

[Holmes] Google OAuth2 登陆鉴权

- [修订历史](#)
- [背景](#)
- [配置流程](#)
 - [针对 Web 应用，创建一个新项目的流程](#)
 - [代码实现](#)
- [其他](#)

修订历史

版本	修订日期	修订者	修订内容
1	2021.08.11	haoyang.huang@shopee.com	添加方案

背景

风控前端登陆基本都是采用谷歌登陆，Holmes 用到的 API 是 [Google OAuth2](#)，鉴权的配置需要在 [Google API Console](#) 创建。

API Console 是基于谷歌邮箱的，所以需要用到公共邮箱：[公共邮箱账号密码](#)

配置流程

用公共邮箱登陆 **API Console**，可以看到已经创建的项目配置。Holmes 的存放在 Risk-Holmes 项目。

Google Cloud Platform

选择项目

搜索产品和资源

API 和服务

信息中心

信息中心

库

凭据

OAuth 同意屏幕

网络验证

页面使用量协议

要查看此页面，请选择一个项目。

选择项目 创建项目

选择近期的项目

Risk-Holmes

项目 ID: risk-holmes

组织: shopeemobile.com

3分钟前访问过

dba jenkins google-login

项目 ID: utopian-surface-302208

组织: shopeemobile.com

28分钟前访问过

credit-admin-th

项目 ID: credit-admin-th

组织: shopeemobile.com

1小时前访问过

Quickstart

项目 ID: quickstart-1564127748631

组织: shopeemobile.com

1小时前访问过

auth

项目 ID: auth-293808

组织: shopeemobile.com

2021年7月26日访问过

My Project 26441

项目 ID: deep-wave-284106

组织: shopeemobile.com

2021年7月14日访问过

针对 Web 应用，创建一个新项目的流程

1. 点击 **创建项目**，填写项目名，选择组织(目前只有 shopeemobile.com)，进行创建。



Google Cloud Platform

新建项目

项目名称 *

My Project 53852

项目 ID: **stately-turbine-322610**。它日后无法更改。 [修改](#)

组织 *

shopeemobile.com

选择要关联到项目的组织。做出选择后，您就无法再更改了。

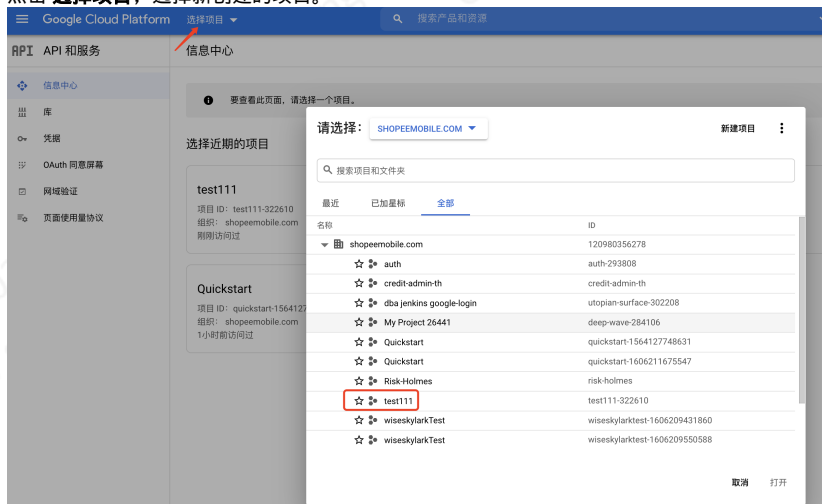
位置 *

shopeemobile.com [浏览](#)

父级组织或文件夹

[创建](#) [取消](#)

2. 点击 **选择项目**，选择新创建的项目。



Google Cloud Platform

信息中心

选择近期的项目

test111

项目 ID: test111-322610

组织: shopeemobile.com

刚刚访问过

Quickstart

项目 ID: quickstart-156412

组织: shopeemobile.com

1小时前访问过

请选择: SHOPEEMOBILE.COM

搜索项目和文件夹

名称	ID
shopeemobile.com	120980356278
auth	auth-293808
credit-admin-th	credit-admin-th
dba-jenkins-google-login	utopian-surface-302208
My Project 26441	deep-wave-284106
Quickstart	quickstart-1564127748631
Quickstart	quickstart-1606211675547
Risk-Holmes	risk-holmes
test111	test111-322610
wiseskylarkTest	wiseskylarkTest-1606209431860
wiseskylarkTest	wiseskylarkTest-1606209550588

[取消](#) [打开](#)

3. 进入 **同意屏幕** 页面，选择 **外部** 然后创建。内部只能组织邮箱能登陆。



Google Cloud Platform

test111

搜索产品和资源

API API 和服务

OAuth 同意屏幕

选择您想要配置和注册应用（包括目标用户）的方式。您只能将一个应用与您的项目关联。

User Type

☐ 内部

☒ 外部

仅供您的组织内的用户使用。您不需要提交应用进行验证。 [了解详情](#)

可供拥有 Google 帐号的任何测试用户使用。您的应用将以测试模式启动，并且只能供添加到测试用户列表中的用户使用。一旦您的应用准备就绪可以推送至生产环境，您可能需要验证应用。 [了解详情](#)

