**[C#开发微信门户及应用(9)-微信门户菜单管理及提交到微信服务器](http://www.cnblogs.com/wuhuacong/p/3734614.html)**

微信公众号（包括服务号和订阅号）都可以对菜单进行自定义设置，我们为了方便管理，一般先把菜单数据在本地管理维护，需要更新的时候，把它们更新到微信服务器上就可以了。本文基于这个方式，介绍我的微信门户平台管理系统中菜单提交到微信服务器上的操作。微信门户应用管理系统，采用基于MVC+EasyUI的路线，由于多数域名服务器上都只能支持.NET4.0，所以以MVC3，C#4.0作为开发基础，基本上能够部署在任何.NET服务器上。

**1、微信菜单的要求及相关界面设计**

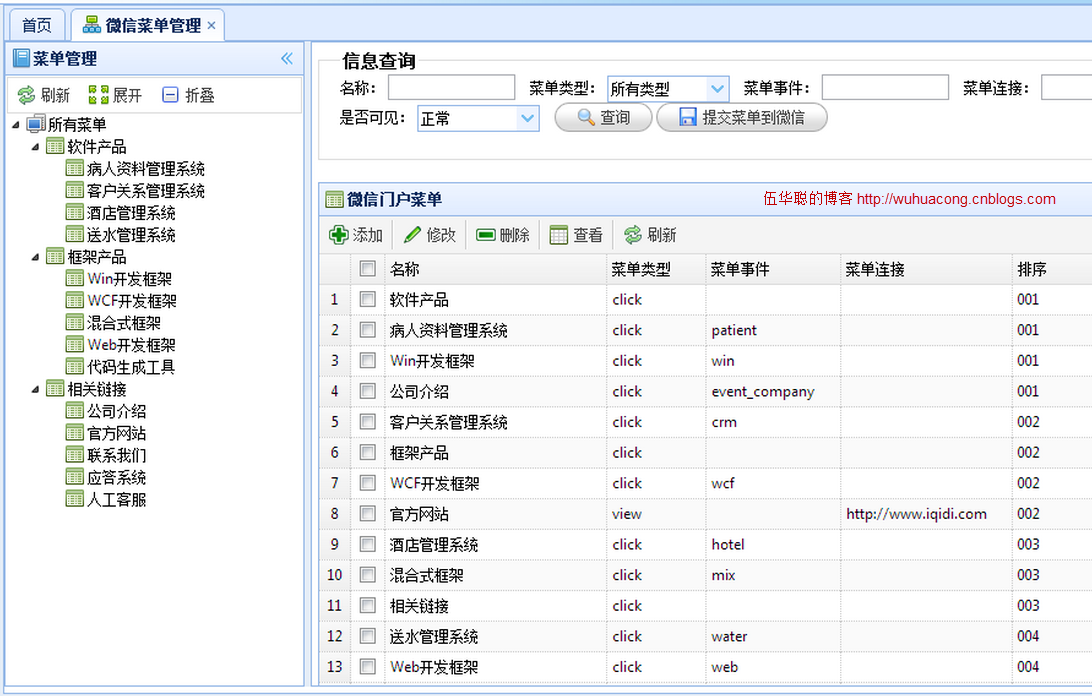
微信公众号的菜单我们可以通过网站进行本地的管理，维护好它们之间的层级关系，由于微信对自定义的菜单要求比较严格，以下是微信对自定义菜单的要求：

**目前自定义菜单最多包括3个一级菜单，每个一级菜单最多包含5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。**

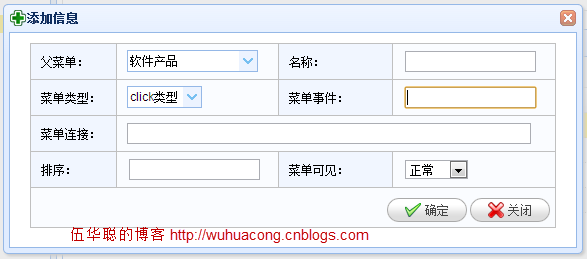
因此我们自己根据约定，不要越界即可，否则提交菜单到服务器，可能会返回一些错误，这些细节，我们在创建本地菜单管理的时候，注意一下就可以了。我在早期的一篇文章也介绍了自定义菜单的一些内容，需要可以进行回顾一下《[C#开发微信门户及应用(6)--微信门户菜单的管理操作](http://www.cnblogs.com/wuhuacong/p/3701961.html)》，本篇主要是介绍在我的平台管理系统里面，调用前面介绍的菜单接口API，实现菜单提交到服务器的操作。

根据微信的自定义菜单要求，我在管理系统里面，对微信的菜单几个基础性的界面设计如下。

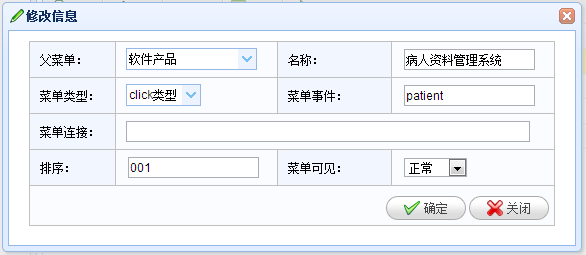
主菜单管理界面如下所示。



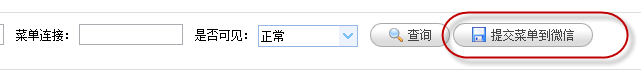
添加菜单的界面设计如下所示



微信菜单的修改界面如下所示



微信菜单定义是存储在数据库里面，如果需要提交到微信服务器上并生效，则需要调用微信API接口进行处理，我在页面的Controller控制器里增加一个提交到服务器的处理方法。



