此文章已于 15:37:46 2015/12/4 发布到 lngg057

关于Google+以及Facebook第三方登录实现的一点总结

类别 [发布为日记]

# 简述

最近项目中有关于第三方登陆的需求，第三方Facebook以及Google +登录。

正好这几天把这个需求做得差不多了，收个尾，作为一个这方面之前基本从未涉及的小白，总结下开发流程以及过程中遇到的一些问题。希望能对大家有所帮助

基本上，目前互联网上的涉及到第三方登录采用的都为OAuth认证方式，相关原理以及实现方法相关的可以网上搜索查看。不过涉及到Facebook以及Google +的登陆，最大的问题是这两个网站在墙外，想翻墙找资料比较费劲，奈何国内这方面的资料也不是很齐全，翻到的官方的文档又全是英文的，阅读无能~

第三方登陆简单可以分为以下几个步骤，分别是：**获得第三方登陆许可认证**；**根据规范添加页面登录样式，跳转到第三方登陆页面登陆并获取授权后，返回原指定页面（或请求）**；**后台对登录信息进行校验，并获取对应第三方登陆用户的信息**；**将第三方用户信息与本系统用户进行关联**。

下面就以上几个步骤的一些关键信息，或者说是我开发当中遇到的问题做一下简要的记录。因为Facebook以及Google +第三方登陆都采用的是Auth2技术，因此只会在具体实现不同的地方单独说明。理解有误的地方，或者是没有说全的地方，还请各位谅解~

# 一 获得第三方登陆许可认证

在获取许可上，需要在对应的开发平台上申请登陆接入许可，基本上你可能需要准备以下几样东西：

* + 你当前需要接入第三方登陆的网址域名（用于第三方平台校验是否这个登录请求来自于你的网站本身）;
  + 给用户展示的网站logo和文字（用于在第三方登陆的时候给用户直观的展（装）示（逼））;
  + 登陆成功后，跳回的URL（用于获取并使用登陆用户的相关信息）;

申请成功之后，第三方平台会提供给你对应的AppID以及AppSecret，用以后台进行额外的操作。基本上操作按照官方api的顺序一步一步来就可以了惹，这里就不再赘述了~（请掌握基本的英语阅读能力…）

# 二 根据规范添加页面登录样式，跳转到第三方登陆页面登陆并获取授权后，返回原指定页面（或请求）

简单点讲就是在页面上添加登录Button，并通过点击触发跳转到第三方的请求。Google+以及Facebook在大体上逻辑一致，在细节上实现略有不同。

**Google+登录**，会引用自己封装的js（<script src="https://apis.google.com/js/platform.js" defer async="true"></script>）来实现登录操作（gapi），并提供了多种登陆实现方案可供选择。这里贴一下，我目前项目中实现的方案，仅供参考。

1.在对应页面引用Google第三方js。<https://apis.google.com/js/api:client.js>（这里的js其实和官方最新的platform.js内容是一样的…吧~）

2.页面定义Google+登录的按钮样式（这里谷歌有内置的规范按钮样式可以直接调用，不用自己定义）

（id="googleLogin"）

3.根据引用的第三方js提供的api，作如下初始化操作：

|  |
| --- |
| auth2 = gapi.auth2.init({  client\_id: clientID, //第一步申请的时候，提供给你的appid  cookiepolicy: 'single\_host\_origin',  scope: 'https://www.googleapis.com/auth/plus.login https://www.googleapis.com/auth/userinfo.email' //需要获取的用户信息领域  }); |

4.初始化后，接着需要对你定义的按钮绑定鼠标点击事件，作如下操作：

|  |
| --- |
| auth2.attachClickHandler('googleLogin', {}, startApp.onSuccess, startApp.onFailure); //googleLogin为按钮的ID |

