**[C#开发微信门户及应用(15)-微信菜单增加扫一扫、发图片、发地理位置功能](http://www.cnblogs.com/wuhuacong/p/3988858.html)**

前面介绍了很多篇关于使用C#开发微信门户及应用的文章，基本上把当时微信能做的接口都封装差不多了，微信框架也积累了不少模块和用户，最近发现微信公众平台增加了不少内容，特别是在自定义菜单里面增加了扫一扫、发图片、发地理位置功能，这几个功能模块很重要，想想以前想在微信公众号里面增加一个扫描二维码的功能，都做不了，现在可以了，还可以拍照上传等功能，本文主要介绍基于我前面的框架系列文章，进一步介绍如何集成和使用这些新增功能。

**1、微信几个功能的官方介绍**

1）. 扫码推送事件

用户点击按钮后，微信客户端将调起扫一扫工具，完成扫码操作后显示扫描结果（如果是URL，将进入URL），且会将扫码的结果传给开发者，开发者可以下发消息。

2）. 扫码推送事件，且弹出“消息接收中”提示框

用户点击按钮后，微信客户端将调起扫一扫工具，完成扫码操作后，将扫码的结果传给开发者，同时收起扫一扫工具，然后弹出“消息接收中”提示框，随后可能会收到开发者下发的消息。

3）. 弹出系统拍照发图

用户点击按钮后，微信客户端将调起系统相机，完成拍照操作后，将拍摄的相片发送给开发者，并推送事件给开发者，同时收起系统相机，随后可能会收到开发者下发的消息。

4）. 弹出拍照或者相册发图

用户点击按钮后，微信客户端将弹出选择器供用户选择“拍照”或者“从手机相册选择”。用户选择后即走其他两种流程。

5）. 弹出微信相册发图器

用户点击按钮后，微信客户端将调起微信相册，完成选择操作后，将选择的相片发送给开发者的服务器，并推送事件给开发者，同时收起相册，随后可能会收到开发者下发的消息。

6）. 弹出地理位置选择器

用户点击按钮后，微信客户端将调起地理位置选择工具，完成选择操作后，将选择的地理位置发送给开发者的服务器，同时收起位置选择工具，随后可能会收到开发者下发的消息。  
但请注意，以上新增能力，均仅支持微信iPhone5.4.1以上版本，和Android5.4以上版本的微信用户，旧版本微信用户点击后将没有回应，开发者也不能正常接收到事件推送。

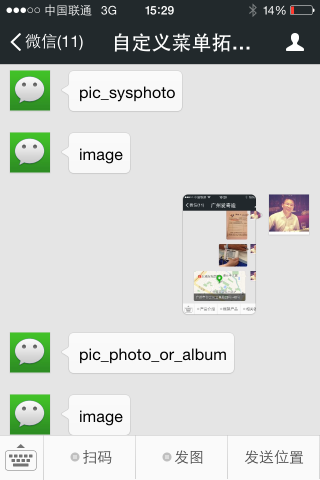
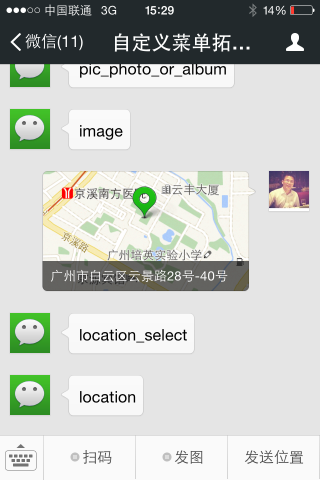
**2、微信新菜单功能的测试公众号**

微信不仅增加了这些功能模块的支持，还考虑到我们开发人员的方便，增加了一个叫做“menutest"的公众号，方便我们测试。我们在公众号搜索“menutest"，然后关注它即可进行测试几个新增功能了。



“menutest"的公众号名称是”自定义菜单拓展测试“，我关注它并进行了测试，二维码、图片、地理位置都很OK，本身能够响应这些事件，并且图片、地理位置自身还能出现一个对应的事件，如下所示。

图片发送可以分为拍照、拍照和相册、微信相册三类，感觉后面两个有点类似，但有这些功能都很不错的。

**3、改进菜单对象和提交菜单**

前面说了，微信提供这些功能，可以在菜单里面进行集成，也就是菜单的类型由原来CLICK/VIEW两种，变为现在8种类型，增加2个扫码操作、3种图片操作、1种地理位置操作。

因此把菜单的枚举类型扩展一下，如下所示。

[复制代码](javascript:void(0);)

