Last updated May 23, 2017. | Unknown author

How to make API requests

You can use the JavaScript client library to interact with Google APIs, such as People, Calendar, and Drive, from your web applications. Follow the instructions on this page to get started.

There are several ways to use the JavaScript client library to make API requests, but they all follow the same basic pattern:

1. The application loads the JavaScript client library.
2. The application initializes the library with API key, OAuth client ID, and API Discovery Document(s).
3. The application sends a request and processes the response.

The following sections show 3 common ways of using the JavaScript client library.

Option 1: Load the API discovery document, then assemble the request.

The following example assumes the user has already signed in. For a full example of how to sign in a user, see the full auth sample.

```
src="https://apis.google.com/js/api.js"


function start() {
  // 2. Initialize the JavaScript client library.
  gapi.client.init({
    'apiKey': 'YOUR_API_KEY',
    // Your API key will be automatically added to the Discovery Document
URLs.
    'discoveryDocs': ['https://people.googleapis.com/$discovery/rest'],
    // clientId and scope are optional if auth is not required.
    'clientId': 'YOUR_WEB_CLIENT_ID.apps.googleusercontent.com',
    'scope': 'profile',
  }).then(function() {
    // 3. Initialize and make the API request.
    return gapi.client.people.people.get({
      'resourceName': 'people/me',
      'requestMask.includeField': 'person.names'
    });
  }).then(function(response) {
    console.log(response.result);
  }, function(reason) {
    console.log('Error: ' + reason.result.error.message);
  });
};
// 1. Load the JavaScript client library.
gapi.load('client', start);
```

Option 2: Use **gapi.client.request**

A more general way to make requests is to use gapi.client.request. Your application does not have to load the Discovery Document as in the first option, but it must still set the API key (and auth for some APIs). While you need to manually fill in REST parameters with this option, it saves one network request and reduces application size.

```
src="https://apis.google.com/js/api.js"

function start() {
  // 2. Initialize the JavaScript client library.
  gapi.client.init({
    'apiKey': 'YOUR_API_KEY',
    // clientId and scope are optional if auth is not required.
    'clientId': 'YOUR_WEB_CLIENT_ID.apps.googleusercontent.com',
    'scope': 'profile',
  }).then(function() {
    // 3. Initialize and make the API request.
    return gapi.client.request({
      'path': 'https://people.googleapis.com/v1/people/me?
requestMask.includeField=person.names',
    })
  }).then(function(response) {
    console.log(response.result);
  }, function(reason) {
    console.log('Error: ' + reason.result.error.message);
  });
};
// 1. Load the JavaScript client library.
gapi.load('client', start);
```

Option 3: Use CORS

Google APIs support CORS. If your application need to do media uploads and downloads, it should use CORS. See the CORS Support page for details.

Supported environments

The JavaScript client library works with the same browsers supported by Google Apps except that mobile browsers are currently not fully supported. It only works within HTML documents with a <body> element served using the https *(preferred)* and http protocols. However, <iframe sandbox> elements and other restricted execution contexts are not supported.

Setup

Get a Google Account

First, sign up for a Google Account if you do not already have one.

Create a Google project

Go to the Google API Console. Click **Create project**, enter a name, and click **Create**.

Enable Google APIs

Next, decide which Google APIs your application needs to use and enable them for your project. Use the APIs Explorer to explore Google APIs that the JavaScript client library can work with.

To enable an API for your project, do the following:

1. Open the API Library in the Google API Console. If prompted, select a project or create a new one. The API Library lists all available APIs, grouped by product family and popularity.
2. If the API you want to enable isn't visible in the list, use search to find it.
3. Select the API you want to enable, then click the **Enable** button.
4. If prompted, enable billing.
5. If prompted, accept the API's Terms of Service.

Get access keys for your application

Google defines two levels of API access:

Level	Description	Requires:
Simple	API calls do not access any private user data	API key
Authorized	API calls can read and write private user data, or the application's own data	OAuth 2.0 credentials

To acquire an API key for simple access, do the following:

1. Open the Credentials page in the API Console.
2. Click **Create credentials > API key** and select the appropriate key type.

To keep your API keys secure, follow the best practices for securely using API keys.

To acquire OAuth 2.0 credentials for authorized access, do the following:

1. Open the Credentials page in the API Console.
2. Click **Create credentials > OAuth client ID** and select the appropriate Application type.

For information about using OAuth 2.0 credentials, see the Authentication page.