5.做完这些之后就等用户点击（上钩）啦~这里需要注意下startApp.onSuccess以及startApp.onFailure方法，这里是在谷歌成功登陆之后的回调函数。在onSuccess里第三方已经登录并返回，我们已经可以根据接口api获取想要的用户信息了，而在onFailure里因为登录失败，我们可以自定义一些提示信息。

6.等等！上面代码中的auth2，startApp是干嘛用的啊喂！别慌，下面附上完整的js！项目开发使用了RequireJs，代码仅供参考！

|  |
| --- |
| define("glogin", ['google', 'jquery'], **function**(google, $) {  **var** startApp = {};  **var** googleUser = {};  **var** clientID = "#your AppID#";  startApp.onSuccess = **function**(googleUser) {  **var** profile = googleUser.getBasicProfile();  **var** authProfile = googleUser.getAuthResponse();  **var** userName = profile.getName();  **var** userImage = profile.getImageUrl();  **var** userEmail = profile.getEmail();  **var** idToken = authProfile.id\_token;    **var** flag = **true**;  **if** (userEmail && idToken) {  $.ajax({  url: "/action.do",  data: {  userName: userName,  firstName: userName,  lastName: userName,  email: userEmail,  idToken: idToken,  userType: '3'  },  dataType: "json",  tyep: "post",  success: **function**(data) {  **if** (data.success == **true**) {  window.location.reload(**true**);  } **else** {  flag = **false**;  }  }  });  }  **if** (!flag) {  alert.alert("XXX.");  }  };  startApp.onFailure = **function**(error) {  console.log(error);  alert.alert("XXX.");  };  startApp.init = **function**() {  gapi.load('auth2', **function**() {  auth2 = gapi.auth2.init({  client\_id: clientID,  cookiepolicy: 'single\_host\_origin',  scope: 'https://www.googleapis.com/auth/plus.login https://www.googleapis.com/auth/userinfo.email'  });  auth2.attachClickHandler('googleLogin', {}, startApp.onSuccess, startApp.onFailure);  });  };  **return** startApp;  });  require(["glogin"], **function**(main) {  main.init();  }); |

**Facebook登录**，同样会调用自己封装的js（<script src="https: //connect.facebook.net/en\_US/sdk.js" defer async="true"></script>）来实现登录操作（FB），实现步骤基本同Google+登陆类似。

1.在对应页面引用Facebook第三方js。[https: //connect.facebook.net/en\_US/sdk.js](https://apis.google.com/js/api:client.js);

2.页面定义Facebook登录的按钮样式

（id="fblogin"）

3.根据引用的第三方js提供的api，作如下初始化操作：

|  |
| --- |
| FB.init ({  appId : '#your AppID#',  cookie : true,  xfbml : true,  version : 'v2.4'  }); |

4.初始化后，接着需要对你定义的按钮绑定鼠标点击登录事件，作如下操作：

|  |
| --- |
| $("#fblogin").click(**function** (){  FB.login(**function**(response) {  statusChangeCallback(response); //登录回调函数  },{scope: 'email'}); //需要获取的信息scope  }); |

5.statusChangeCallback为登录回调函数，需要在里面做下登陆返回的相关操作：

|  |
| --- |
| **var** statusChangeCallback = **function** (response){  **if** (response.status === 'connected') {  **var** accessToken = response.authResponse.accessToken; //取得 accessToken    FB.api('/me?fields=name,first\_name,last\_name,email', **function**(response) { // /me为API指定的调用方法，用于获取登陆用户相关信息  **if**(response.email!=**null**){  $.ajax({  url:"/action.do",  data:{  userName:response.name,  firstName:response.first\_name,  lastName:response.last\_name,  email:response.email,  idToken:accessToken,  userType:'2'  },  dataType:"json",  tyep:"post",  success:**function**(data){  **if**(data.success == **true**){  window.location.reload(**true**);  }  }  });  }**else**{  alert("XXX.");  }  });  }  }; |

