### 文本消息

<xml>

<ToUserName></ToUserName>

<FromUserName></FromUserName>

<CreateTime></CreateTime>

<MsgType></MsgType>

<Content></Content>

<MsgId></MsgId>

</xml>

|  |  |
| --- | --- |
| 参数 | 描述 |
| ToUserName | 开发者微信号 |
| FromUserName | 发送方微信号（OpenID） |
| CreateTime | 消息创建时间（整型） |
| MsgType | 消息类型（此例为text） |
| Content | 文本消息内容（根据消息类型字段参数会有所不同，此例为文本内容） |
| MsgId | 消息Id，64位整型 |

#### 图片消息

<xml>

<ToUserName></ToUserName>

<FromUserName></FromUserName>

<CreateTime></CreateTime>

<MsgType></MsgType>

<PicUrl></PicUrl>

<MediaId></MediaId>

<MsgId></MsgId>

</xml>

|  |  |
| --- | --- |
| 参数 | 描述 |
| ToUserName | 开发者微信号 |
| FromUserName | 发送方微信号（OpenID） |
| CreateTime | 消息创建时间（整型） |
| MsgType | 消息类型（此例为image） |
| PicUrl | 图片链接 |
| MediaId | 图片消息媒体id，可以调用多媒体文件下载接口拉取数据。 |
| MsgId | 消息Id，64位整型 |

#### 语音消息

<xml>

<ToUserName></ToUserName>

<FromUserName></FromUserName>

<CreateTime></CreateTime>

<MsgType></MsgType>

<MediaId></MediaId>

<Format></Format>

<MsgId></MsgId>

</xml>

|  |  |
| --- | --- |
| 参数 | 描述 |
| ToUserName | 开发者微信号 |
| FromUserName | 发送方微信号（OpenID） |
| CreateTime | 消息创建时间（整型） |
| MsgType | 消息类型（此例为voice） |
| MediaId | 语音消息媒体id，可以调用多媒体文件下载接口拉取数据。 |
| Format | 语音格式，如amr，speex等 |
| MsgId | 消息Id，64位整型 |

#### 视频消息

<xml>

<ToUserName></ToUserName>

<FromUserName></FromUserName>

<CreateTime></CreateTime>

<MsgType></MsgType>

<MediaId></MediaId>

<ThumbMediaId></ThumbMediaId>

<MsgId></MsgId>

</xml>

|  |  |
| --- | --- |
| 参数 | 描述 |
| ToUserName | 开发者微信号 |
| FromUserName | 发送方微信号（OpenID） |
| CreateTime | 消息创建时间（整型） |
| MsgType | 消息类型（此例为video，小视频为shortvideo） |
| MediaId | 视频消息媒体id，可以调用多媒体文件下载接口拉取数据。 |
| ThumbMediaId | 视频消息缩略图的媒体id，可以调用多媒体文件下载接口拉取数据。 |
| MsgId | 消息Id，64位整型 |

#### 地理位置消息

<xml>

<ToUserName></ToUserName>

<FromUserName></FromUserName>

<CreateTime></CreateTime>

<MsgType></MsgType>

<Location\_X></Location\_X>

<Location\_Y></Location\_Y>

<Scale>20</Scale>

<Label></Label>

<MsgId></MsgId>

</xml>

|  |  |
| --- | --- |
| 参数 | 描述 |
| ToUserName | 开发者微信号 |
| FromUserName | 发送方微信号（OpenID） |
| CreateTime | 消息创建时间（整型） |
| MsgType | 消息类型（此例为location） |
| Location\_X | 地理位置维度 |
| Location\_Y | 地理位置经度 |
| Scale | 地图缩放大小 |
| Label | 地理位置信息 |
| MsgId | 消息Id，64位整型 |

#### 链接消息

<xml>

<ToUserName></ToUserName>

<FromUserName></FromUserName>

<CreateTime></CreateTime>

<MsgType></MsgType>

<Title></Title>

<Description></Description>

<Url></Url>

<MsgId></MsgId>

</xml>

|  |  |
| --- | --- |
| 参数 | 描述 |
| ToUserName | 开发者微信号 |
| FromUserName | 发送方微信号（OpenID） |
| CreateTime | 消息创建时间（整型） |
| MsgType | 消息类型（此例为link） |
| Title | 消息标题 |
| Description | 消息描述 |
| Url | 消息链接 |
| MsgId | 消息Id，64位整型 |

