

oppor 开放平台

产品 活动 社区 学堂 文档中心

推送服务

Q

管理中心 中国站

登录

注册

全部文档

请输入您想搜索的文档名称

搜索

账号服务

Q 搜索本目录下的文档

欢太账号OAuth授权SDK介绍

欢太账号OAuth授权SDK开启指南

欢太账号OAuth授权SDK合规使...

欢太账号OAuth授权SDK接入指导

欢太账号OAuth授权SDK服务端API

联系我们

欢太账号OAuth授权SDK服务端API

一、使用入门

序号	步骤	说明
1	注册开平账号并创建应用	注册成为OPPO企业开发者, 开通账号服务
2	获取AccessToken	应用接入帐号SDK, 获取到Access Token。
3	调用账号API (可选)	应用在得到用户授权后的Access Token后, 可请求对应账号API接口及资源。
4	刷新Access Token (可选)	应用可根据Refresh Token获取新的Access Token。

二、开发准备

快速接入

第 1 步: 注册并通过认证

首先访问OPPO开放平台, 注册成为OPPO企业开发者, 详情请参考 [《OPPO企业开发者账号注册流程》](#)

第 2 步: 创建应用

然后登录OPPO企业开发者账号, 并创建应用, 详情请参考 [《应用接入流程》](#)

第 3 步: 申请开通账号服务

1、登录OPPO开放平台, 依次选择“产品”-“账号服务”进入账号服务详情页

2、点击“申请接入”进入应用申请页面

3、在申请界面可以看到已开启服务应用和未开启服务应用

1) 已开启服务: 已申请账号服务权限并通过的应用

2) 未开启服务: 可申请账号服务权限的应用

4、在未开启服务的应用中选择需要申请账号服务权限的应用, 进入账号服务申请界面。

5、如应用账号未激活, 需要点击“激活账号”进行激活操作; 如应用已激活可忽略此步骤

6、在账号服务能力项中, 依次点击openid、公开信息、unionid右边的“申请开通”按钮, 提交申请, 即可开通成功。

三、接口说明

服务端接口禁止客户端调用, 只支持服务端调用

3.1、环境域名

环境	域名
国内测试	http://uc-oauth-test.wanyol.com
国内生产	https://api.uc.qqomobile.com
海外生产	https://uc-api-glbeidaqqomobile.com (暂未上线)

3.2、获取访问令牌

请求说明

通过授权码获取访问令牌

请求URI	/oauth2/token/token-code
HTTP Method	POST
数据格式	application/json

请求示例

```
{
  "appKey": "9f*****91",
  "appSecret": "126*****37",
  "code": "HAT_7d*****mgvc96XfvnRijnzLtzQWkkahAx9_*****",
}
```

参数说明:

参数	类型	是否必需	说明
----	----	------	----

文档是否有帮助解决问题?

有帮助

无帮助

咨询

合作

评价

https://open.oppomobile.com/documentation/page/info?id=10155

1/7

oppor 开放平台

产品 活动 社区 学堂 文档中心

Q

管理中心 中国站 登录 注册

全部文档

Q

账号服务

三

Q

欢太账号OAuth授权SDK介绍

欢太账号OAuth授权SDK开启指南

欢太账号OAuth授权SDK合规使...

欢太账号OAuth授权SDK接入指导

欢太账号OAuth授权SDK服务端API

联系我们

```
{
  "success": false,
  "error": null,
  "data": {
    "accessToken": "ACCESS_*****",
    "refreshToken": "REFRESH_*****",
    "openId": "*****",
    "scope": "name",
    "expireIn": 1024
  }
}
```

异常响应示例

```
{
  "success": false,
  "error": {
    "code": "2020003",
    "message": "invalid_client"
  },
  "data": null
}
```

返回说明

参数	类型	是否必需	说明
success	布尔值	是	请求是否成功
error	对象	否	请求错误的原因
data	对象	否	请求成功返回的数据

data对象结构

参数	类型	是否必需	说明
accessToken	字符串	是	授权服务器颁发的访问令牌
refreshToken	字符串	是	刷新令牌，在访问令牌过期后，可以用来获取新的访问令牌以及刷新令牌
openId	字符串	是	用户的身份标识
scope	字符串	是	用户同意授权的范围
expireIn	数字	是	以秒为单位的访问令牌生命周期

error对象结构

参数	类型	是否必需	说明
code	字符串	是	错误码
message	字符串	是	错误说明

3.3、刷新访问令牌

请求说明

通过刷新令牌获取新的访问令牌（若访问令牌未过期，则返回原来的访问令牌，反之刷新访问。若刷新令牌过期则需重新授权）

请求URI	/oauth2/token/refresh-token
HTTP Method	POST

文档是否有帮助解决问题？

有帮助 无帮助

咨询

合作

评价

https://open.oppomobile.com/documentation/page/info?id=10155

2/7

oppor 开放平台

产品 活动 社区 学堂 文档中心

Q

管理中心 中国站 登录 注册

全部文档

Q

账号服务

三

Q

欢太账号OAuth授权SDK介绍

欢太账号OAuth授权SDK开启指南

欢太账号OAuth授权SDK合规使...

