微博SDK-Android接入文档

SDK版本：13.10.0

接入须知

工程要求

属性	限制内容
targetSdkVersion	<= 33
ABI	支持目录：armeabi、armeabi-v7a、arm64-v8a
FileProvider	SDK中已包含FileProvider，暂不支持自定义FileProvider
分享目录	目录限制：getExternalFilesDir、getExternalCacheDir目录及子目录；目前不支持网络路径

业务限制

属性	限制内容
Web分享	Web分享只支持分享文字
多图分享	目前支持图片格式：jpg、png、gif 单图分享时同样建议使用多图分享API
视频分享	目前支持视频格式：mp4
超话分享	非完全开放类型，若有需要，请找微博商务洽谈，取得白名单权限后再进行该功能开发

工程配置

集成SDK（请选择其中一种方式集成）

- 方式一，使用远程依赖

    1）在project根目录的build.gradle文件中添加依赖配置

    ```
    allprojects {
     repositories {
       mavenCentral()
       ……
        }
    }
    ```

    2）在module的build.gradle文件中添加依赖和属性配置

    ```
    android {
     ……
     defaultConfig {
       ndk {
            //根据项目ABI定制
         abiFilters 'armeabi'//,'armeabi-v7a','arm64-v8a'
       }
      }
    }

    dependencies {
     implementation 'io.github.sinaweibosdk:core:13.10.0@aar'
    }
    ```

- 方式二，使用aar本地依赖

    1. [打开最新文档](#)下载需要依赖的SDK版本的aar，[将AAR文件作为工程依赖](#)
    2. 在module的build.gradle文件中添加依赖和属性配置abi

    ```
    android {
        ……
        defaultConfig {
            ndk {
            //根据项目ABI定制
                abiFilters 'armeabi'//,'armeabi-v7a','arm64-v8a'
            }
        }
    }
    ```

混淆

```
-keep public class com.sina.weibo.sdk.**{*;}
```

接入代码

初始化SDK

```java
public class DemoApplication extends Application {
    private static final String APP_KY = "在微博开发平台为应用申请的App Key";
    private static final String REDIRECT_URL = "在微博开放平台设置的授权回调页";
    private static final String SCOPE = "在微博开放平台为应用申请的高级权限";

    @Override
    protected void onCreate() {
        super.onCreate();
        initSdk();
    }

    /**
     * 初始化sdk。
     * 理论上使用前只要初始化一次即可，具体分享及授权登录时将不需要再次初始化。
     */
    private void initSdk() {
        AuthInfo authInfo = new AuthInfo(this, APP_KY, REDIRECT_URL, SCOPE);

        mWBAPI = WBAPIFactory.createWBAPI(this); // 传Context即可，不再依赖于Activity

        mWBAPI.registerApp(this, authInfo, new SdkListener() {
            @Override
            public void onInitSuccess() {
                // SDK初始化成功回调，成功一次后再次初始化将不再有任何回调
            }

            @Override
            public void onInitFailure(Exception e) {
                // SDK初始化失败回调
            }
        });
    }
}
```