### 事件消息

#### 关注/取消关注事件

<xml>

<ToUserName></ToUserName>

<FromUserName></FromUserName>

<CreateTime></CreateTime>

<MsgType></MsgType>

<Event></Event>

</xml>

|  |  |
| --- | --- |
| 参数 | 描述 |
| ToUserName | 开发者微信号 |
| FromUserName | 发送方微信号（OpenID） |
| CreateTime | 消息创建时间（整型） |
| MsgType | 消息类型（此例为event） |
| Event | 事件类型，subscribe(订阅)、unsubscribe(取消订阅) |

#### 点击菜单拉取消息时的事件推送

<xml>

<ToUserName></ToUserName>

<FromUserName></FromUserName>

<CreateTime></CreateTime>

<MsgType></MsgType>

<Event></Event>

<EventKey></EventKey>

</xml>

|  |  |
| --- | --- |
| 参数 | 描述 |
| ToUserName | 开发者微信号 |
| FromUserName | 发送方微信号（OpenID） |
| CreateTime | 消息创建时间（整型） |
| MsgType | 消息类型（此例为event） |
| Event | 事件类型，click |
| EventKey | 事件KEY值，与自定义菜单接口中KEY值对应 |

#### 点击菜单跳转链接时的事件推送

<xml>

<ToUserName></ToUserName>

<FromUserName></FromUserName>

<CreateTime></CreateTime>

<MsgType></MsgType>

<Event></Event>

<EventKey></EventKey>

</xml>

|  |  |
| --- | --- |
| 参数 | 描述 |
| ToUserName | 开发者微信号 |
| FromUserName | 发送方微信号（OpenID） |
| CreateTime | 消息创建时间（整型） |
| MsgType | 消息类型（此例为event） |
| Event | 事件类型，view |
| EventKey | 事件KEY值，设置的跳转URL |

#### 扫一扫事件推送

首先创建菜单的事件类型为scancode\_waitmsg/scancode\_push，点击菜单后会弹出微信扫码界面，扫描正确的二维码后，扫码所得的信息会由微信推送给开发者，推送的XML数据包结构为：

<xml><ToUserName><![CDATA[gh\_e136c6e50636]]></ToUserName>

<FromUserName><![CDATA[oMgHVjngRipVsoxg6TuX3vz6glDg]]></FromUserName>

<CreateTime>1408090606</CreateTime>

<MsgType><![CDATA[event]]></MsgType>

<Event><![CDATA[scancode\_waitmsg]]></Event>

<EventKey><![CDATA[6]]></EventKey>

<ScanCodeInfo><ScanType><![CDATA[qrcode]]></ScanType>

<ScanResult><![CDATA[2]]></ScanResult>

</ScanCodeInfo>

</xml>

获取其中的ScanResult参数的内容即可获取二维码对应的字符串信息。

#### 上传图片事件推送

首先创建菜单的事件类型为pic\_photo\_or\_album/ pic\_sysphoto/ pic\_weixin，点击菜单后可以选择拍照或者从手机相册中选择照片上传给公众号。上传照片后，公众号会以图片消息的数据包格式传给开发者，数据包结构为：

<xml>

<ToUserName><![CDATA[toUser]]></ToUserName>

<FromUserName><![CDATA[fromUser]]></FromUserName>

<CreateTime>1348831860</CreateTime>

<MsgType><![CDATA[image]]></MsgType>

<PicUrl><![CDATA[this is a url]]></PicUrl>

<MediaId><![CDATA[media\_id]]></MediaId>

<MsgId>1234567890123456</MsgId>

</xml>

其中PicUrl参数为上传的照片临时保存在微信服务器上的URL，会在三天以后自动删除。

#### 发送位置事件推送

首先创建菜单的事件类型为location\_select，点击菜单后会弹出地理位置选择器，发送位置后，公众号会以地理位置消息的数据包格式传给开发者。

**Ps**：公众号有三种方式可以获取到用户位置：

1）用户发送地理位置，即公众号接收地理位置消息；

2）通过自定义菜单设置菜单类型为location\_select，点击菜单后会弹出地理位置选择器；

3）通过JS SDK在网页上弹出是否允许使用地理位置的询问，用户允许后才可获得。

菜单发送地理位置方式是发送地理位置消息格式的数据包，不会触发事件方法。而地理位置选择器是可以搜索位置发送的，也就是上报的位置有可能不是用户当前所在是位置

## 公众号用户管理

### 获取用户列表

公众号可通过本接口来获取帐号的关注者列表，关注者列表由一串OpenID（加密后的微信号，每个用户对每个公众号的OpenID是唯一的）组成。一次拉取调用最多拉取10000个关注者的OpenID，可以通过多次拉取的方式来满足需求。

http请求方式: GET（请使用https协议）

<https://api.weixin.qq.com/cgi-bin/user/get?access_token=ACCESS_TOKEN&next_openid=NEXT_OPENID>