欢太账号OAuth授权SDK接入指导

欢太账号OAuth授权SDK服务端API

联系我们

参数说明:

参数	类型	是否必需	说明
appKey	字符串	是	开平申请的应用appKey
accessToken	字符串	是	授权服务器颁发的访问令牌
refreshToken	字符串	是	刷新令牌, 在访问令牌过期后, 可以用来获取新的访问令牌以及刷新令牌

返回示例

```
{
  "success": false,
  "error": null,
  "data": {
    "accessToken": "ACCESS_*****",
    "refreshToken": "REFRESH_*****",
    "expireIn": 1024
  }
}
```

返回说明

参数	类型	是否必需	说明
success	布尔值	是	请求是否成功
error	对象	否	请求错误的原因
data	对象	否	请求成功返回的数据

data对象结构

参数	类型	是否必需	说明
accessToken	字符串	是	授权服务器颁发的访问令牌
refreshToken	字符串	是	刷新令牌, 在访问令牌过期后, 可以用来获取新的访问令牌以及刷新令牌
expireIn	数字	是	以秒为单位的访问令牌生命周期

3.4、获取用户公开信息

请求说明

请求获取用户数据

请求URI	/oauth2/userinfo/profile (用户公开信息 需要的scope为profile)
HTTP Method	POST
数据格式	application/json

请求示例

```
{
  "appKey": "9f*****f9f",
  "openId": "*****",
  "accessToken": "ACCESS_*****",
}
```

参数说明:

文档是否有帮助解决问题?

有帮助 无帮助

oppor 开放平台

产品 活动 社区 学堂 文档中心

Q

管理中心 中国站 登录 注册

全部文档

Q

账号服务

Q

欢太账号OAuth授权SDK介绍

欢太账号OAuth授权SDK开启指南

欢太账号OAuth授权SDK合规使...

欢太账号OAuth授权SDK接入指导

欢太账号OAuth授权SDK服务端API

联系我们

```
{
  "success": false,
  "error": null,
  "data": {
    "nickname": "用户0*****10",
    "avatars": {
      "default": "https://****_*****.com/***/****/1****4.png"
    }
  }
}
```

返回说明

参数	类型	是否必需	说明
success	布尔值	是	请求是否成功
error	对象	否	请求错误的原因
data	对象	否	请求成功返回的数据

data对象结构

参数	类型	是否必需	说明
nickname	字符串	是	用户昵称
avatars	哈希表	是	用户头像

3.5、获取用户手机号

请求说明

请求获取用户数据

请求URI	/oauth2/userinfo/phone (用户手机号 需要的scope为phone)
HTTP Method	POST
数据格式	application/json

请求示例

```
{
  "appKey": "38*****ff",
  "openId": "*****",
  "accessToken": "ACCESS_***** _*****",
}
```

参数说明：

参数	类型	是否必需	说明
appKey	字符串	是	开平申请的应用appKey
openId	字符串	是	用户身份标识
accessToken	字符串	是	授权服务器颁发的访问令牌

返回示例

```
{
  "success": false,
  "error": null,
  "data": {
    "countryCallingCode": "+86",
    "mobile": "*****"
  }
}
```

文档是否有帮助解决问题？

有帮助

无帮助

咨询

合作

评价

https://open.oppomobile.com/documentation/page/info?id=10155

4/7

账号服务

☰

Q

欢太账号OAuth授权SDK介绍

欢太账号OAuth授权SDK开启指南

欢太账号OAuth授权SDK合规使...

欢太账号OAuth授权SDK接入指导

欢太账号OAuth授权SDK服务端API

联系我们

data对象结构

参数	类型	是否必需	说明
countryCallingCode	字符串	是	国家区号
mobile	字符串	是	用户手机号（密文，需要使用appSecret解密）解密代码见附录

3.6、获取用户实名信息

请求说明

请求获取用户数据

请求URI	/oauth2/userinfo/realname (用户实名信息 需要的scope为realname)
HTTP Method	POST
数据格式	application/json

请求示例

```
{
  "appKey": "38*****6ff",
  "openId": "*****",
  "accessToken": "ACCESS_***** _*****",
}
```

参数说明：

参数	类型	是否必需	说明
appKey	字符串	是	开平申请的应用appKey
openId	字符串	是	用户身份标识
accessToken	字符串	是	授权服务器颁发的访问令牌

返回示例

```
{
  "success": false,
  "error": null,
  "data": {
    "idNumber": "*****",
    "realName": "****"
  }
}
```