/// <summary>

/// 菜单按钮类型

/// </summary>

public enum ButtonType

{

/// <summary>

/// 点击

/// </summary>

click,

/// <summary>

/// Url

/// </summary>

view,

/// <summary>

/// 扫码推事件的事件推送

/// </summary>

scancode\_push,

/// <summary>

/// 扫码推事件且弹出“消息接收中”提示框的事件推送

/// </summary>

scancode\_waitmsg,

/// <summary>

/// 弹出系统拍照发图的事件推送

/// </summary>

pic\_sysphoto,

/// <summary>

/// 弹出拍照或者相册发图的事件推送

/// </summary>

pic\_photo\_or\_album,

/// <summary>

/// 弹出微信相册发图器的事件推送

/// </summary>

pic\_weixin,

/// <summary>

/// 弹出地理位置选择器的事件推送

/// </summary>

location\_select

}

[复制代码](javascript:void(0);)

然后在Winform里面调用创建菜单操作代码如下所示：

[复制代码](javascript:void(0);)

private void btnCreateMenu\_Click(object sender, EventArgs e)

{

MenuJson productInfo = new MenuJson("新功能测试", new MenuJson[] {

new MenuJson("扫码推事件", ButtonType.scancode\_push, "scancode\_push")

,new MenuJson("系统拍照发图", ButtonType.pic\_sysphoto, "pic\_sysphoto")

, new MenuJson("拍照相册发图", ButtonType.pic\_photo\_or\_album, "pic\_photo\_or\_album")

, new MenuJson("微信相册发图", ButtonType.pic\_weixin, "pic\_weixin")

, new MenuJson("地理位置选择", ButtonType.location\_select, "location\_select")

});

MenuJson frameworkInfo = new MenuJson("框架产品", new MenuJson[] {

new MenuJson("Win开发框架", ButtonType.click, "win"),

new MenuJson("WCF开发框架", ButtonType.click, "wcf"),

new MenuJson("混合式框架", ButtonType.click, "mix"),

new MenuJson("Web开发框架", ButtonType.click, "web")

,new MenuJson("代码生成工具", ButtonType.click, "database2sharp")

});

MenuJson relatedInfo = new MenuJson("相关链接", new MenuJson[] {

new MenuJson("公司介绍", ButtonType.click, "event\_company"),

new MenuJson("官方网站", ButtonType.view, "http://www.iqidi.com"),

new MenuJson("联系我们", ButtonType.click, "event\_contact"),

new MenuJson("应答系统", ButtonType.click, "set-1"),

new MenuJson("人工客服", ButtonType.click, "event\_customservice")

});

MenuListJson menuJson = new MenuListJson();

menuJson.button.AddRange(new MenuJson[] { productInfo, frameworkInfo, relatedInfo });

if (MessageUtil.ShowYesNoAndWarning("您确认要创建菜单吗") == System.Windows.Forms.DialogResult.Yes)

{

IMenuApi menuBLL = new MenuApi();

CommonResult result = menuBLL.CreateMenu(token, menuJson);

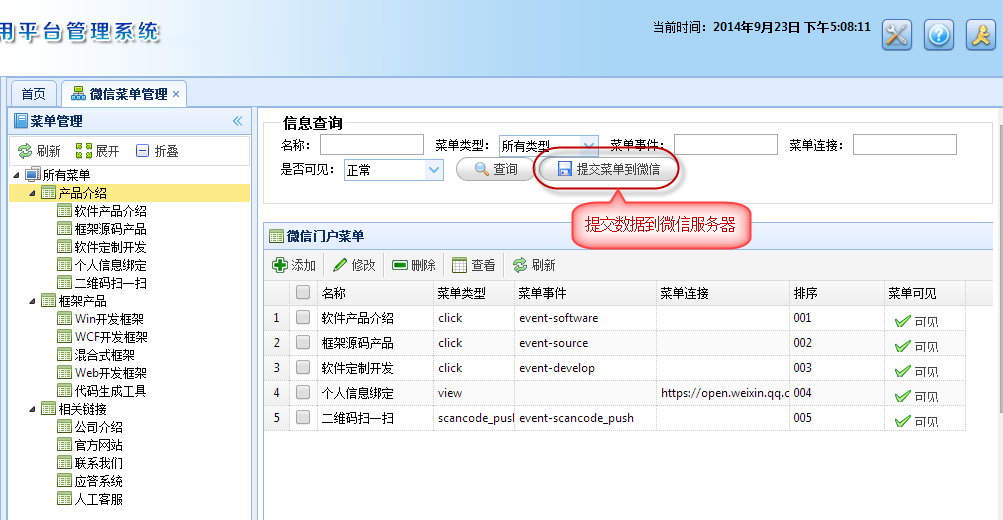
Console.WriteLine("创建菜单：" + (result.Success ? "成功" : "失败:" + result.ErrorMessage));

}

}

[复制代码](javascript:void(0);)

当然，一般情况下我们都是在Web后台系统进行的，维护菜单都是在自己微信平台上进行菜单管理，然后一次性提交到微信服务器即可。



而在Web后台，只需要把数据库的数据变化为Json数据提交即可，操作和上面的类似。

[复制代码](javascript:void(0);)