其中next\_Openid为第一个拉取的Opendid，不填默认从头开始拉取。

正确时返回的Json数据包：

{"total":2,"count":2,"data":{"openid":["","OPENID1","OPENID2"]},"next\_openid":"NEXT\_OPENID"}

其中total表示关注该公众号的总用户数，count为拉取的总用户数，data为openid列表，next\_openid为拉取的最后一个用户的openid。

### 获取某用户基本信息

在关注者与公众号产生消息交互后，公众号可获得关注者的OpenID（加密后的微信号，每个用户对每个公众号的OpenID是唯一的。对于不同公众号，同一用户的openid不同）。公众号可通过本接口来根据OpenID获取用户基本信息，包括昵称、头像、性别、所在城市、语言和关注时间。

http请求方式: GET

<https://api.weixin.qq.com/cgi-bin/user/info?access_token=ACCESS_TOKEN&openid=OPENID&lang=zh_CN>

### 用户分组管理（标签）

开发者可以使用用户标签管理的相关接口，实现对公众号的标签进行创建、查询、修改、删除等操作，也可以对用户进行打标签、取消标签等操作。

可以利用标签实现分组功能，标签可用于不同的个性化菜单，即可实现不同分组下的用户看到的菜单不一样。

1. 创建标签：

一个公众号，最多可创建100个标签，且标签名在30个字符内。

http请求方式: POST（请使用https协议）

<https://api.weixin.qq.com/cgi-bin/tags/create?access_token=ACCESS_TOKEN>

POST数据格式：json

POST数据例子：{ "tag" : { "name" : "广东" //标签名 } }

返回数据包：{ "tag":{ "id":134,//标签id "name":"广东" } }，其中id由微信分配。

1. 获取公众号已创建的标签：

http请求方式: GET（请使用https协议）

<https://api.weixin.qq.com/cgi-bin/tags/get?access_token=ACCESS_TOKEN>

返回数据包：{ "tags":[{ "id":1, "name":"每天一罐可乐星人", "count":0 //此标签下粉丝数 },{"id":2,"name":"星标组","count":0 },{"id":127, "name":"广东","count":5 } ] }

其中id为分组编号，name为分组名称，count为组内用户数量

1. 编辑标签

http请求方式: POST（请使用https协议）

<https://api.weixin.qq.com/cgi-bin/tags/update?access_token=ACCESS_TOKEN>

POST数据格式：json

POST数据例子：{ "tag" : { "id" : 134, "name" : "广东人" } }，其中id为分组编号，name为需要修改的分组名称

1. 删除标签

当某个标签下的粉丝超过10w时，后台不可直接删除标签。此时，开发者可以对该标签下的openid列表，先进行取消标签的操作，直到粉丝数不超过10w后，才可直接删除该标签。

http请求方式: POST（请使用https协议）

<https://api.weixin.qq.com/cgi-bin/tags/delete?access_token=ACCESS_TOKEN>

POST数据格式：json

POST数据例子：{"tag":{"id":108}}

1. 获取标签下粉丝列表

http请求方式：GET（请使用https协议）

https://api.weixin.qq.com/cgi-bin/user/tag/get?access\_token=ACCESS\_TOKEN

POST数据格式：JSON

POST数据例子：

{ "tagid" : 134,

"next\_openid":""//第一个拉取的OPENID，不填默认从头开始拉取

}

返回数据包：

{ "count":2,//这次获取的粉丝数量

"data":{//粉丝列表

"openid":[

"ocYxcuAEy30bX0NXmGn4ypqx3tI0",

"ocYxcuBt0mRugKZ7tGAHPnUaOW7Y" ]

