微博Android SDK 3.2文档

1.1 认证授权

新浪微博Android SDK为开发者提供了Oauth2.0授权认证，并集成SSO登录功能，使第三方应用无需了解复杂的验证机制即可进行授权登录操作，并提供微博分享功能，第三方应用可直接通过微博客户端进行分享。
本文档介绍了新浪微博Android SDK的三种授权方式，各种分享并给出简单的示例分析，帮助第三方开发者快速集成应用。

1.2 名词解释

名词	注解
AppKey	分配给每个第三方应用的唯一身份认证，用来区分来源等功能（获取方式：http://open.weibo.com/development/mobile）
RedirectURI	第三方应用授权回调页面。 授权回调页对移动客户端应用来说对用户是不可见的，所以定义为何种形式都将不影响，但是没有定义将无法使用SDK认证登录。建议使用默认回调页https://api.weibo.com/oauth2/default.html 可以在“新浪微博开放平台->我的应用->应用信息->高级应用->授权设置->应用回调页”中找到。
Scope	Scope是OAuth2.0新版授权页的一个功能，通过scope平台将开放更多的微博核心功能给开发者，同时也加强用户隐私保护，提升用户体验，用户在新OAuth2.0授权页中有权利选择赋予应用的功能。
AccessToken	表明用户身份的Token，通过SSO授权后可以获取
Oauth 2.0 Web授权	通过WebView进行授权，返回Token信息，无需安装微博客户端

名词	注解
SSO授权	通过唤起微博客户端进行身份授权，返回Token信息

2:集成前的准备

2.1申请APP_KEY

第三方需要接入微博SDK必须在微博开放平台上对应用进行注册，并获取APP_KEY,添加应用的授权回调页面（Redirect URI），详情请参考：微博移动接入平台

