⎇ master ▾   •••

chrome-webstore-upload / How to generate Google API keys.md

🐱 **Leslie-Wong-H** Update API keys guide to include additional step (#49) ✓   🕑

👥 **5 contributors**   👤 👤 👤 👤 🐱

---

83 lines (60 sloc) | 3.79 KB   •••

How to generate Google API keys

chrome-webstore-upload uses the Chrome Web Store API.

Here's how to get its 3 access keys: `clientId`, `clientSecret`, `refreshToken`

Note: the names you enter here don't really matter. This will take approximately 10 minutes and Google likes to change these screens often. Sorry.

1. Visit https://console.developers.google.com/apis/credentials

2. Create a project:
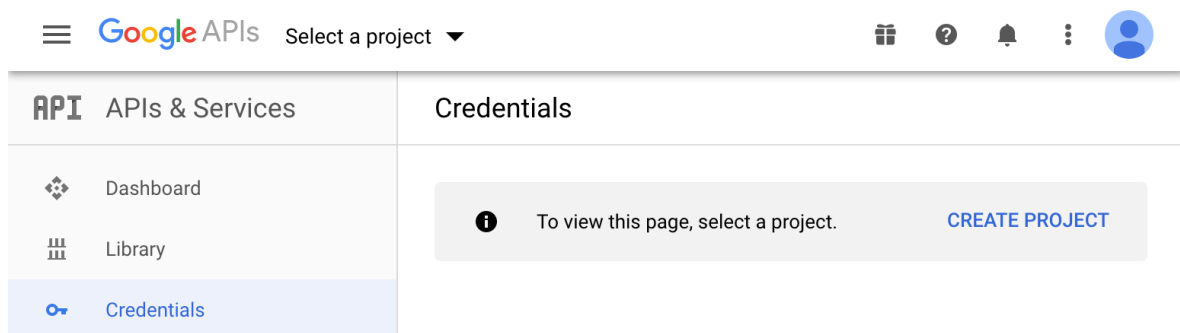
3. Enter `chrome-webstore-upload` and **Create**

4. Click on **Configure consent screen**

5. Select on **External** and **Create**

6. Only enter the Application name (e.g. `chrome-webstore-upload`) and required email fields, and click **Save**

> ## OAuth consent screen
>
> Before your users authenticate, this consent screen will allow them to choose whether they want to grant access to their private data, as well as give them a link to your terms of service and privacy policy. This page configures the consent screen for all applications in this project.
>
> **Verification status**
> Not published
>
> **Application name** ⑦
> The name of the app asking for consent
>
> chrome-webstore-upload

7. On the 3rd screen, add your own email address:

> ## Edit app registration
>
> ✅ OAuth consent screen — ✅ Scopes — ③ **Test users** — ④ Summary
>
> ### Test users
>
> While publishing status is set to "Testing", only test users are able to access the app. Allowed user cap prior to app verification is 100, and is counted over the entire lifetime of the app. Learn more
>
> ＋ ADD USERS

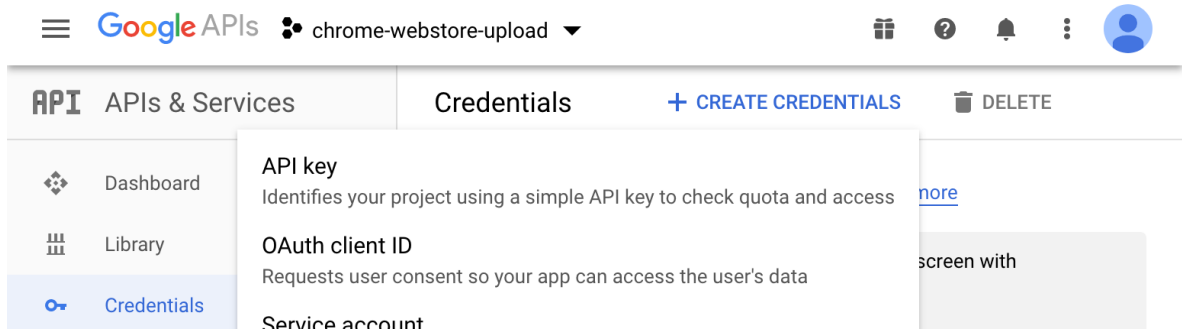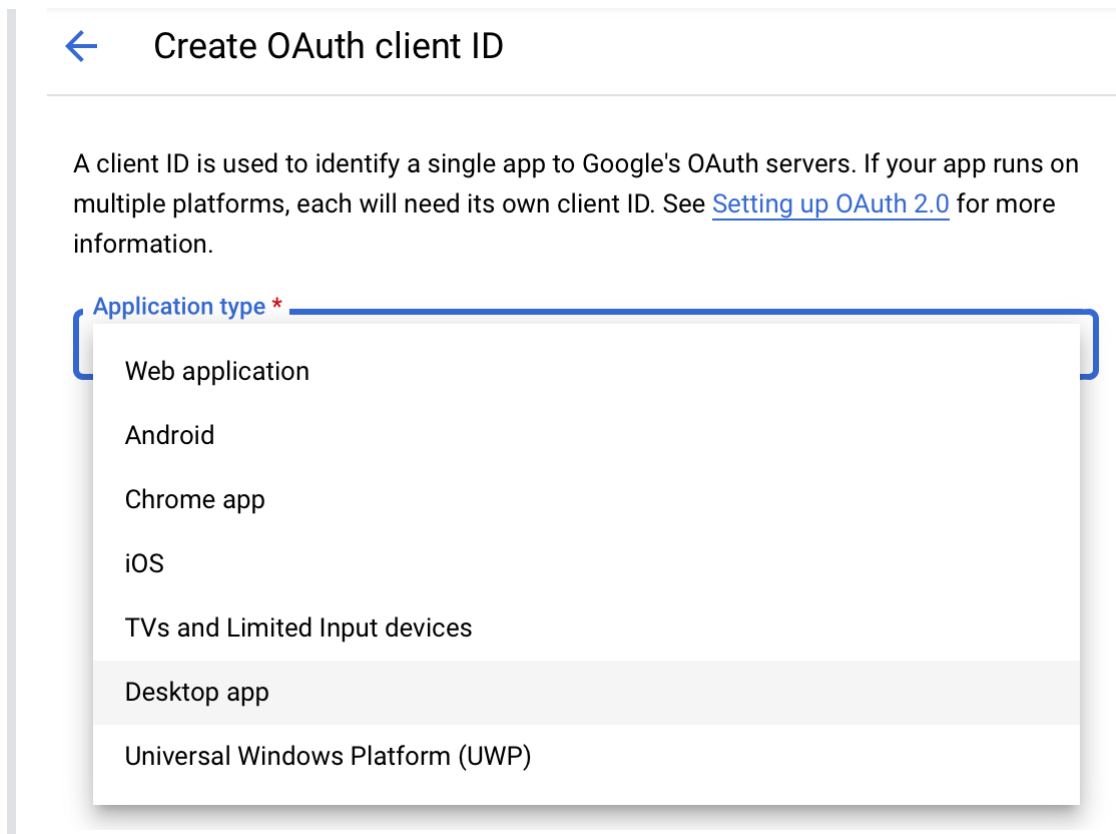8. Visit https://console.developers.google.com/apis/library/chromewebstore.googleapis.com