使用微博授权

1. 在Activity中调用授权API（请选择其中一种方式）

```java
//方式一，优先客户端授权，未安装时使用Web授权
private void startAuth() {
    mWBAPI.authorize(this, new WbAuthListener() {
        @Override
        public void onComplete(Oauth2AccessToken token) {
            Toast.makeText(MainActivity.this, "微博授权成功",
Toast.LENGTH_SHORT).show();
        }

        @Override
        public void onError(UiError error) {
            Toast.makeText(MainActivity.this, "微博授权出错",
Toast.LENGTH_SHORT).show();
        }

        @Override
        public void onCancel() {
            Toast.makeText(MainActivity.this, "微博授权取消",
Toast.LENGTH_SHORT).show();
        }
    });
}
//方式二，指定通过客户端授权操作
private void startClientAuth() {
    mWBAPI.authorizeClient(this, new WbAuthListener() {
        @Override
        public void onComplete(Oauth2AccessToken token) {
            Toast.makeText(MainActivity.this, "微博授权成功",
Toast.LENGTH_SHORT).show();
        }

        @Override
        public void onError(UiError error) {
            Toast.makeText(MainActivity.this, "微博授权出错",
Toast.LENGTH_SHORT).show();
        }

        @Override
        public void onCancel() {
            Toast.makeText(MainActivity.this, "微博授权取消",
Toast.LENGTH_SHORT).show();
        }
    });
}
```

```
    //方式三，指定通过网页（Web）授权操作
    private void startWebAuth() {
        mWBAPI.authorizeWeb(this, new WbAuthListener() {
            @Override
            public void onComplete(Oauth2AccessToken token) {
                Toast.makeText(MainActivity.this, "微博授权成功",
Toast.LENGTH_SHORT).show();
            }

            @Override
            public void onError(UiError error) {
                Toast.makeText(MainActivity.this, "微博授权出错:" +
error.errorDetail, Toast.LENGTH_SHORT).show();
            }

            @Override
            public void onCancel() {
                Toast.makeText(MainActivity.this, "微博授权取消",
Toast.LENGTH_SHORT).show();
            }
        });
    }
```

2. 重写Activity的onActivityResult()方法，示例如下

```
    @Override
    protected void onActivityResult(int requestCode, int resultCode, @Nullable
Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        if (mWBAPI != null) {
            mWBAPI.authorizeCallback(requestCode, resultCode, data);
        }
    }
```

> 注：这里是设置授权的回调，如果不重写，将收不到授权结果。

3. 错误码定位：OAuth2.0 错误码

分享内容到微博

1. 实现回调接口，代码示例如下

```
    private static class ShareCallback implements WbShareCallback {
        @Override
        public void onComplete() {
            Toast.makeText(ShareActivity.this, "分享成功",
Toast.LENGTH_SHORT).show();
```

```
        }

        @Override
        public void onError(UiError error) {
            Toast.makeText(ShareActivity.this, "分享失败:" + error.errorMessage,
Toast.LENGTH_SHORT).show();
        }

        @Override
        public void onCancel() {
            Toast.makeText(ShareActivity.this, "分享取消",
Toast.LENGTH_SHORT).show();
        }
    }
```

2. 重写Activity的onActivityResult()方法，示例如下

```
    @Override
    protected void onActivityResult(int requestCode, int resultCode, @Nullable
Intent data) {
        super.onActivityResult(requestCode, resultCode, data);
        if (mWBAPI != null) {
            mWBAPI.doResultIntent(data, this);
        }
    }
```