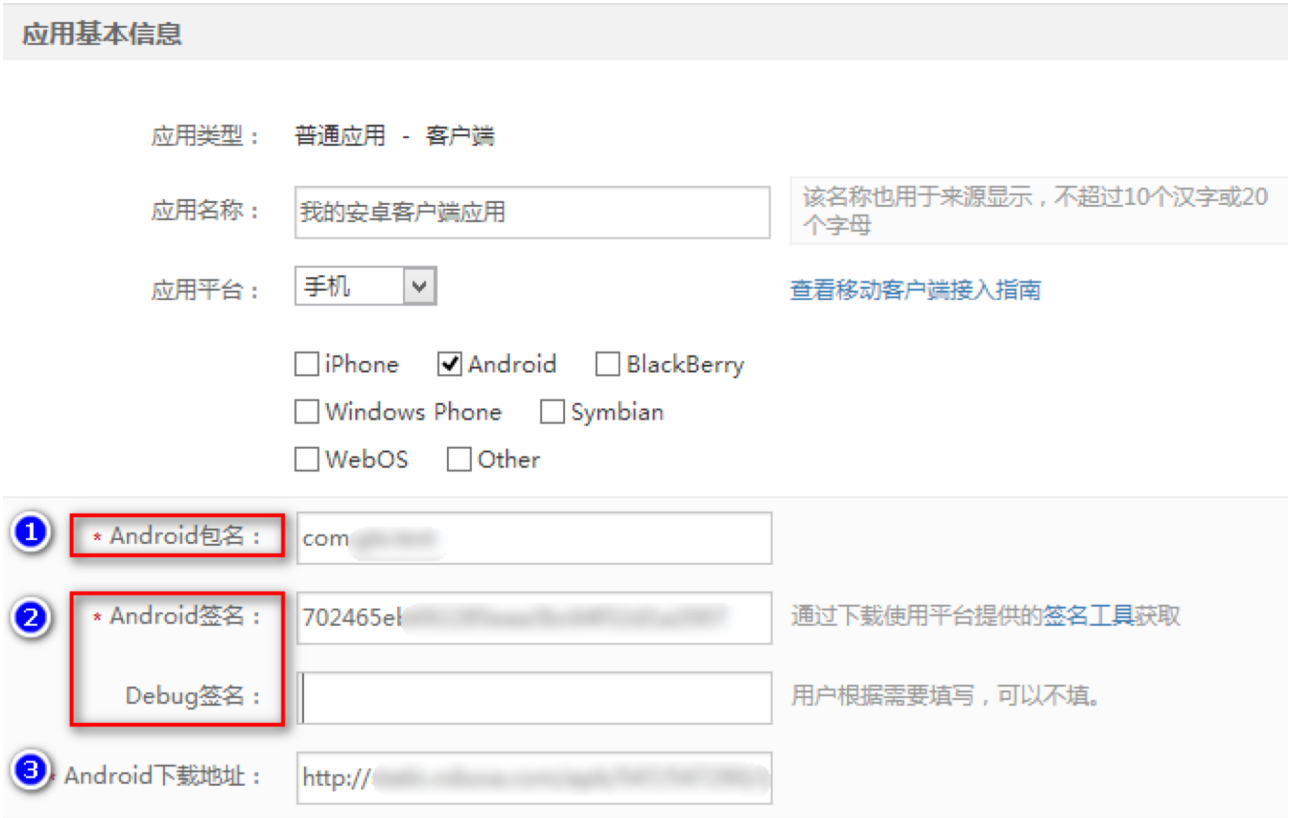
2.2注册程序包名和签名

集成SDK前，需要在微博开放平台上注册应用的包名和签名。

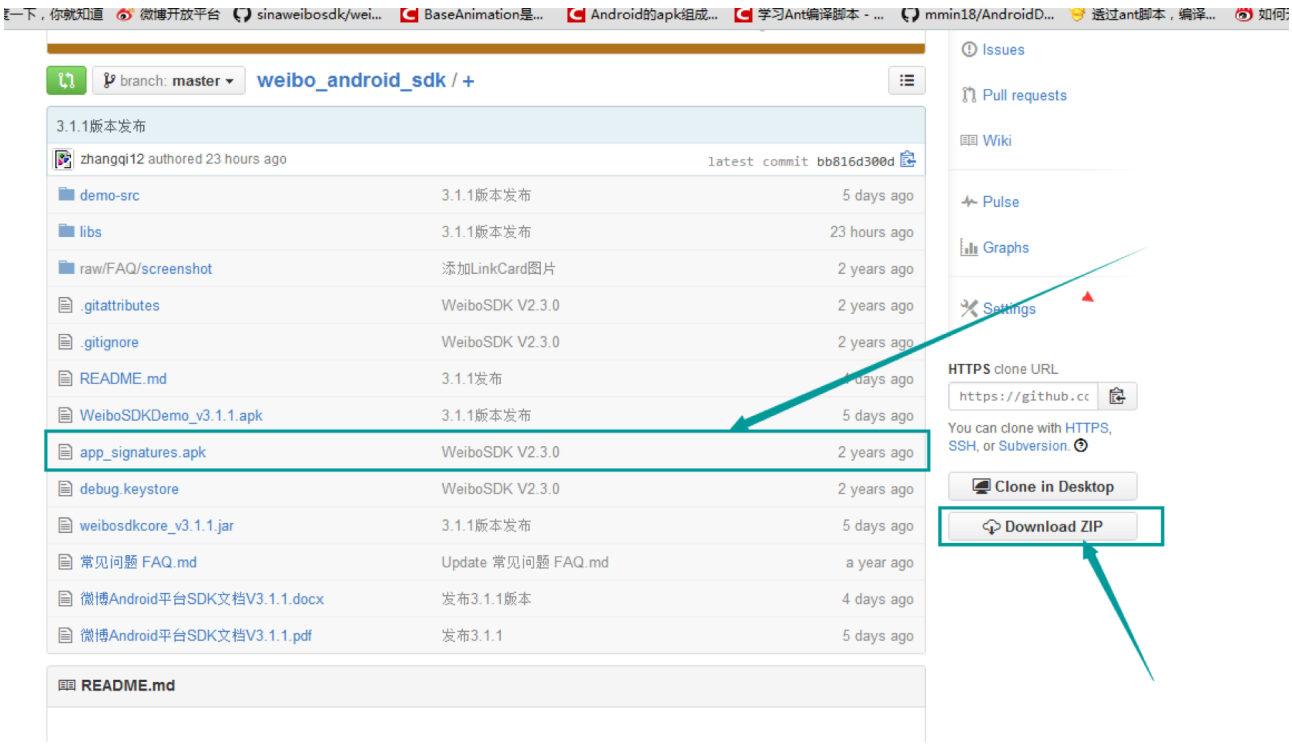
注意：包名和签名未注册，或者编译运行时的签名和注册签名不一致都可能导致无法授权(debug运行apk和发布时的apk)。

应用包名获取方式：工程主modules目录下build.gradle中的applicationId的值，或者AndroidManifest.xml中package节点的数据.

