**Getting started with the REST API**

**In this article**

[About the GitHub REST API](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28" \l "about-the-github-rest-api)

[Making a request](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28" \l "making-a-request)

[Authenticating](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28" \l "authenticating)

[Using headers](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28" \l "using-headers)

[Using path parameters](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28" \l "using-path-parameters)

[Using query parameters](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28" \l "using-query-parameters)

[Using body parameters](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28" \l "using-body-parameters)

[Using the response](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28" \l "using-the-response)

[Next steps](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28" \l "next-steps)

Learn how to use the GitHub REST API.

**Tool navigation**

* [GitHub CLI](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28&tool=cli)
* [**curl**](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28&tool=curl)
* [JavaScript](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28&tool=javascript)

[**About the GitHub REST API**](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28#about-the-github-rest-api)

This article describes how to use the GitHub REST API using GitHub CLI, JavaScript, or curl. For a quickstart guide, see "[Quickstart for GitHub REST API](https://docs.github.com/en/rest/quickstart)."

When you make a request to the REST API, you will specify an HTTP method and a path. Additionally, you might also specify request headers and path, query, or body parameters. The API will return the response status code, response headers, and potentially a response body.

The REST API reference documentation describes the HTTP method, path, and parameters for every operation. It also displays example requests and responses for each operation. For more information, see the [REST reference documentation](https://docs.github.com/en/rest).

For more information about GitHub's APIs, see "[Comparing GitHub's REST API and GraphQL API](https://docs.github.com/en/rest/overview/about-githubs-apis)."

[**Making a request**](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28#making-a-request)

To make a request, first find the HTTP method and the path for the operation that you want to use. For example, the "Get Octocat" operation uses the GET method and the /octocat path. For the full reference documentation for this operation, see "[Meta](https://docs.github.com/en/rest/meta#get-octocat)."

Prepend the base URL for the GitHub REST API, https://api.github.com, to the path to get the full URL: https://api.github.com/octocat.

Use the curl command in your command line. Use the --request or -X flag followed by the HTTP method. Use the --url flag followed by the full URL.

curl --request GET \

--url "https://api.github.com/octocat"

**Note**: If you get a message similar to "command not found: curl", you may need to download and install curl. For more information, see [the curl project download page](https://curl.se/download.html).

Continue reading to learn how to authenticate, send parameters, and use the response.

[**Authenticating**](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28#authenticating)

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

[**About tokens**](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28#about-tokens)

You can authenticate your request by adding a token.

If you want to use the GitHub REST API for personal use, you can create a personal access token. The REST API operations used in this article require repo scope for personal access tokens (classic) or, unless otherwise noted, read-only access to public repositories for fine-grained personal access tokens. Other operations may require different scopes or permissions. For more information about creating a personal access token, see "[Managing your personal access tokens](https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/creating-a-personal-access-token)."

If you want to use the API on behalf of an organization or another user, GitHub recommends that you use a GitHub App. If an operation is available to GitHub Apps, the REST reference documentation for that operation will say "Works with GitHub Apps." The REST API operations used in this article require issues read and write permissions for GitHub Apps. Other operations may require different permissions. For more information, see "[Registering a GitHub App](https://docs.github.com/en/apps/creating-github-apps/setting-up-a-github-app/creating-a-github-app)", "[About authentication with a GitHub App](https://docs.github.com/en/apps/creating-github-apps/authenticating-with-a-github-app/about-authentication-with-a-github-app), and "[Authenticating with a GitHub App on behalf of a user](https://docs.github.com/en/apps/creating-github-apps/authenticating-with-a-github-app/identifying-and-authorizing-users-for-github-apps)."

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](https://docs.github.com/en/actions/security-guides/automatic-token-authentication#permissions-for-the-github_token)."

For more information about best practices you can use to keep your tokens secure, see "[Keeping your API credentials secure](https://docs.github.com/en/rest/overview/keeping-your-api-credentials-secure)."

[**Authentication example**](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28#authentication-example)

**Warning**: Treat your access token like a password.

To help keep your account secure, you can use GitHub CLI instead of curl commands. GitHub CLI will take care of authentication for you. For more information, see the GitHub CLI version of this page.

You can also store your token as a Codespaces secret and use the command line through Codespaces. For more information, see "[Managing secrets for your codespaces](https://docs.github.com/en/codespaces/managing-your-codespaces/managing-encrypted-secrets-for-your-codespaces)."

If these options are not possible, consider using another CLI service to store your token securely.

In curl commands, you will send an Authorization header with your token. Replace YOUR-TOKEN with your token:

curl --request GET \

--url "https://api.github.com/octocat" \

--header "Authorization: Bearer YOUR-TOKEN"

**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.

[**Authentication example for GitHub Actions**](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28#authentication-example-for-github-actions)

You can also use the run keyword to execute curl commands in your GitHub Actions workflows. For more information, see "[Workflow syntax for GitHub Actions](https://docs.github.com/en/actions/using-workflows/workflow-syntax-for-github-actions#jobsjob_idstepsrun)."