/// <summary>

///更新微信菜单

/// </summary>

/// <returns></returns>

public ActionResult UpdateWeixinMenu()

{

string token = base.GetAccessToken();

MenuListJson menuJson = GetWeixinMenu();

IMenuApi menuApi = new MenuApi();

CommonResult result = menuApi.CreateMenu(token, menuJson);

return ToJsonContent(result);

}

[复制代码](javascript:void(0);)

**4、微信扫一扫功能集成**

 前面讲了，有了最新的功能，我们就可以实现扫一扫功能，从而可以扫描条形码，二维码的功能。有了条形码、二维码的快速和识别，我们就能开发一些如条码查询、商品处理等功能了。

这里我们介绍如何在我的微信开发框架里面整合这个扫一扫的功能处理操作。

前面已经增加了一些新功能的测试菜单，我们要做的就是响应这些事件处理，然后对他们进行应答处理就可以了。

下面是根据事件进行的一些API跳转处理，我们同时定义了几个相关的实体类用来处理他们的信息，如RequestEventScancodePush、RequestEventScancodeWaitmsg、RequestEventPicSysphoto等等。

RequestEventScancodeWaitmsg实体类的代码如下所示，其他的类似处理。

[复制代码](javascript:void(0);)

/// <summary>

/// 扫码推事件且弹出“消息接收中”提示框的事件推送

/// </summary>

[System.Xml.Serialization.XmlRoot(ElementName = "xml")]

public class RequestEventScancodeWaitmsg : BaseEvent

{

public RequestEventScancodeWaitmsg()

{

this.MsgType = RequestMsgType.Event.ToString().ToLower();

this.Event = RequestEvent.scancode\_waitmsg.ToString();

this.ScanCodeInfo = new ScanCodeInfo();

}

/// <summary>

/// 事件KEY值，由开发者在创建菜单时设定

/// </summary>

public string EventKey { get; set; }

/// <summary>

/// 扫描信息

/// </summary>

public ScanCodeInfo ScanCodeInfo { get; set; }

}

[复制代码](javascript:void(0);)

而根据实体类强类型的处理接口流转操作如下所示。

[复制代码](javascript:void(0);)

case RequestEvent.scancode\_push:

{

//扫码推事件的事件推送

RequestEventScancodePush info = XmlConvertor.XmlToObject(postStr, typeof(RequestEventScancodePush)) as RequestEventScancodePush;

if (info != null)

{

responseContent = actionBLL.HandleEventScancodePush(info);

}

}

break;

case RequestEvent.scancode\_waitmsg:

{

//扫码推事件且弹出“消息接收中”提示框的事件推送

RequestEventScancodeWaitmsg info = XmlConvertor.XmlToObject(postStr, typeof(RequestEventScancodeWaitmsg)) as RequestEventScancodeWaitmsg;

if (info != null)

{

responseContent = actionBLL.HandleEventScancodeWaitmsg(info);

}

}

break;

case RequestEvent.pic\_sysphoto:

{

//弹出系统拍照发图的事件推送

RequestEventPicSysphoto info = XmlConvertor.XmlToObject(postStr, typeof(RequestEventPicSysphoto)) as RequestEventPicSysphoto;

if (info != null)

{

responseContent = actionBLL.HandleEventPicSysphoto(info);

}

}

break;  
..................

[复制代码](javascript:void(0);)

处理扫描结果并返回的最终代码如下所示。

[复制代码](javascript:void(0);)

/// <summary>

/// 扫码推事件且弹出“消息接收中”提示框的事件推送的处理

/// </summary>

/// <param name="info">扫描信息</param>

/// <returns></returns>

public string HandleEventScancodeWaitmsg(RequestEventScancodeWaitmsg info)

{

ResponseText response = new ResponseText(info);

response.Content = string.Format("您的信息为：{0}，可以结合后台进行数据查询。", info.ScanCodeInfo.ScanResult);

return response.ToXml();

}

[复制代码](javascript:void(0);)

最后我们测试扫描一个条形码，可以看到返回的结果界面操作如下所示。



**5、新菜单功能测试发现的问题**

前面介绍了一些新菜单功能模块的集成，我个人对这种扫一扫菜单功能非常赞赏，这也是微信逐步整合更多硬件资源和接口处理的趋向，不过在集成使用的时候，发现公众号偶尔出现闪退的情况，还有就是这些新功能虽然后台能够实现数据的处理和接收，但是有一些不能返回应答消息，很郁闷。也许随着版本研发的加快，这些功能很快得到完善和解决。

另外微信开放平台也投入使用了，好些认证也是300元一年，不过暂时没有其应用的场景，我只是用到了它来获取微信账号的unionid的功能，其他功能慢慢了解吧。

还有就是微信的企业号也已经出来了，而且我也已经申请认证通过，它的开发用户的API也有不少，有空继续研究并整合到微信开发框架里面吧。