},

"next\_openid":"ocYxcuBt0mRugKZ7tGAHPnUaOW7Y"//拉取列表最后一个用户的//openid

}

1. 获取用户身上的标签列表

http请求方式：POST（请使用https协议）

https://api.weixin.qq.com/cgi-bin/tags/getidlist?access\_token=ACCESS\_TOKEN

POST数据格式：JSON

POST数据例子：{ "openid" : "ocYxcuBt0mRugKZ7tGAHPnUaOW7Y" }

返回数据包：

{ "tagid\_list":[//被置上的标签列表 134, 2 ] }

1. 批量为用户打标签

http请求方式：POST（请使用https协议）

https://api.weixin.qq.com/cgi-bin/tags/members/batchtagging?access\_token=ACCESS\_TOKEN

POST数据例子：

{ "openid\_list" : [//粉丝列表

"ocYxcuAEy30bX0NXmGn4ypqx3tI0",

"ocYxcuBt0mRugKZ7tGAHPnUaOW7Y" ],

"tagid" : 134 }

1. 批量为用户取消标签

http请求方式：POST（请使用https协议）

https://api.weixin.qq.com/cgi-bin/tags/members/batchuntagging?access\_token=ACCESS\_TOKEN

POST数据格式：JSON

POST数据例子：

{ "openid\_list" : [//粉丝列表

"ocYxcuAEy30bX0NXmGn4ypqx3tI0",

"ocYxcuBt0mRugKZ7tGAHPnUaOW7Y" ],

"tagid" : 134 }

处理请求程序框架说明：

（1）WeChatHttpHandler.ashx：请求入口，即发布在服务器上的地址入口，用于接收请求消息和返回响应消息。

（2）WeChatService.cs：处理请求，产生相应。其中包含了验证签名（GET）和处理消息（POST）的方法。若需要处理请求，将请求消息传给HandlerFactory（处理工厂）进行请求类型的判断和处理，最后将得到的响应消息返回给WeChatHttpHandler。

（3）Handlers：包含HandlerFactory.cs, EventHandler.cs, TextHandler.cs等。HandlerFactory用来判断请求消息类型，根据消息类型转入相应的类型处理类（例如EventHandler.cs）。EventHandler.cs中还需要根据事件类型进行相应的处理。

（4）Models：包含不同种消息类型的模型类，根据微信公众号开发文档中各普通消息的XML数据包结构解析而来。

（5）Menus：用来响应不同菜单功能的类，其中MenuFactory类用来判断用户点击了哪一个菜单项（根据菜单的Key来对应），转入不同的类方法进行处理。

代码文件说明

1.EventHandler.cs:先根据EventMessage中的Event字段的值判断消息类型，再根据EventKey字段的值判断对应的菜单的Key值，进行请求处理。

2.TextHandler.cs:将请求信息通过loadFromXml方法解析请求内容，还可以通过HandleOther方法来根据用户输入不同的文字信息来做出不同的响应。

3. ImageHandler.cs：通过LoadFromXml方法将上传的照片信息进行解析，得到picurl字段的值并返回给用户。

4. EventMessage.cs：事件消息类，用于解析事件消息的XML数据包结构，根据事件类型的不同，解析的字段也会有所不同。

5. ImageMessage.cs：图片消息类，用于解析图片消息的XML数据包结构。

6. TextMessage.cs：文本消息类，解析文本消息的XML数据包结构。

7. NewsMessage.cs：图文消息类，该类下可以包含多条SingleNews类的实例，可以提供多条目供用户选择。SingleNews中可以设置消息的标题，描述，跳转链接等信息。

**JS SDK**

网页上获取地理位置

**使用微信内置地图查看位置接口**

wx.openLocation({  
latitude: 0, // 纬度，浮点数，范围为90 ~ -90  
longitude: 0, // 经度，浮点数，范围为180 ~ -180。  
name: '', // 位置名  
address: '', // 地址详情说明  
scale: 1, // 地图缩放级别,整形值,范围从1~28。默认为最大  
infoUrl: '' // 在查看位置界面底部显示的超链接,可点击跳转  
});

**获取地理位置接口**

wx.getLocation({  
type: 'wgs84', // 默认为wgs84的gps坐标，如果要返回直接给openLocation用的火星坐标，可传入'gcj02'  
success: function (res) {  
var latitude = res.latitude; // 纬度，浮点数，范围为90 ~ -90  
var longitude = res.longitude; // 经度，浮点数，范围为180 ~ -180。  
var speed = res.speed; // 速度，以米/每秒计  
var accuracy = res.accuracy; // 位置精度  
}  
});