应用程序签名：该签名是通过官方提供的签名工具生成的MD5值（按照下图所示）



下载签名软件：下载地址



使用方式：首先要安装你已经签名的应用，然后再安装该工具，安装后，输入应用的包名，点击生成按钮，既可以获取MD5签名，如下图所示。

注意：要签名的应用程序必须安装到该设备上才能生成对应的MD5签名。如果你在开发平台上只填写的是发布版本的签名MD5,请确保你平时debug版本的签名和发布版本签名一致，否则授权不

能通过。

2接入方式

全新的SDK已经上传到中央仓库

```
buildscript {
    repositories {
        jcenter()
        mavenCentral()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:1.5.0'
        classpath 'com.jfrog.bintray.gradle:gradle-bintray-plugin:1.4'
        classpath 'com.github.dcendents:android-maven-gradle-plugin:1.3'
    }
}

allprojects {
    repositories {
        jcenter()
    }
}
task clean(type: Delete) {
    delete rootProject.buildDir
}
```

这里按照Android studio为例子，在项目根目录的build.gradle中设置中央仓库 jenter

在需要引入SDK的module目录的build.gradle中引入sdk-core依赖

dependencies {

compile 'com.sina.weibo.sdk:openDefault:1.0.0:openDefaultRelease@aar'

}

```
dependencies {
    compile 'org.jbundle.util.osgi.wrapped:org.jbundle.util.osgi.wrapped.org.apache.http.client:4.1.2'
    compile 'com.sina.weibo.sdk:openDefault:1.0.0:openDefaultRelease@aar'
}
```

点击同步按钮，等待SDK库下载完成。

也可以直接在github上下载微博 SDK demo，参考demo接入方案。

示例代码分析

1:认证授权

```
mAuthInfo = new AuthInfo(this, Constants.APP_KEY, Constants.REDIRECT_URL, Constants.SCOPE);
mSsoHandler = new SsoHandler(WBAuthActivity.this, mAuthInfo);
```

1)首先初始化AutoInfo，SsoHandler对象

AuthInfo维护了授权需要的基本信息，APP_KEY(开发平台生成的唯一key)、Redirect URI（授权回调）、SCOPE（需要请求的权限功能，默认参考demo中数据）。

SsoHandler是发起授权的核心类。

2）调用授权

SDK中有三种模式的授权方案：

1：AuthorizeClientSSo:只通过微博客户端进行授权

2：AuthorizeWeb:通过SDK自带的WebView打开H5页面进行授权

3：Authorize:如果安装了微博客户端通过客户端授权，如果没有通过Web方式授权

调用授权登录：

```
// SSO 授权, 仅客户端
findViewById(R.id.obtain_token_via_sso).setOnClickListener((v) → {
      mSsoHandler.authorizeClientSso(new AuthListener());
});

// SSO 授权, 仅Web
findViewById(R.id.obtain_token_via_web).setOnClickListener((v) → {
        mSsoHandler.authorizeWeb(new AuthListener());
});

// SSO 授权, ALL IN ONE   如果手机安装了微博客户端则使用客户端授权,没有则进行网页授权
findViewById(R.id.obtain_token_via_signature).setOnClickListener((v) → {
        mSsoHandler.authorize(new AuthListener());
});
```

AutoListener：授权结果回调，实现WeiboAuthListener接口

```
10
11  public interface WeiboAuthListener {
12      void onComplete(Bundle var1);
13
14      void onWeiboException(WeiboException var1);
15
16      void onCancel();
17  }
18
```

onComplete:授权成功回调,bundle中存储了授权相关信息，新版的微博SDK中，已经自动解析了bundle信息并存储为Oauth2AccessToken结构的数据，通过AccessTokenKeeper 的readAccesstoken方法获取Oauth2AccessToken数据。

onWeiboException：授权失败回调(更多相关失败信息，请参考文档结尾错误码)

onCancel：授权取消回调。

注意：要接收到授权的相关数据，必须在当前页面Activity或者Fragment的onActivityResult方法中添加SSOHandler的调用，如下所示

```
/**
 * 当 SSO 授权 Activity 退出时，该函数被调用。
 *
 * @see {@link Activity#onActivityResult}
 */
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);

    // SSO 授权回调
    // 重要：发起 SSO 登陆的 Activity 必须重写 onActivityResults
    if (mSsoHandler != null) {
        mSsoHandler.authorizeCallBack(requestCode, resultCode, data);
    }

}
```