返回说明

参数	类型	是否必需	说明
success	布尔值	是	请求是否成功
error	对象	否	请求错误的原因
data	对象	否	请求成功返回的数据

data对象结构

参数	类型	是否必需	说明
realName	字符串	是	用户实名（密文，需要使用appSecret解密）解密代码见附录
idNumber	字符串	是	用户身份证（密文，需要使用appSecret解密）解密代码见附录

四、错误码

文档是否有帮助解决问题？

👍 有帮助

👎 无帮助

oppor 开放平台

产品 活动 社区 学堂 文档中心

Q

管理中心 中国站 登录 注册

全部文档

Q

账号服务

三

Q

欢太账号OAuth授权SDK介绍

欢太账号OAuth授权SDK开启指南

欢太账号OAuth授权SDK合规使...

欢太账号OAuth授权SDK接入指导

欢太账号OAuth授权SDK服务端API

联系我们

1117001	环境不安全	appid、业务pkgName和业务的包签名不匹配，或者风控检测环境不安全
4041	accessToken失效	需要刷新token
4042	refreshToken失效	需要重新授权
2020002	authenticate_failed	用户或应用身份验证失败。
2020003	invalid_client	未知的应用
2020004	invalid_grant	提供的授权码无效
2020005	invalid_request	请求缺少必需的参数
2020006	invalid_scope	请求的授权范围无效、未知或格式不正确
2020008	invalid_token	accessToken过期或无效
2020016	user_phone_no_found	用户未绑定手机
2020017	real_name_info_no_found	用户未实名

五、后端加密算法

```
import java.nio.charset.StandardCharsets;
import java.security.SecureRandom;
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.spec.SecretKeySpec;
import org.apache.commons.codec.binary.Base64;
import org.apache.commons.lang3.StringUtils;

public class AESUtils {
    public static String base64Encode(byte[] bytes) {
        return Base64.encodeBase64String(bytes);
    }
    /**
     * base 64 decode
     *
     * @param base64Code 待解码的base 64 code
     * @return 解码后的byte[]
     */
    public static byte[] base64Decode(String base64Code) throws Exception {
        return StringUtils.isEmpty(base64Code) ? null : org.apache.commons.codec.binary.Base64.decodeBase64(base64Code);
    }
    /**
     * AES加密
     *
     * @param content 待加密的内容
     * @param encryptKey 加密密钥
     * @return 加密后的byte[]
     */
    public static byte[] aesEncryptToBytes(String content, String encryptKey) throws Exception {
        KeyGenerator kgen = KeyGenerator.getInstance("AES");
        SecureRandom random = SecureRandom.getInstance("SHA1PRNG");
        random.setSeed(encryptKey.getBytes("utf-8"));
        kgen.init(128, random);
        Cipher cipher = Cipher.getInstance("AES");
        cipher.init(Cipher.ENCRYPT_MODE, new SecretKeySpec(kgen.generateKey().getEncoded(), "AES"));
        return cipher.doFinal(content.getBytes("utf-8"));
    }
    /**
     * AES加密为base 64 code
     */
}
```

文档是否有帮助解决问题?

有帮助 无帮助

咨询

合作

评价

oppor 开放平台

产品 活动 社区 学堂 文档中心

Q

管理中心 中国站 登录 注册

全部文档

Q

账号服务

Q

欢太账号OAuth授权SDK介绍

欢太账号OAuth授权SDK开启指南

欢太账号OAuth授权SDK合规使...

欢太账号OAuth授权SDK接入指导

欢太账号OAuth授权SDK服务端API

联系我们

```

*
* @param encryptBytes 待解密的byte[]
* @param decryptKey 解密密钥
* @return 解密后的String
*/
public static String aesDecryptByBytes(byte[] encryptBytes, String decryptKey) throws Exception
{
    KeyGenerator kgen = KeyGenerator.getInstance("AES");
    SecureRandom random = SecureRandom.getInstance("SHA1PRNG");
    random.setSeed(decryptKey.getBytes("utf-8"));
    kgen.init(128, random);
    Cipher cipher = Cipher.getInstance("AES");
    cipher.init(Cipher.DECRYPT_MODE, new SecretKeySpec(kgen.generateKey().getEncoded(), "AES"));
    byte[] decryptBytes = cipher.doFinal(encryptBytes);
    return new String(decryptBytes, StandardCharsets.UTF_8);
}

/**
 * 将base 64 code AES解密
 *
 * @param encryptStr 待解密的base 64 code
 * @param decryptKey 解密密钥
 * @return 解密后的string
 */
public static String aesDecrypt(String encryptStr, String decryptKey) throws Exception {
    return StringUtils.isEmpty(encryptStr) ? null : aesDecryptByBytes(base64Decode(encryptStr), decryptKey);
}
}

```

文档是否有帮助解决问题?

有帮助

无帮助

咨询

合作

评价