GitHub recommends that you authenticate with the built-in GITHUB\_TOKEN instead of creating a token. If this is not possible, store your token as a secret and replace GITHUB\_TOKEN in the example below with the name of your secret. For more information about GITHUB\_TOKEN, see "[Automatic token authentication](https://docs.github.com/en/actions/security-guides/automatic-token-authentication)." For more information about secrets, see "[Using secrets in GitHub Actions](https://docs.github.com/en/actions/security-guides/encrypted-secrets)."

jobs:

use\_api:

runs-on: ubuntu-latest

permissions: {}

steps:

- env:

GH\_TOKEN: ${{ secrets.GITHUB\_TOKEN }}

run: |

curl --request GET \

--url "https://api.github.com/octocat" \

--header "Authorization: Bearer $GH\_TOKEN"

[**Using headers**](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28#using-headers)

Most operations specify that you should pass an Accept header with a value of application/vnd.github+json. Other operations may specify that you should send a different Accept header or additional headers.

To send a header in a curl command, use the --header or -H flag followed by the header in key: value format.

curl --request GET \

--url "https://api.github.com/octocat" \

--header "Accept: application/vnd.github+json" \

--header "Authorization: Bearer YOUR-TOKEN" \

--header "X-GitHub-Api-Version: 2022-11-28"

[**Using path parameters**](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28#using-path-parameters)

Path parameters modify the operation path. For example, the "List repository issues" path is /repos/{owner}/{repo}/issues. The curly brackets {} denote path parameters that you need to specify. In this case, you must specify the repository owner and name. For the reference documentation for this operation, see "[Issues](https://docs.github.com/en/rest/issues/issues#list-repository-issues)."

To get issues from the octocat/Spoon-Knife repository, replace {owner} with octocat and {repo} with Spoon-Knife. To build the full path, prepend the base URL for the GitHub REST API, https://api.github.com: https://api.github.com/repos/octocat/Spoon-Knife/issues.

curl --request GET \

--url "https://api.github.com/repos/octocat/Spoon-Knife/issues" \

--header "Accept: application/vnd.github+json" \

--header "Authorization: Bearer YOUR-TOKEN"

The operation returns a list of issues and data about each issue. For more information about using the response, see the "[Using the response](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28#using-the-response)" section.

[**Using query parameters**](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28#using-query-parameters)

Query parameters allow you to control what data is returned for a request. For example, a query parameter may let you specify how many items are returned when the response is paginated.

By default, the "List repository issues" operation returns thirty issues, sorted in descending order by the date they were created. You can use the per\_page parameter to return two issues instead of 30. You can use the sort parameter to sort the issues by the date they were last updated instead of by the date they were created. You can use the direction parameter to sort the results in ascending order instead of descending order.

For curl commands, add a ? to the end of the path, then append your query parameter name and value in the form parameter\_name=value. Separate multiple query parameters with &.

curl --request GET \

--url "https://api.github.com/repos/octocat/Spoon-Knife/issues?per\_page=2&sort=updated&direction=asc" \

--header "Accept: application/vnd.github+json" \

--header "Authorization: Bearer YOUR-TOKEN"

Some operations use query parameters that are arrays. To send an array in the query string, use the query parameter once per array item, and append [] after the query parameter name. For example, to provide an array of two repository IDs, use ?repository\_ids[]=REPOSITORY\_A\_ID&repository\_ids[]=REPOSITORY\_B\_ID.

The operation returns a list of issues and data about each issue. For more information about using the response, see the "[Using the response](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28#using-the-response)" section.

[**Using body parameters**](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28#using-body-parameters)

Body parameters allow you to pass additional data to the API. For example, the "Create an issue" operation requires you to specify a title for the new issue. It also lets you specify other information, such as text to put in the issue body. For the full reference documentation for this operation, see "[Issues](https://docs.github.com/en/rest/issues/issues#create-an-issue)."

The "Create an issue" operation uses the same path as the "List repository issues" operation in the examples above, but it uses a POST method instead of a GET method.

If you are using a fine-grained personal access token, you must replace octocat/Spoon-Knife with a repository that you own or that is owned by an organization that you are a member of. Your token must have access to that repository and have read and write permissions for repository issues. For more information about creating a repository, see "[Create a repo](https://docs.github.com/en/get-started/quickstart/create-a-repo)." For more information about granting access and permissions to a fine-grained personal access token, see "[Managing your personal access tokens](https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/creating-a-personal-access-token)."