**2、提交菜单到微信服务器的操作**

上面几个界面，主要就是根据微信菜单的属性，对菜单进行维护管理，我们最终的目的是把它们放到服务器上去，供我们处理客户的相关事件操作的。

提交菜单的操作，我们在MVC的View页面里面，使用JQuery的Ajax提交即可（前提是我们在控制器里面添加相应的处理，后面介绍），界面脚本代码如下所示。

[复制代码](javascript:void(0);)

//绑定提交按钮的的点击事件

function BindSubmitEvent() {

$("#btnSubmit").click(function () {

$.messager.confirm("提交菜单确认", "您确认需要提交菜单到微信服务器吗？", function (action) {

if (action) {

//提交数据

$.ajax({

url: '**/Menu/UpdateWeixinMenu**',

type: 'post',

dataType: 'json',

success: function (data) {

if (data.Success) {

$.messager.alert("提示", "提交微信菜单成功");

}

else {

$.messager.alert("提示", "提交微信菜单失败:" + data.ErrorMessage);

}

},

data: ''

});

}

});

});

}

[复制代码](javascript:void(0);)

上面红色的代码，就是我们在MVC的控制器里面定义的方法，我们只需要通过POST方法，对控制器方法调用，就能实现菜单提交到微信服务器上，至于具体里面的细节，我们可以把它挪到控制器或者更底层进行处理就是了，页面不需要涉及太多的逻辑就是了。

上面那个Menu控制器的UpdateWeixinMenu的方法代码如下所示（主要就是根据我前面介绍过的开发模型进行处理就是了）。

[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);)

上面的几个方法这里逐一介绍一下。GetAccessToken主要就是获得当前操作的访问令牌，这里的操作可以用缓存进行缓存，否则频繁的获取AccessToken，达到每天指定的次数后，当天就不能再用了。

GetWeixinMenu方法，主要就是为了方便，对获取构造微信的自定义菜单数据进行了一个函数封装，具体代码如下所示。

[复制代码](javascript:void(0);)

/// <summary>

/// 生成微信菜单的Json数据

/// </summary>

/// <returns></returns>

private MenuListJson GetWeixinMenu()

{

MenuListJson menuJson = new MenuListJson();

List<MenuNodeInfo> menuList = BLLFactory<Menu>.Instance.GetTree();

foreach (MenuNodeInfo info in menuList)

{

ButtonType type = (info.Type == "click") ? ButtonType.click : ButtonType.view;

string value = (type == ButtonType.click) ? info.Key : info.Url;

MenuJson weiInfo = new MenuJson(info.Name, type, value);

AddSubMenuButton(weiInfo, info.Children);

menuJson.button.Add(weiInfo);

}

return menuJson;

}

[复制代码](javascript:void(0);)

[复制代码](javascript:void(0);)

private void AddSubMenuButton(MenuJson menu, List<MenuNodeInfo> menuList)

{

if (menuList.Count > 0)

{

menu.sub\_button = new List<MenuJson>();

}

foreach (MenuNodeInfo info in menuList)

{

ButtonType type = (info.Type == "click") ? ButtonType.click : ButtonType.view;

string value = (type == ButtonType.click) ? info.Key : info.Url;

MenuJson weiInfo = new MenuJson(info.Name, type, value);

menu.sub\_button.Add(weiInfo);

AddSubMenuButton(weiInfo, info.Children);

}

}

[复制代码](javascript:void(0);)

上面的代码，就是把本地存储的MenuNodeInfo数据，通过递归遍历的方式，转换为微信的自定义菜单实体MenuJson，这样我们调用API就非常方便了，这个函数主要负责构造对应的实体信息就是了。至于调用微信API提交菜单的事情，还是让API自己亲自处理为好，他们的代码如下所示（也就是上面函数的部分代码）。

IMenuApi menuApi = new MenuApi();

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

return ToJsonContent(result);

最终的结果是返回一个通用的结果CommonResult，这个结果对象，非常方便脚本的处理，如果有错误，则提示错误，否则也方便判断布尔值，也就是上面的页面代码脚本。

[复制代码](javascript:void(0);)

success: function (data) {

if (data.Success) {

$.messager.alert("提示", "提交微信菜单成功");

}

else {

$.messager.alert("提示", "提交微信菜单失败:" + data.ErrorMessage);

}

},

[复制代码](javascript:void(0);)

通过以上几部分的代码，我们就可以实现前台MVC的视图界面，调用后台封装好的微信API，实现菜单的提交处理了。

如果感兴趣或者体验相关的客服应答功能，可以关注我的微信了解下。具体效果可以关注我的微信门户：广州爱奇迪，也可以扫描下面二维码进行关注了解。