# 三 后台对登录信息进行校验，并获取对应第三方登陆用户的信息

在用户正确的登录第三方平台并进行授权之后，调用返回函数，这时需要对用户的登录信息和状态进行确认！这里非常关键，因为要确保用户的确是从对应第三方平台登录，所以我们需要第三方平台提供的access\_token值，根据一定的校验规则进行校验，放行校验通过的登陆请求。

由于验证信息涉及到app的隐私信息，以及确保验证不会被人为的篡改，因此需要在服务器后端进行token的验证。

参见第二条，在第三方调用返回函数之后，代码中会进行ajax请求，调用后台的对应接口 /action.do，来在后台进行access\_token的正确性判断。

后台会根据登陆的第三方类型，进行https请求验证，来确保用户登陆的信息无误。

Google+的token验证地址：https://www.googleapis.com/oauth2/v3/tokeninfo?id\_token=XXX

Facebook的token验证地址（%7C为encode的 ‘|’符号）：https://graph.facebook.com/debug\_token?access\_token={Your AppId}%7C{Your AppSecret}&input\_token=XXX

因为这里后台采用JAVA开发，在验证地址为HTTPS请求验证的时候，后台利用HttpClient请求的时候涉及到证书验证的问题，所以这里采用了 @rongyongfeikai2同学的方法：《JAVA利用HttpClient进行POST请求（HTTPS）》，有需要的可以参考借鉴。（http://blog.csdn.net/rongyongfeikai2/article/details/41659353）

后台根据对应官方的api文档的token校验方式，对获取到的token进行校验。如果请求的验证地址返回的信息为校验通过，则可以进行后面的操作了。

# 四 将第三方用户信息与本系统用户进行关联

在进行前端js请求的时候，对应平台的api接口会获取到用户的信息（参照上述代码）。与此同时，在进行用户token校验的时候，返回的对应json对象中也会包含对应的用户信息，这些信息可以在后台进行解析和获取，详细的数据结构可以参考官方文档。为了确保用户信息不会被中途截断篡改，可以在后台进行数据获取。

在获取到第三方登录用户的信息之后，就可以与本系统的账户进行匹配操作了。之后的操作存在几种不同的方式类型，对于本系统来讲，采用的是以下鉴定方式。

|  |
| --- |
| 第三方登陆  Token校验通过（区分来源）  与系统内部账户号进行匹配  查询无对应用户  在系统内部创建该用户，生成随即六位数密码  将本系统内的帐号及密码信息发送到用户对应第三方账号邮箱中  查询有对应用户  数据库校验字段为1  数据库校验字段为0  直接登录系统  引导用户到邮箱帐号校验页，并发送唯一code校验码进行校验  邮箱url校验通过  将数据库对应校验字段置为1 |

# 后叙

以上为目前我们系统中的第三方登录的基本操作信息，也许并不太完整（毕竟基本上之前没怎么涉及过…），但是希望其中的涉及到的点能够帮助到大家，也希望大家能指出写的不对的地方，并多多包涵哈。

PS：第一次用Word的博客文章分享写blog并发到博客园，好紧张~

PPS：第一次写博客，不知道该怎么写，明明想的很多，一写出来就不知道该写什么好惹……

以下为开发过程中参考的文章地址，有需求的朋友可以自行点击查阅！

Google+登录api：<https://developers.google.com/identity/sign-in/web/devconsole-project>

Google+自定义button：<https://developers.google.com/identity/sign-in/web/build-button>

Google+ token验证：<https://developers.google.com/identity/sign-in/web/backend-auth>

Facebook登录api：<https://developers.facebook.com/docs/facebook-login?locale=zh_CN>

Facebooktoken验证 api：<https://developers.facebook.com/docs/facebook-login/access-tokens/debugging-and-error-handling>

Facebook token验证解决方案：<http://stackoverflow.com/questions/19359868/how-would-you-perform-debug-token-call-using-facebook-php-sdk>

后台https地址请求解决方法：<http://blog.csdn.net/rongyongfeikai2/article/details/41659353>