自定义菜单

一 介绍

> 微信支持我们自定义菜单实现一些功能,但是因为微信的页面显示是由微信提供的,并不是我们的服务端提供的,所以我们必须按照微信的要求提供数据
>
> 微信要求我们需要将要显示的菜单提前通知到微信,由微信做存储,显示菜单的时候,他们会从自己的服务器上面根据我们传递的 token 来决定显示哪个公众号的按钮
>
> 所以自定义菜单 需要我们自己编写我们自己的后台页面,提供一个后台接口,在我们的后台将要添加的按钮的内容发送到腾讯服务器,腾讯服务器做存储
>
> 具体参考开发文档
>
> https://mp.weixin.qq.com/wiki?t=resource/res_main&id=mp1421141013

1.0 流程

1. 首先官网获取到 appid 和开发者密码
2. 编写自己的服务器
3. 在自己的服务器内部调用微信提供的接口 获取 ACCES_TOKEN
4. 在自己的服务器内部调用微信创建自定义菜单的按钮地址,将 自己的ACCES_TOKEN替换掉地址地址中的值,然后按照微信要求传递对应格式的 菜单 json 数据即可
5. 访问微信公众号,使用菜单

1.1 请求地址

1.1.1 获取 ACCES_TOKEN 的地址

> GET 请求
>
> grant_type为固定值
>
> appid 为我们的微信 id
>
> secret 为我们的开发者密码
>
> 开发者密码获取方式:后台设置的基本设置中开启开发者密码,扫码后获取,注意服务器不存储这个内容,我们需要自己存储,否则只能重置,最后添加白名单,只有白名单中的 ip 地址才可以访问这些接口
>
> 注意: 此接口有调用次数限制 一天200次

```
https://api.weixin.qq.com/cgi-bin/token?
grant_type=client_credential&appid=APPID&secret=APPSECRET
```

1.1.2 创建按钮的地址

> POST 请求
>
> ACCESS_TOKEN为我们上面方法获取到的值

```
https://api.weixin.qq.com/cgi-bin/menu/create?access_token=ACCESS_TOKEN
```

1.1.3 发送的数据类型

```
{
     "button":[
     {
          "type":"click",
          "name":"今日歌曲",
          "key":"V1001_TODAY_MUSIC"
      },
      {
           "name":"菜单",
           "sub_button":[
           {
               "type":"view",
               "name":"搜索",
               "url":"http://www.soso.com/"
            },
            {
                 "type":"miniprogram",
                 "name":"wxa",
                 "url":"http://mp.weixin.qq.com",
                 "appid":"wx286b93c14bbf93aa",
                 "pagepath":"pages/lunar/index"
            },
            {
               "type":"click",
               "name":"赞一下我们",
               "key":"V1001_GOOD"
            }]
       }]
}
```

参数说明

> 所有的按钮在点击后会将请求发送至我们的基本服务器地址,消息类型为 event 类型

参数	是否必须	说明
button	是	一级菜单数组，个数应为1~3个
sub_button	否	二级菜单数组，个数应为1~5个
type	是	菜单的响应动作类型，view表示网页类型，click表示点击类型，miniprogram表示小程序类型,type 会被微信服务器作为 event 参数值传递到我们的服务器
name	是	菜单标题，不超过16个字节，子菜单不超过60个字节
key	click等点击类型必须	菜单KEY值，用于消息接口推送，不超过128字节,此参数会被微信以EventKey的参数值传递到我们的服务器
url	view、miniprogram类型必须	网页 链接，用户点击菜单可打开链接，不超过1024字节。type为miniprogram时，不支持小程序的老版本客户端将打开本url。
media_id	media_id类型和view_limited类型必须	调用新增永久素材接口返回的合法media_id
appid	miniprogram类型必须	小程序的appid（仅认证公众号可配置）
pagepath	miniprogram类型必须	小程序的页面路径

二 服务器搭建

> 此处服务器未使用数据库,而是使用假数据

2.0 pom

```
<!--在前面基础项目的基础上添加此依赖包-->
<dependency>
    <groupId>net.sf.json-lib</groupId>
    <artifactId>json-lib</artifactId>
    <version>2.4</version>

</dependency>
```