3微博分享

3.1从第三方应用唤起微博客户端进行分享

3.1.1 准备工作：

开始微博分享前，在AndroidManifest.xml中为当前分享所在页面的Activity添加接收消息intent-filter，声明Action为:com.sina.weibo.sdk.action.ACTION_SDK_REQ_ACTIVITY

效果如下：

```
35          <!-- 分享 -->
36          <activity
37              android:name=".WBShareActivity"
38              android:configChanges="keyboardHidden|orientation"
39              android:screenOrientation="portrait" >
40              <intent-filter>
41                  <action android:name="com.sina.weibo.sdk.action.ACTION_SDK_REQ_ACTIVITY" />
42                  <category android:name="android.intent.category.DEFAULT" />
43              </intent-filter>
44          </activity>
```

在分享当前页面实现WeiboHandler.Response接口，效果如下：

```
public class WBShareActivity extends Activity implements OnClickListener, IWeiboHandler.Response {

    private static final String TAG = "WBShareActivity";
```

WeiboHandler.Response的时间样式如下：

```
    /**
     * 接收微客户端博请求的数据。
     * 当微博客户端唤起当前应用并进行分享时，该方法被调用。
     */
    @Override
    public void onResponse(BaseResponse baseResp) {
        if(baseResp!= null){
            switch (baseResp.errCode) {
            case WBConstants.ErrorCode.ERR_OK:
                Toast.makeText(this, "分享成功", Toast.LENGTH_LONG).show();
                break;
            case WBConstants.ErrorCode.ERR_CANCEL:
                Toast.makeText(this, "取消分享", Toast.LENGTH_LONG).show();
                break;
            case WBConstants.ErrorCode.ERR_FAIL:
                Toast.makeText(this,
                        "分享失败" + "Error Message: " + baseResp.errMsg,
                        Toast.LENGTH_LONG).show();
                break;
            }
        }
    }
```

实现Activity的onNewIntent方法：

```
/**
 * @see {@link Activity#onNewIntent}
 */
@Override
protected void onNewIntent(Intent intent) {
    super.onNewIntent(intent);

    // 从当前应用唤起微博并进行分享后，返回到当前应用时，需要在此处调用该函数
    // 来接收微博客户端返回的数据；执行成功，返回 true，并调用
    // {@link IWeiboHandler.Response#onResponse}；失败返回 false，不调用上述回调
    mWeiboShareAPI.handleWeiboResponse(intent, this);
}

/**
```

注意*：实现分享当前Activity必须准备以上四个步骤：

（1）修改AndroidManifest.xml

（2）Activity实现IWeiboHandler.Response接口

（3）Activity处理IWeiboHandler.Response的方法

（4）Activity处理onNewIntent方法

以上四步少一步都可能会导致分享无法实现。

**********以下是接入时可能会导致无法分享的一些说明**********

由于老版本的微博客户端分享存在一定逻辑缺陷，在没有用户登录的状态下进行第一次分享时可能导致第三方程序鉴定失败引起不能正常分享。这里特别说明下，在进行第一次分享的时候，可以先进行SSO授权登录，确保微博客户端处于账户登录状态，然后进行分享。

3.1.2实现分享

在分享页面启动时创建微博分享实例，并注册应用（请确保先注册，后分享，否则可能造成无法完成分享）

```
109         // 创建微博分享接口实例
110         mWeiboShareAPI = WeiboShareSDK.createWeiboAPI(this, Constants.APP_KEY);
111         // 注册第三方应用到微博客户端中，注册成功后该应用将显示在微博的应用列表中。
112         // 但该附件栏集成分享权限需要合作申请，详情请查看 Demo 提示
113         // NOTE：请务必提前注册，即界面初始化的时候或是应用程序初始化时，进行注册
114         mWeiboShareAPI.registerApp();
115
```