> 注：和授权登录同理，这里是设置分享的回调，如果不重写，将收不到分享结果。

3. 微博分享示例代码如下

```
    private void doWeiboShare() {
        WeiboMultiMessage message = new WeiboMultiMessage();

        TextObject textObject = new TextObject();
        String text = "我正在使用微博客户端发博器分享文字。";

        // 分享文字
        if (mShareText.isChecked()) {
            text = "这里设置您要分享的内容！";
            textObject.text = text;
            message.textObject = textObject;
        }

        // 分享图片
        if (mShareImage.isChecked()) {
            ImageObject imageObject = new ImageObject();
            Bitmap bitmap = BitmapFactory.decodeResource(getResources(),
R.drawable.share_image);
```

```java
            imageObject.setImageData(bitmap);
            message.imageObject = imageObject;
        }

        // 分享网页
        if (mShareUrl.isChecked()) {
            WebpageObject webObject = new WebpageObject();
            webObject.identify = UUID.randomUUID().toString();
            webObject.title = "标题";
            webObject.description = "描述";
            Bitmap bitmap = BitmapFactory.decodeResource(getResources(),
R.drawable.ic_logo);
            ByteArrayOutputStream os = null;
            try {
                os = new ByteArrayOutputStream();
                bitmap.compress(Bitmap.CompressFormat.JPEG, 85, os);
                webObject.thumbData = os.toByteArray();
            } catch (Exception e) {
                e.printStackTrace();
            } finally {
                try {
                    if (os != null) {
                        os.close();
                    }
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
            webObject.actionUrl = "https://weibo.com";
            webObject.defaultText = "分享网页";
            message.mediaObject = webObject;
        }

        if (mShareMultiImage.isChecked()) {
            // 分享多图
            MultiImageObject multiImageObject = new MultiImageObject();
            ArrayList<Uri> list = new ArrayList<>();
            File externalFilesDir = getExternalFilesDir(null);
            File externalCacheDir = getExternalCacheDir();
            if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.N) {
                String authority = this.getPackageName() + ".fileprovider";
                list.add(FileProvider.getUriForFile(this, authority, new
File(externalFilesDir, "aaa.png")));
                list.add(FileProvider.getUriForFile(this, authority, new
File(externalCacheDir, "bbb.jpg")));
                list.add(FileProvider.getUriForFile(this, authority, new
File(externalFilesDir + "/file_images/ccc.JPG")));
                list.add(FileProvider.getUriForFile(this, authority, new
File(externalFilesDir + "/cache_images/ddd.jpg")));
```

```java
                    list.add(FileProvider.getUriForFile(this, authority, new
File(externalCacheDir + "/file_images/eee.jpg")));
                    list.add(FileProvider.getUriForFile(this, authority, new
File(externalFilesDir + "/file_images/fff.jpg")));
                    list.add(FileProvider.getUriForFile(this, authority, new
File(externalFilesDir+"/file_images/ggg.JPG")));
                    list.add(FileProvider.getUriForFile(this, authority, new
File(externalFilesDir + "/file_images/hhhh.jpg")));
                } else {
                    list.add(Uri.fromFile(new File(externalFilesDir, "aaa.png")));
                    list.add(Uri.fromFile(new File(externalCacheDir, "bbb.jpg")));
                    list.add(Uri.fromFile(new File(externalFilesDir +
"/file_images/ccc.JPG")));
                    list.add(Uri.fromFile(new File(externalFilesDir +
"/cache_images/ddd.jpg")));
                    list.add(Uri.fromFile(new File(externalCacheDir +
"/file_images/eee.jpg")));
                    list.add(Uri.fromFile(new File(externalFilesDir +
"/file_images/fff.jpg")));
                    list.add(Uri.fromFile(new
File(externalFilesDir+"/file_images/ggg.JPG")));
                    list.add(Uri.fromFile(new File(externalFilesDir +
"/file_images/hhhh.jpg")));
                }
                multiImageObject.imageList = list;
                message.multiImageObject = multiImageObject;
            }

            if (mShareVideo.isChecked()) {
                // 分享视频
                VideoSourceObject videoObject = new VideoSourceObject();
                if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.N) {
                    String filePath = getExternalFilesDir(null) +
"/file_video/eeee.mp4";
                    File videoFile = new File(filePath);
                    if (!videoFile.getParentFile().exists()) {
                        videoFile.getParentFile().mkdir();
                    }
                    videoObject.videoPath = FileProvider.getUriForFile(this,
this.getPackageName() + ".fileprovider", videoFile);
                } else {
                    videoObject.videoPath = Uri.fromFile(new
File(getExternalFilesDir(null) + "/file_video/eeee.mp4"));
                }
                message.videoSourceObject = videoObject;
            }

            if (mShareSuperGroup.isChecked()) {
                SuperGroupObject superGroupObject = new SuperGroupObject();
```

```java
        String sgName =
((EditText)findViewById(R.id.edit_sg_name)).getText().toString().trim();
            if (TextUtils.isEmpty(sgName)) {
                sgName = this.getString(R.string.demo_sg_name);
            }
            superGroupObject.sgName = sgName;
            String sgSection =
((EditText)findViewById(R.id.edit_sg_section)).getText().toString().trim();
            superGroupObject.secName = sgSection;
            String sgExt =
((EditText)findViewById(R.id.edit_sg_ext)).getText().toString().trim();
            superGroupObject.sgExtParam = sgExt;
            message.superGroupObject = superGroupObject;
        }

        mWBAPI.shareMessage(this, message, mShareClientOnly.isChecked());
    }
```