2.1 controller

```
/**
 * Created by jackiechan on 2018/1/24/上午9:13
 */
@RestController
public class MenuController {
    @RequestMapping("/createMenu")
    @ResponseBody
    public String createMenu() {
      //此处首先获取 token, 因为此接口不能每次都调用,所以应该调用此接口获取数据后保存起
来,首次需要调用此方法
       // AccessToken accessToken = WeiXinUtil.getAccessToken(WeiXinUtil.APPID,
WeiXinUtil.APPSECRET);
      /*  if (accessToken == null) {
            return "token 获取失败,创建菜单失败";
        } else {
*/
            int menu = WeiXinUtil.createMenu(init(), "此处需要设置为上面注释代码返回
的 TOKEN");
            if (menu == 0) {
                return "创建成功";
            } else {
                return "创建菜单失败";
            }
       // }
    }
//初始化一些菜单
    public Menu init() {
        ViewButton viewButton = new ViewButton();//点击的按钮,自定义的对象
        viewButton.setName("网址按钮");//按钮名字长度16个字节内
        viewButton.setType("view");//注意 type 必须严格按照微信要求填写,否则会出现需要
utf-8字符集的错误
        viewButton.setUrl("http://www.baidu.com");
        ClickButton clickButton = new ClickButton();
        clickButton.setName("点击按钮");
        clickButton.setKey("clickkey");
        clickButton.setType("click");
        CommonButton commonButton = new CommonButton();
        commonButton.setName("按钮1");
        List<Button> buttons = new ArrayList<>();
        buttons.add(clickButton);
        buttons.add(viewButton);
        commonButton.setSub_button(buttons);

        Menu menu = new Menu();//菜单
        List<Button> menuList = new ArrayList<>();
        menuList.add(commonButton);
        ClickButton clickButton1 = new ClickButton();
```

```
        clickButton1.setType("click");
        clickButton1.setKey("clickkey2");
        clickButton1.setName("立刻购买");
        menuList.add(clickButton1);
        menu.setButton(menuList);
        return menu;
    }
}
```

2.2AccessToken

```
public class AccessToken {
    // 获取到的凭证
    private String token;
    // 凭证有效时间，单位：秒
    private int expiresIn;

    public String getToken() {
        return token;
    }

    public void setToken(String token) {
        this.token = token;
    }

    public int getExpiresIn() {
        return expiresIn;
    }

    public void setExpiresIn(int expiresIn) {
        this.expiresIn = expiresIn;
    }
}
```

2.3 WeiXinUtil

```
/**
 * Created by jackiechan on 2018/1/24/上午9:40
 */
public class WeiXinUtil {
    public static final String APPID = "自己的 APPID";
    public static final String APPSECRET = "自己的开发者密码";
```

```
    // 获取access_token的接口地址（GET） 限200（次/天）
    public final static String access_token_url = "https://api.weixin.qq.com/cgi-bin/token?grant_type=client_credential&appid=APPID&secret=APPSECRET";
    // 菜单创建（POST） 限100（次/天）
    public static String menu_create_url = "https://api.weixin.qq.com/cgi-bin/menu/create?access_token=ACCESS_TOKEN";

    public static JSONObject httpRequest(String requestUrl, String requestMethod, String outputStr) {
        JSONObject jsonObject = null;
        StringBuffer buffer = new StringBuffer();
        try {
            // 创建SSLContext对象，并使用我们指定的信任管理器初始化
            TrustManager[] tm = { new MyX509TrustManager() };
            SSLContext sslContext = SSLContext.getInstance("SSL", "SunJSSE");
            sslContext.init(null, tm, new java.security.SecureRandom());
            // 从上述SSLContext对象中得到SSLSocketFactory对象
            SSLSocketFactory ssf = sslContext.getSocketFactory();

            URL url = new URL(requestUrl);
            HttpsURLConnection httpUrlConn = (HttpsURLConnection) url.openConnection();
            httpUrlConn.setSSLSocketFactory(ssf);
            httpUrlConn.setDoOutput(true);
            httpUrlConn.setRequestProperty("Content-Type", "application/json;charset=utf-8");
            httpUrlConn.setDoInput(true);
            httpUrlConn.setUseCaches(false);
            // 设置请求方式（GET/POST）
            httpUrlConn.setRequestMethod(requestMethod);

            if ("GET".equalsIgnoreCase(requestMethod))
                httpUrlConn.connect();

            // 当有数据需要提交时
            if (null != outputStr) {
                OutputStream outputStream = httpUrlConn.getOutputStream();
                // 注意编码格式，防止中文乱码
                outputStream.write(outputStr.getBytes("UTF-8"));
                outputStream.close();
            }

            // 将返回的输入流转换成字符串
            InputStream inputStream = httpUrlConn.getInputStream();
            InputStreamReader inputStreamReader = new InputStreamReader(inputStream, "utf-8");
            BufferedReader bufferedReader = new BufferedReader(inputStreamReader);

            String str = null;
```

```
            while ((str = bufferedReader.readLine()) != null) {
                buffer.append(str);
            }
            bufferedReader.close();
            inputStreamReader.close();
            // 释放资源
            inputStream.close();
            inputStream = null;
            httpUrlConn.disconnect();
            System.out.println(buffer.toString());
            jsonObject = JSONObject.fromObject(buffer.toString());
        } catch (ConnectException ce) {
            System.out.println("Weixin server connection timed out.");
        } catch (Exception e) {
            System.err.println("https request error:{}");
//            log.error("https request error:{}", e);
        }
        return jsonObject;
    }

    /**
     * 获取access_token
     *
     * @param appid 凭证
     * @param appsecret 密钥
     * @return
     */
    public static AccessToken getAccessToken(String appid, String appsecret) {
        AccessToken accessToken = null;

                String requestUrl = access_token_url.replace("APPID",
appid).replace("APPSECRET", appsecret);
        JSONObject jsonObject = httpRequest(requestUrl, "GET", null);
        // 如果请求成功
        if (null != jsonObject) {
            try {
                accessToken = new AccessToken();
                accessToken.setToken(jsonObject.getString("access_token"));
                accessToken.setExpiresIn(jsonObject.getInt("expires_in"));
            } catch (Exception e) {
                accessToken = null;
                // 获取token失败
                System.out.println("获取token失败
errcode:"+jsonObject.getInt("errcode")+"errmsg:"+jsonObject.getString("errmsg"));
//                log.error("获取token失败 errcode:{} errmsg:{}",
jsonObject.getInt("errcode"), jsonObject.getString("errmsg"));
            }
        }
        return accessToken;
```

```
    }
    /**
     * 创建菜单
     *
     * @param menu 菜单实例
     * @param accessToken 有效的access_token
     * @return 0表示成功，其他值表示失败
     */
    public static int createMenu(Menu menu, String accessToken) {
        int result = 0;

        // 拼装创建菜单的url
        String url = menu_create_url.replace("ACCESS_TOKEN", accessToken);
        // 将菜单对象转换成json字符串
        String jsonMenu = JSONObject.fromObject(menu).toString();
        System.out.println("发送的数据--"+jsonMenu);
        // 调用接口创建菜单
        JSONObject jsonObject = httpRequest(url, "POST", jsonMenu);

        if (null != jsonObject) {
            if (0 != jsonObject.getInt("errcode")) {
                result = jsonObject.getInt("errcode");
                System.out.println("创建菜单失败
errcode:"+jsonObject.getInt("errcode")+"errmsg:"+jsonObject.getString("errmsg"));
//                log.error("创建菜单失败 errcode:{} errmsg:{}",
jsonObject.getInt("errcode"), jsonObject.getString("errmsg"));
            }
        }

        return result;
    }
}
```