9. Click **Enable**

10. Visit https://console.developers.google.com/apis/credentials

11. Click **Create credentials** > **OAuth client ID**:

API APIs & Services   **Credentials** **+ CREATE CREDENTIALS** 🗑 **DELETE**

✦ Dashboard   API key
                    Identifies your project using a simple API key to check quota and access nore

⫴ Library   OAuth client ID
                    Requests user consent so your app can access the user's data screen with

🔑 Credentials   Service account

12. Select **Desktop app** (or **Other**), enter `chrome-webstore-upload` and click **Create**

←   **Create OAuth client ID**

A client ID is used to identify a single app to Google's OAuth servers. If your app runs on multiple platforms, each will need its own client ID. See Setting up OAuth 2.0 for more information.

Application type *

Web application

Android

Chrome app

iOS

TVs and Limited Input devices

Desktop app

Universal Windows Platform (UWP)

13. Save your ☑ `clientId` and ☑ `clientSecret`, these are 2 of the required 3 keys

14. Visit https://console.cloud.google.com/apis/credentials/consent

15. Click **PUBLISH APP** and confirm

API APIs & Services   OAuth consent screen

- ❖ Dashboard
- ⅲ Library
- ⊶ Credentials
- ⅲ OAuth consent screen
- ☑ Domain verification
- ⩧ Page usage agreements

chrome-webstore-upload ✏ EDIT APP

Publishing status ❓

Testing

PUBLISH APP

User type

External ❓

MAKE INTERNAL

Test users

While publishing status is set to "Testing", only test users are able to access the app. Allowed user cap prior to app verification is 100, and is counted over the entire lifetime of the app. Learn more

+ ADD USERS

━━━━━━━━ 1 user (1 test, 0 other) / 100 user cap ❓

16. Place your `clientId` in this URL and open it:

    https://accounts.google.com/o/oauth2/auth?
    client_id=YOUR_CLIENT_ID&response_type=code&scope=https://www.googleapis.com/a
    uth/chromewebstore&redirect_uri=urn:ietf:wg:oauth:2.0:oob

17. Follow its steps and warnings (this is your own personal app)

18. Wait on this page:

Google

Sign in

Please copy this code, switch to your application and paste it there:

4/yAEsxNKMzgTZLSGJOhz0RPYaOsWxm1–
Jauq319EWBU7sqyUYlp0A8w ⧉

19. Run this in your browser console **on that last page**. It's a wizard to create your `refresh_token`:

```javascript
(async () => {
  const response = await fetch('https://accounts.google.com/o/oauth2/token', {
    method: "POST",
    body: new URLSearchParams([
      ['client_id', prompt('Enter your clientId')],
      ['client_secret', prompt('Enter your clientSecret')],
      ['code', new URLSearchParams(location.search).get('approvalCode')],
      ['grant_type', 'authorization_code'],
      ['redirect_uri', 'urn:ietf:wg:oauth:2.0:oob']
    ]),
    headers: {
      'Content-Type': 'application/x-www-form-urlencoded'
    }
  });
  const json = await response.json();
  console.log(json);
  if (!json.error) {
    if (typeof copy === 'function') {
      copy(json.refresh_token);
      alert('The refresh_token has been copied into your clipboard. You're done!'
    } else {
      console.log('Copy your token:', json.refresh_token);
      alert('Copy your refresh_token from the console output. You're done!');
    }
  }
})();
```

9001. Done. Now you should have ✓ `clientId` , ✓ `clientSecret` and ✓ `refreshToken` . You can use these for all your extensions, but don't share them!