About creating GitHub Apps

In this article

GitHub Apps let you build integrations to automate processes and extend GitHub's functionality.

About GitHub Apps

A GitHub App is a type of integration that you can build to interact with and extend the functionality of GitHub. You can build a GitHub App to provide flexibility and reduce friction in your processes, without needing to sign in a user or create a service account.

Common use cases for GitHub Apps include:

- Automating tasks or background processes
- Supporting "Sign in with GitHub," which allows users to sign in with their GitHub account to provide their identity in your ecosystem
- As a developer tool, allowing users to work with GitHub by signing into your GitHub App, which can then act on their behalf
- Integrating your tool or external service with GitHub

Like OAuth apps, GitHub Apps use OAuth 2.0 and can act on behalf of a user. Unlike OAuth apps, GitHub Apps can also act independently of a user.

GitHub Apps can be installed directly on organizations and personal accounts and granted access to specific repositories. They come with built-in webhooks and narrow, specific permissions.

By default, only organization owners can manage the settings of GitHub Apps in an organization. To allow additional users to change the developer settings of GitHub Apps owned by the organization, an owner can grant them GitHub App manager permissions. GitHub App Managers can't manage third-party applications. For more information about adding and removing GitHub App managers in your organization, see "Roles in an organization."

Building a GitHub App

In order to build a GitHub App, you first need to register a GitHub App. For more information, see "Registering a GitHub App."

Then, you need to write code to add functionality to your GitHub App. You can use the credentials from your GitHub App registration to make authenticated requests to GitHub's APIs. For more information about writing code for your GitHub App, see "About writing code for a GitHub App." For more information about making authenticated requests, see "About authentication with a GitHub App."

Once you have written the code for your GitHub App, your app needs to run somewhere. If your app is a website or web app, you might host your app on a server like Azure App Service. If your app is a client-side app, it might run on a user's device.

In order to use your GitHub App, you must install the app on your organization or personal account. If your GitHub App is private, you can only install the GitHub App on the account that owns the app. If your GitHub App is public, other users and organizations can install your app. For more information, see "Installing your own GitHub App" and "Sharing your GitHub App."

Understanding what type of GitHub App to build

There are multiple ways to design a GitHub App that you will want to consider, based on the functionality you want the app to have.

GitHub Apps that act on behalf of a user

If you want your app to take actions on behalf of a user, you should use a user access token for authentication. This type of request is sometimes called "user-to-server," and it means that the app will be limited by the permissions that have been given to the app as well as the user's permission. With this pattern, the user must authorize the app before the app can take action. For more information, see "Authenticating with a GitHub App on behalf of a user."

Some examples of automations you could create with a GitHub App, where the app acts on a user's behalf, include:

- A GitHub App that uses GitHub as an identity provider for your ecosystem.
- A GitHub App that adds a service on top of GitHub.com that might be useful to a GitHub user. You can share the app with other developers via GitHub Marketplace or by making the app public.

GitHub Apps that act on their own behalf

If you want your app to take actions on behalf of itself, rather than a user, you should use an installation access token for authentication. This type of request is sometimes called "server-to-server," and it means that the app will be limited by the permissions that have been given to the app. For more information, see "Authenticating as a GitHub App installation."

Some examples of automations you could create with a GitHub App, where the app acts on its own behalf, include:

- A GitHub App that uses webhooks to react to an event given a certain set of criteria. For example, you could create an automation around the REST API endpoints for reviewing requests for fine-grained personal access token that approves a request given a certain policy.
- A GitHub App that helps repository contributors. For example, the app could post helpful resources after a contributor creates a pull request or makes a comment.
- A GitHub App that generates short-lived tokens to give to other CI/CD tools, or to pull information from a repository.

GitHub Apps that respond to webhooks

If you want your app to respond to events on GitHub, your app should subscribe to webhooks. For example, you may want your app to leave a comment when a pull request is opened. For more information, see "Using webhooks with GitHub Apps."

GitHub Apps that can take certain actions

When you set up your GitHub App, you can select specific permissions for the app. These permissions determine what the app can do via the GitHub API, what they can do on behalf of a signed in user, and what webhooks the app can receive. For more information, see "Choosing permissions for a GitHub App."