For curl commands, use the --data flag to pass the body parameters in a JSON object.

curl --request POST \

--url "https://api.github.com/repos/octocat/Spoon-Knife/issues" \

--header "Accept: application/vnd.github+json" \

--header "Authorization: Bearer YOUR-TOKEN" \

--data '{

"title": "Created with the REST API",

"body": "This is a test issue created by the REST API"

}'

The operation creates an issue and returns data about the new issue. In the response, find the html\_url of your issue and navigate to your issue in the browser. For more information about using the response, see the "[Using the response](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28#using-the-response)" section.

[**Using the response**](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28#using-the-response)

[**About the response code and headers**](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28#about-the-response-code-and-headers)

Every request will return an HTTP status code that indicates the success of the response. For more information about response codes, see [the MDN HTTP response status code documentation](https://developer.mozilla.org/en-US/docs/Web/HTTP/Status).

Additionally, the response will include headers that give more details about the response. Headers that start with X- or x- are custom to GitHub. For example, the x-ratelimit-remaining and x-ratelimit-reset headers tell you how many requests you can make in a time period.

To view the status code and headers, use the --include or --i flag when you send your request.

For example, this request:

curl --request GET \

--url "https://api.github.com/repos/octocat/Spoon-Knife/issues?per\_page=2" \

--header "Accept: application/vnd.github+json" \

--header "Authorization: Bearer YOUR-TOKEN" \

--include

returns the response code and headers like:

HTTP/2 200

server: GitHub.com

date: Thu, 04 Aug 2022 20:07:51 GMT

content-type: application/json; charset=utf-8

cache-control: public, max-age=60, s-maxage=60

vary: Accept, Accept-Encoding, Accept, X-Requested-With

etag: W/"7fceb7e8c958d3ec4d02524b042578dcc7b282192e6c939070f4a70390962e18"

x-github-media-type: github.v3; format=json

link: <https://api.github.com/repositories/1300192/issues?per\_page=2&sort=updated&direction=asc&page=2>; rel="next", <https://api.github.com/repositories/1300192/issues?per\_page=2&sort=updated&direction=asc&page=7409>; rel="last"

access-control-expose-headers: ETag, Link, Location, Retry-After, X-GitHub-OTP, X-RateLimit-Limit, X-RateLimit-Remaining, X-RateLimit-Used, X-RateLimit-Resource, X-RateLimit-Reset, X-OAuth-Scopes, X-Accepted-OAuth-Scopes, X-Poll-Interval, X-GitHub-Media-Type, X-GitHub-SSO, X-GitHub-Request-Id, Deprecation, Sunset

access-control-allow-origin: \*

strict-transport-security: max-age=31536000; includeSubdomains; preload

x-frame-options: deny

x-content-type-options: nosniff

x-xss-protection: 0

referrer-policy: origin-when-cross-origin, strict-origin-when-cross-origin

content-security-policy: default-src 'none'

x-ratelimit-limit: 15000

x-ratelimit-remaining: 14996

x-ratelimit-reset: 1659645535

x-ratelimit-resource: core

x-ratelimit-used: 4

accept-ranges: bytes

content-length: 4936

x-github-request-id: 14E0:4BC6:F1B8BA:208E317:62EC2715

In this example, the response code is 200, which indicates a successful request.

[**About the response body**](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28#about-the-response-body)

Many operations will return a response body. Unless otherwise specified, the response body is in JSON format. For example, this request returns a list of issues with data about each issue:

curl --request GET \

--url "https://api.github.com/repos/octocat/Spoon-Knife/issues?per\_page=2" \

--header "Accept: application/vnd.github+json" \

--header "Authorization: Bearer YOUR-TOKEN"

Unlike the GraphQL API where you specify what information you want, the REST API typically returns more information than you need. If desired, you can parse the response to pull out specific pieces of information.

For example, you can use > to redirect the response to a file:

curl --request GET \

--url "https://api.github.com/repos/octocat/Spoon-Knife/issues?per\_page=2" \

--header "Accept: application/vnd.github+json" \

--header "Authorization: Bearer YOUR-TOKEN" > data.json

Then you can use jq to get the title and author ID of each issue:

jq '.[] | {title: .title, authorID: .user.id}' data.json

The previous two commands return something like:

{

"title": "Update index.html",

"authorID": 10701255

}

{

"title": "Edit index file",

"authorID": 53709285

}

For more information about jq, see [the jq documentation](https://stedolan.github.io/jq/).

[**Next steps**](https://docs.github.com/en/rest/guides/getting-started-with-the-rest-api?apiVersion=2022-11-28#next-steps)

This article demonstrated how to list and create issues in a repository. For more practice, try to comment on an issue, edit the title of an issue, or close an issue. For more information about these operations, see "[Issues](https://docs.github.com/en/rest/issues#create-an-issue-comment)" and "[Issues](https://docs.github.com/en/rest/issues/issues#update-an-issue)."

For more information about the operations that you can use, see the [REST reference documentation](https://docs.github.com/en/rest).