OAuth 2.0

OAuth 2.0 is stands for open Authorization. It is a kind of protocol

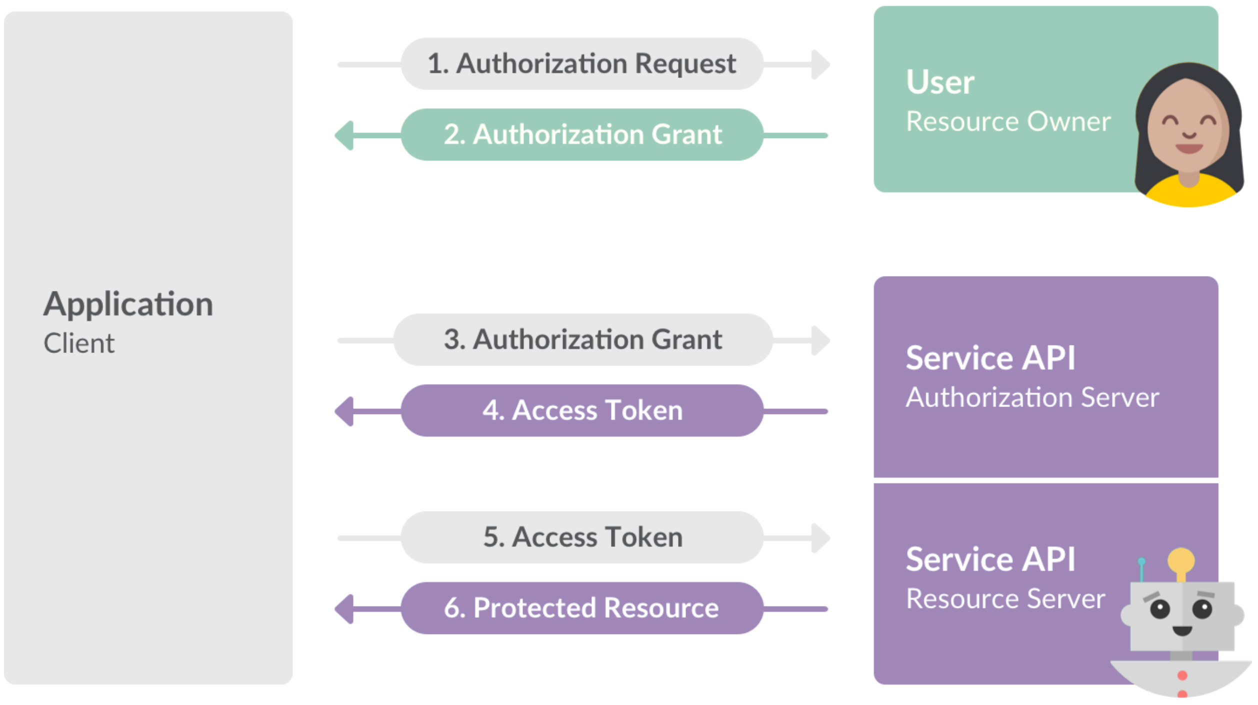
Google APIs use the OAuth 2.0 protocol for authentication and authorization. Google supports common OAuth 2.0 scenarios such as those for web server, client-side, installed, and limited-input device applications.

Obtain OAuth 2.0 credentials from the Google API Console.

Visit the Google API Console to obtain OAuth 2.0 credentials such as a client ID and client secret that are known to both Google and your application. The set of values varies based on what type of application you are building. For example, a JavaScript application does not require a secret, but a web server application does.

TERMINOLOGY USED FOR OAUTH

- Authorization code
- Scope
- Redirect the URL
- Access Token



Client make a request to server then the server send Authorization code. The client use that authorization code and sent that code to the server. Then the server returns the Access Token. with the help of the Access token user can access the API.

SCOPE:

- DATA TO BE ACCESSED BY APP
- REQUEST INFO BY SCOPES
- READ SCOPE READS DATA
- WRITE SCOPE WRITES DATA

Scope is a kind of information that the app just requests in the application.