2.4 MyX509TrustManager

```
/**
 * 证书信任管理器（用于https请求）
 *
 *Created by jackiechan on 2018/1/24/上午9:40
 */
public class MyX509TrustManager implements X509TrustManager{
    public void checkClientTrusted(X509Certificate[] chain, String authType)
throws CertificateException {
    }

    public void checkServerTrusted(X509Certificate[] chain, String authType)
throws CertificateException {
    }
```

```
    public X509Certificate[] getAcceptedIssuers() {
        return null;
    }
}
```

2.5 Menu

```
/**
 * Created by jackiechan on 2018/1/24/上午9:45
 * 菜单的基类,用于生成 json 数据,内部放多个按钮
 */
public class Menu {
    private List<Button> button;

    public List<Button> getButton() {
        return button;
    }

    public void setButton(List<Button> button) {
        this.button = button;
    }
}
```

2.6 Button

```
/**
 * Created by jackiechan on 2018/1/24/上午9:45
 * 按钮的基类
 */
public class Button {
    private String name;

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }
}
```

2.7 ClickButton

```
/**
 * Created by jackiechan on 2018/1/24/上午9:39
 * 点击类型的按钮
 */
public class ClickButton  extends  Button{
    private String type = "click";
    private String key;

    public String getType() {
        return type;
    }

    public void setType(String type) {
        this.type = type;
    }

    public String getKey() {
        return key;
    }

    public void setKey(String key) {
        this.key = key;
    }
}
```

2.8 ViewButton

```
/**
 * Created by jackiechan on 2018/1/24/上午9:38
 * 网址 小程序类型的按钮
 */
    public class ViewButton extends Button {
    private String type = "view";

    private String url;

    public String getUrl() {
        return url;
    }

    public void setUrl(String url) {
        this.url = url;
    }

    public String getType() {
```

```
        return type;
    }

    public void setType(String type) {
        this.type = type;
    }
}
```

2.9 CommonButton

```
/**
 * Created by jackiechan on 2018/1/24/上午9:37
 * 带有子菜单的按钮
 */
public class CommonButton extends  Button {
    private List<Button> sub_button;

    public List<Button> getSub_button() {
        return sub_button;
    }

    public void setSub_button(List<Button> sub_button) {
        this.sub_button = sub_button;
    }
}
```

2.10 启动服务器,测试

我们需要将项目发布到白名单地址的服务器上面才可以,然后访问我们添加菜单的接口地址 测试即可

三 使用菜单

微信的按钮使用,view 的点击后会跳转到对应的地址

click 类型的点击后会发送数据到我们的服务器(地址为我们基础配置中的地址),相当于聊天方式发送了特定的内容,只不过消息类型不是文本类型

```
[CreateTime=1516763250, EventKey=clickkey, Event=CLICK,
ToUserName=gh_ea92b17cb9e9, FromUserName=oUuptwrJudIfdihz1Z_T1AciMahs,
MsgType=event]
```

其中MsgType就是消息类型,为固定值event
如果消息类型为 event, 则EventKey代表的就是发送过来的内容(相当于文本消息的 content),这个内容就是我们当初创建按钮时候设置的 key 的内容
Event 参数对应的值 则代表按钮的 type
所以我们只需要判断用户发送的是什么类型的请求,获取到对应的内容,然后根据预先定好的内容,执行对应的业务逻辑操作即可

错误

1. 如果按钮的 type 写的不对 也会提示需要 utf-8错误
2. 名字超出长度