Authenticating to the REST API

In this article

You can authenticate to the REST API to access more endpoints and have a higher rate limit.

About authentication

Many REST API endpoints require authentication or return additional information if you are authenticated. Additionally, you can make more requests per hour when you are authenticated.

You can authenticate your request by sending a token in the `Authorization` header of your request. In the following example, replace `YOUR-TOKEN` with a reference to your token:

```
curl --request GET \
--url "https://api.github.com/octocat" \
--header "Authorization: Bearer YOUR-TOKEN" \
--header "X-GitHub-Api-Version: 2022-11-28"
```

Note: In most cases, you can use `Authorization: Bearer` or `Authorization: token` to pass a token. However, if you are passing a JSON web token (JWT), you must use `Authorization: Bearer`.

If you try to use a REST API endpoint without a token or with a token that has insufficient permissions, you will receive a `404 Not Found` or `403 Forbidden` response.

Authenticating with a personal access token

If you want to use the GitHub REST API for personal use, you can create a personal access token. If possible, GitHub recommends that you use a fine-grained personal access token instead of a personal access token (classic). For more information about creating a personal access token, see "Managing your personal access tokens."

If you are using a fine-grained personal access token, your fine-grained personal access token requires specific permissions in order to access each REST API endpoint. For more information about the permissions that are required for each endpoint, see "Permissions required for fine-grained personal access tokens." If you are using a personal access token (classic), your personal access token (classic) requires specific scopes in order to access each REST API endpoint. For general guidance about what scopes to choose, see

"Scopes for OAuth apps."

If you use a personal access token (classic) to access an organization that enforces SAML single sign-on (SSO) for authentication, you will need to authorize your token after creation. Fine-grained personal access tokens are authorized during token creation, before access to the organization is granted. For more information, see "Authorizing a personal access token for use with SAML single sign-on."

If you do not authorize your personal access token (classic) for SAML SSO before you try to use it to access an organization that enforces SAML SSO, you may receive a `404 Not Found` or a `403 Forbidden` error. If you receive a `403 Forbidden` error, you can follow the URL in the `X-GitHub-SSO` header to authorize your token. The URL expires after one hour. If you requested data that could come from multiple organizations, the API will not return results from the organizations that require SAML SSO. The `X-GitHub-SSO` header will indicate the ID of the organizations that require SAML SSO authorization of your personal access token (classic). For example: `X-GitHub-SSO: partial-results; organizations=21955855,20582480`.

Authenticating with a token generated by an app

If you want to use the API for an organization or on behalf of another user, GitHub recommends that you use a GitHub App. For more information, see "About authentication with a GitHub App."

Your GitHub App requires specific permissions in order to access each REST API endpoint. For more information about the permissions that are required for each endpoint, see "Permissions required for GitHub Apps."

You can also create an OAuth token with an OAuth app to access the REST API. However, GitHub recommends that you use a GitHub App instead. GitHub Apps allow more control over the access and permission that the app has.

Access tokens created by apps are automatically authorized for SAML SSO.

Using basic authentication

Some REST API endpoints for GitHub Apps and OAuth apps require you to use basic authentication to access the endpoint. You will use the app's client ID as the username and the app's client secret as the password.

For example:

```
curl --request POST \
--url "https://api.github.com/applications/YOUR_CLIENT_ID/token" \
--user "YOUR_CLIENT_ID:YOUR_CLIENT_SECRET" \
--header "Accept: application/vnd.github+json" \
--header "X-GitHub-Api-Version: 2022-11-28"
--data '{
  "access_token": "ACCESS_TOKEN_TO_CHECK"
}'
```

The client ID and client secret are associated with the app, not with the owner of the app or a user who authorized the app. They are used to perform operations on behalf of the app, such as creating access tokens.

If you are the owner of a GitHub App or OAuth app, or if you are an app manager for a GitHub App, you can find the client ID and generate a client secret on the settings page for your app. To navigate to your app's settings page:

1. In the upper-right corner of any page on GitHub, click your profile photo.

2. Navigate to your account settings.
    - For an app owned by a personal account, click **Settings**.
    - For an app owned by an organization:
        a. Click **Your organizations**.
        b. To the right of the organization, click **Settings**.
3. In the left sidebar, click <> **Developer settings**.
4. In the left sidebar, click **GitHub Apps** or **OAuth apps**.
5. For GitHub Apps, to the right of the GitHub App you want to access, click **Edit**. For OAuth apps, click the app that you want to access.
6. Next to **Client ID**, you will see the client ID for your app.
7. Next to **Client secrets**, click **Generate a new client secret** to generate a client secret for your app.

Authenticating in a GitHub Actions workflow

If you want to use the API in a GitHub Actions workflow, GitHub recommends that you authenticate with the built-in `GITHUB_TOKEN` instead of creating a token. You can grant permissions to the `GITHUB_TOKEN` with the `permissions` key. For more information, see "Automatic token authentication."

Authenticating with username and password

Authentication with username and password is not supported. If you try to authenticate with user name and password, you will receive a 4xx error.

Further reading

- "Keeping your API credentials secure."