Basically, data to be accessed by app. App Request in the form of scopes and read scope is used to reads the data. Write scopes is used to writes the data.

REDIRECT URL IN OAUTH2

SERVER

Basically, Redirect URL is assigned by the server.

It is useful for checking the client is valid or not.

This URL is generated on the server site and then the server redirects the user to this URL.

CLIENT ID AND CLIENT SECRET

CLIENT ID

CLIENT SERVER

UNIQUE

Client Id and Client server is a part of the information that you have to setup before you can make or Oauth2 Application.

There are 2 unique information that you want to store anywhere.

ACCESS TOKEN

- SERVER SENDS IT
- EXPLORE API
- REQUEST USER DATA

Access Token is generated by the server. You can request API or you can request profile information.

Tokens can vary in size, up to the following limits:

- Authorization codes: 256 bytes
- Access tokens: 2048 bytes
- Refresh tokens: 512 bytes

Google reserves the right to change token size within these limits, and your application must support variable token sizes accordingly.

REFRESH THE ACCESS TOKEN

Access tokens have limited lifetimes. If your application needs access to a Google API beyond the lifetime of a single access token, it can obtain a refresh token. A refresh token allows your application to obtain new access tokens.

- The refresh token has not been used for six months.
- The user changed passwords and the refresh token contains Gmail scopes.
- The user account has exceeded a maximum number of granted (live) refresh tokens.

A Google Cloud Platform project with an OAuth consent screen configured for an external user type and a publishing status of "Testing" is issued a refresh token expiring in 7 days.

There is currently a limit of 50 refresh tokens per Google Account per OAuth 2.0 client ID. If the limit is reached, creating a new refresh token automatically invalidates the oldest refresh token without warning. This limit does not apply to service accounts.

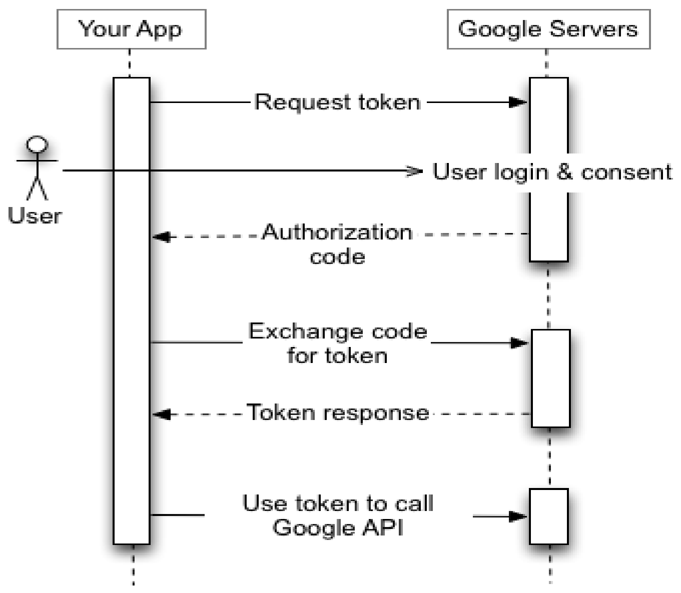
There is also a larger limit on the total number of refresh tokens a user account or service account can have across all clients. Most normal users won't exceed this limit but a developer's account used to test an implementation might.

Web server applications

The Google OAuth 2.0 endpoint supports web server applications that use languages and frameworks such as PHP, Java, Python, Ruby, and ASP.NET.

The authorization sequence begins when your application redirects a browser to a Google URL; the URL includes query parameters that indicate the type of access being requested. Google handles the user authentication, session selection, and user consent. The result is an authorization code, which the application can exchange for an access token and a refresh token.

The application should store the refresh token for future use and use the access token to access a Google API. Once the access token expires, the application uses the refresh token to obtain a new one.


Your App
Google Servers
Request token
User
User login & consent
Authorization code
Exchange code for token
Token response
Use token to call Google API