1：分享类型

微博sdk现在只提供两种分享类型，取消了以前的LinkCard模式，请以前以LinkCard模式的进行分享的使用图片加文字组合进行替换。

2：分享方式

微博sdk提供两种分享方式，分别是：仅通过微博客户端分享、如果安装微博客户端通过客户端分享，如果没有安装客户端通过H5页面进行分享。

```
/**
 * 获取分享的文本模板。
 *
 * @return 分享的文本模板
 */
private String getSharedText() {
    int formatId = "我正在使用微博客户端发博器分享文字";
    String format = getString(formatId);
    String text = format;
    if (mTextCheckbox.isChecked() || mImageCheckbox.isChecked()) {
        text = "@大屁老师，这是一个很漂亮的小狗，朕甚是喜欢-_-!!";
    }
    return text;
}

/**
 * 创建文本消息对象。
 * @return 文本消息对象。
 */
private TextObject getTextObj() {
    TextObject textObject = new TextObject();
    textObject.text = getSharedText();
    return textObject;
}

/**
 * 创建图片消息对象。
 *
 * @return 图片消息对象。
 */
private ImageObject getImageObj() {
    ImageObject imageObject = new ImageObject();
    BitmapDrawable bitmapDrawable = (BitmapDrawable) mImageView.getDrawable();
    //设置缩略图。 注意：最终压缩过的缩略图大小不得超过 32kb。
    Bitmap  bitmap = BitmapFactory.decodeResource(getResources(), R.drawable.ic_logo);
    imageObject.setImageObject(bitmap);
    return imageObject;
}
```

3：开始进行分享

如上图，想要进行分享，必须将需要分享的内容通过TextObject和ImageObject进行封装。

*分享的图片大小不能超过32kb

```
/**
 * 第三方应用发送请求消息到微博，唤起微博分享界面。
 */
private void sendMultiMessage(boolean hasText, boolean hasImage) {
    // 1. 初始化微博的分享消息
    WeiboMultiMessage weiboMessage = new WeiboMultiMessage();
    if (hasText) {
        weiboMessage.textObject = getTextObj();
    }
    if (hasImage) {
        weiboMessage.imageObject = getImageObj();
    }
    // 2. 初始化从第三方到微博的消息请求
    SendMultiMessageToWeiboRequest request = new SendMultiMessageToWeiboRequest();
    // 用transaction唯一标识一个请求
    request.transaction = String.valueOf(System.currentTimeMillis());
    request.multiMessage = weiboMessage;
    // 3. 发送请求消息到微博，唤起微博分享界面
    if (mShareType == SHARE_CLIENT) {
        mWeiboShareAPI.sendRequest(WBShareActivity.this, request);
    }
    else if (mShareType == SHARE_ALL_IN_ONE) {
        AuthInfo authInfo = new AuthInfo(this, Constants.APP_KEY, Constants.REDIRECT_URL, Constants.SCOPE);
        Oauth2AccessToken accessToken = AccessTokenKeeper.readAccessToken(getApplicationContext());
        String token = "";
        if (accessToken != null) {
            token = accessToken.getToken();
        }
        mWeiboShareAPI.sendRequest(this, request, authInfo, token, new WeiboAuthListener() {

            @Override
            public void onWeiboException( WeiboException arg0 ) {
            }

            @Override
            public void onComplete( Bundle bundle ) {
                // TODO Auto-generated method stub
                Oauth2AccessToken newToken = Oauth2AccessToken.parseAccessToken(bundle);
                AccessTokenKeeper.writeAccessToken(getApplicationContext(), newToken);
                Toast.makeText(getApplicationContext(), "onAuthorizeComplete token = " + newToken.getToken(), 0).show();
            }

            @Override
            public void onCancel() {
            }
        });
    }
}
/**
```

（1）初始化WeiboMultiMessage对象

（2）给WeiboMultiMessage填充分享内容（TextObject和ImageObject）

（3）初始化SendMultiMessageToWeiboRequest对象

（4）设置分享开始时的统计时间（必须设置否则可能导致分享失败）

（5）给request设置分享内容既WeiboMultiMessage实例

（6）根据自己业务需要选择只进行客户端分享还是All_In分享，通过WeiboShareApi调用sendRequest方法（两个参数方法为只使用客户端分享，5个参数方法为All_In分享）。

（7）在Activity的onResponse中进行分享结果处理

*：具体想知道如接入weibo sdk请参考weibo sdk 官方github地址中的demo:

https://github.com/sinaweibosdk/weibo_android_sdk

错误code集合：

code码	解释
C8998	开放平台填写的签名和程序签名不一致，检查下开放平台填写的是发布签名，运行程序为debug签名。