[创建](#)

[告诉我们您对 OAuth 流程的看法](#)

4. 进入凭据页面，点击 **创建凭据**，选择 **OAuth 客户端 ID**。



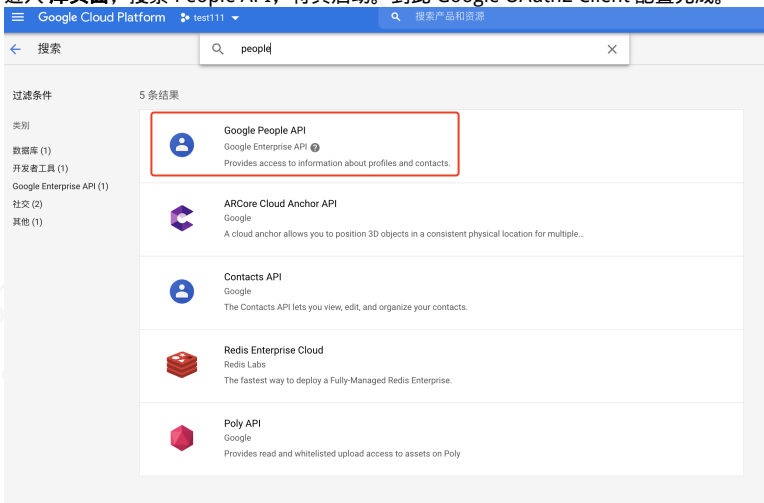
5. 创建 **OAuth2 Client**，选择应用类型 **Web 应用**，添加授权的域名。



6. 点击 创建, 生成 clientID、clientSecret。



7. 进入 库页面, 搜索 People API, 将其启动。到此 Google OAuth2 Client 配置完成。



代码实现

- 前端代码实现
chrome 浏览器内置 gapi 实例, 可用于创建 OAuth2 Client。通过配置生成的 clientID 获取授权码 code。

创建 OAuth2 Client 实例

```
/**
 * Google
 * @async
 * @throws
 */
export const initGoogleAuthorization = () => {
  if (typeof gapi === 'undefined' || !gapi) {
    const error = VUE.$ts('gapi.unload')
    callErrorNotice('Google Sign-In OAuth', error)
    throw new Error(error)
  }
  return new Promise((resolve, reject) => {
    gapi.load('auth2', () => {
      gapi.auth2.init({ client_id: OAUTH2_CLIENT_ID }).then(
        (auth2) => {
          LABEL = true
          resolve(auth2)
        },
        (err) => {
            Sentry.captureMessage('GOOGLE_INIT_ERR', {
```

```

        extra: {
          message: err
        },
        tags: {
          type: 'GOOGLE_INIT_ERR'
        }
      })
      callErrorNotice(err.error, err.details)
      reject(err)
    }
  }
}

/**
 * GoogleAuth
 * @async
 */
export const getGoogleAuthorization = () => {
  if (LABEL) {
    return Promise.resolve(gapi.auth2.getAuthInstance())
  }
  return initGoogleAuthorization()
}

/**
 * Google
 * @async
 * @throws
 */
export const getAuthorizationCode = async () => {
  // async/await TypeScript
  const auth2 = (await getGoogleAuthorization()) as gapi.auth2.GoogleAuth
  return auth2
    .grantOfflineAccess()
    .then((result) => result.code)
    .catch((err) => {
      callErrorNotice(err.error, err.details)
      return ''
    })
}

```

- Node端实现

Node端通过 **OAuth2Client** 库创建 OAuth2 Client。

通过 **getToken** 方法可以鉴定前端传来的 **code** 是否合法。如果前端和Node端 clientID 不匹配，调用 **getToken** 出错。

通过 **setCredentials** 方法用于设置 token 有效期。

通过 **oAuth2Client.request({ url: GAXIOS_URL })** 调用 Google People API 获取谷歌账户信息。

解析授权码进行鉴权

```

import { OAuth2Client } from 'google-auth-library'

export const getOAuth2Client = async (redirectUri: string, code?: string, token?: string) => {
  const oAuth2Client = new OAuth2Client(OAUTH2_CLIENT_ID, OAUTH2_CLIENT_SECRET, redirectUri)
  if (code) {
    const r = await oAuth2Client.getToken(code)
    oAuth2Client.setCredentials({
      ...r.tokens,
      expiry_date: Date.now() + EXPIRATION_TIME
    })
  } else if (token) {
    oAuth2Client.setCredentials({
      refresh_token: token,
      expiry_date: Date.now() + EXPIRATION_TIME
    })
  }
  return oAuth2Client
}

```

```
}

export const getUserProfileByCode = async (
  redirectUri: string,
  code: string
): Promise<OAuth2Profile> => {
  const oAuth2Client = await getOAuth2Client(redirectUri, code)
  const res = await oAuth2Client.request({ url: GAXIOS_URL })
  const token = oAuth2Client.credentials.refresh_token
  return { ...formatProfile(res.data), token }
}
```

其他

Holmes 登陆态除了 Google OAuth2 鉴权外，还做了 jwt 处理，减少谷歌 API 调用次数，加快页面呈现。

