About GitHub's APIs

In this article

Learn about GitHub's APIs to extend and customize your GitHub experience.

About GitHub's APIs

GitHub provides two APIs: a REST API and a GraphQL API. You can interact with both APIs using GitHub CLI, curl, the official Octokit libraries, and third party libraries. Occasionally, a feature may be supported on one API but not the other.

You should use the API that best aligns with your needs and that you are most comfortable using. You don't need to exclusively use one API over the other. Node IDs let you move between the REST API and GraphQL API. For more information, see "Using global node IDs."

This article discusses the benefits of each API. For more information about the GraphQL API, see "About the GraphQL API." For more information about the REST API, see the REST documentation.

Choosing the GraphQL API

The GraphQL API returns exactly the data that you request. GraphQL also returns the data in a pre-known structure based on your request. In contrast, the REST API returns more data than you requested and returns it in a pre-determined structure. You can also accomplish the equivalent of multiple REST API request in a single GraphQL request. The ability to make fewer requests and fetch less data makes GraphQL appealing to developers of mobile applications.

For example, to get the GitHub Enterprise Cloud login of ten of your followers, and the login of ten followers of each of your followers, you can send a single request like:

```
{
  viewer {
    followers(first: 10) {
      nodes {
        login
        followers(first: 10) {
          nodes {
            login
          }
        }
      }
    }
  }
}
```

The response will be a JSON object that follows the structure of your request.

In contrast, to get this same information from the REST API, you would need to first make a request to `GET /user/followers`. The API would return the login of each follower, along with other data about the followers that you don't need. Then, for each follower, you would need to make a request to `GET /users/{username}/followers`. In total, you would need to make 11 requests to get the same information that you could get from a single GraphQL request, and you would receive excess data.

Choosing the REST API

Because REST APIs have been around for longer than GraphQL APIs, some developers are more comfortable with the REST API. Since REST APIs use standard HTTP verbs and concepts, many developers are already familiar with the basic concepts to use the REST API.

For example, to create an issue in the `octocat/Spoon-Knife` repository, you would need to send a request to `POST /repos/octocat/Spoon-Knife/issues` with a JSON request body:

```
{
  "title": "Bug with feature X",
  "body": "If you do A, then B happens"
}
```

In contrast, to make an issue using the GraphQL API, you would need to get the node ID of the `octocat/Spoon-Knife` repository and then send a request like:

```
mutation {
  createIssue(
    input: {
      repositoryId: "MDEwOlJlcG9zaXRvcnkxMzAwMTky"
      title: "Bug with feature X"
      body: "If you do A, then B happens"}
  ) {
    issue {
      number
      url
    }
  }
}
```