Quickstart for building GitHub Apps

In this article

Introduction
Prerequisites
Step 1: Clone the app code
Step 2: Get a webhook proxy URL
Step 3: Register a GitHub App
Step 4: Store identifying information and credentials
Step 5: Install your app
Step 6: Start your server
Step 7: Test your app
Next steps

Quickly build a GitHub App that comments on pull requests.

Introduction

GitHub Apps let you automate processes or integrate other platforms with GitHub. For more info, see "About creating GitHub Apps."

This quickstart describes how to quickly create a GitHub App. When a pull request is opened in a repository that the app was granted access to, the app will add a comment to the pull request.

This quickstart uses pre-written code to help you get started quickly. For a more detailed tutorial that helps you write the code, see "Building a GitHub App that responds to webhook events."

Prerequisites

Your computer or codespace should use Node.js version 12 or greater. For more information, see Node.js.

Step 1: Clone the app code

To help you get started quickly, we wrote code that you can use. If you want to learn how to write the code yourself, see "Building a GitHub App that responds to webhook events".

1. Clone the github/github-app-js-sample repository. For more information, see "Cloning a repository." You may use a local clone or GitHub Codespaces.
2. In a terminal window, navigate to the directory where your clone is stored.
3. Run `npm install` to install the dependencies.

Step 2: Get a webhook proxy URL

In order to develop your app locally, you can use a webhook proxy URL to forward webhooks from GitHub to your computer or codespace. This quickstart uses Smee.io to provide a webhook proxy URL and forward webhooks.

1. In your browser, navigate to https://smee.io/.
2. Click **Start a new channel**.
3. Copy the full URL under "Webhook Proxy URL". You will use this URL in a later step.

Step 3: Register a GitHub App

The following steps will guide you through configuring the app settings that are required for this quickstart. For more information about the settings, see "Registering a GitHub App."

1. In the upper-right corner of any page on GitHub, click your profile photo.
2. Navigate to your account settings.
   - For a GitHub App owned by a personal account, click **Settings**.
   - For a GitHub 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**.
5. Click **New GitHub App**.
6. Under "GitHub App name", enter a name for your app. For example, `USERNAME-quickstart-app` where `USERNAME` is your GitHub username.
7. Under "Homepage URL", enter `https://github.com/github/github-app-js-sample#readme`.
8. Skip the "Identifying and authorizing users" and "Post installation" sections for this quickstart. For more information about these settings, see "Registering a GitHub App."
9. Make sure that **Active** is selected under "Webhooks."
10. Under "Webhook URL", enter your webhook proxy URL from earlier. For more information, see "Step 2: Get a webhook proxy URL."
11. Under "Webhook secret", enter a random string. You will use this string later.
12. Under "Repository permissions", next to "Pull requests," select **Read & write**.
13. Under "Subscribe to events", select **Pull request**.
14. Under "Where can this GitHub App be installed?", select **Only on this account**.

15. Click **Create GitHub App**.

Step 4: Store identifying information and credentials

In this quickstart, you will store your app's credentials and identifying information as environment variables in a `.env` file. When you deploy your app, you will want to change how you store the credentials. For more information, see "Deploy your app."

Make sure that you are on a secure machine before performing these steps since you will store your credentials locally.

Create a `.env` file

Your cloned repository includes `.env` in the `.gitignore` file. This will prevent you from accidentally committing your app's credentials. For more information about `.gitignore` files, see "Ignoring files."

1. Navigate to the directory where your clone of github/github-app-js-sample is stored.
2. Create a file called `.env` at the top level of this directory.
3. Add the following contents to your `.env` file. Replace `YOUR_HOSTNAME` with the name of your GitHub Enterprise Server instance. You will update the other values in a later step.

```
Text
APP_ID="YOUR_APP_ID"
WEBHOOK_SECRET="YOUR_WEBHOOK_SECRET"
PRIVATE_KEY_PATH="YOUR_PRIVATE_KEY_PATH"
HOSTNAME="YOUR_HOSTNAME"
```

Navigate to your app settings

If you navigated away from your app settings after creating your app, navigate to the settings page for your app:

1. In the upper-right corner of any page on GitHub, click your profile photo.
2. Navigate to your account settings.
   - For a GitHub App owned by a personal account, click **Settings**.
   - For a GitHub 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**.
5. Next to your app's name, click **Edit**.

Get your app credentials and identifying information

1. On your app's settings page, next to "App ID", find the app ID for your app.
2. In your `.env` file, replace `YOUR_APP_ID` with the app ID of your app.
3. On your app's settings page, under "Private keys", click **Generate a private key**. You will see a private key in PEM format downloaded to your computer. For more information, see "Managing private keys for GitHub Apps."
4. If you are using a codespace, move the downloaded PEM file into your codespace so that your codespace can access the file.
5. In your `.env` file, replace `YOUR_PRIVATE_KEY_PATH` with the full path to your private key, including the `.pem` extension.
6. In your `.env` file, replace `YOUR_WEBHOOK_SECRET` with the webhook secret for your app. If you have forgotten your webhook secret, under "Webhook secret (optional)", click **Change secret**. Enter a new secret, then click **Save changes**.

Step 5: Install your app

In order for your app to leave a comment on pull requests in a repository, it must be installed on the account that owns the repository and granted access to that repository. Since your app is private, it can only be installed on the account that owns the app.

1. In the account that owns the app you created, create a new repository to install the app on. For more information, see "Creating a new repository."
2. If you navigated away from your app settings after creating your app, navigate to the settings page for your app. For more information, see "Navigate to your app settings."
3. Click **Public page**.
4. Click **Install**.
5. Select **Only select repositories**.
6. Select the **Select repositories** dropdown menu and click the repository that you chose at the start of this section.
7. Click **Install**.

Step 6: Start your server

For testing, you will use your computer or codespace as a server. Your app will only be active when your server is running.

1. In a terminal window, navigate to the directory where your clone of github/github-app-js-sample is stored.
2. To receive forwarded webhooks from Smee.io, run `npx smee -u WEBHOOK_PROXY_URL -t http://localhost:3000/api/webhook`. Replace `WEBHOOK_PROXY_URL` with your webhook proxy URL. If you forgot your URL, you can find it in the "webhook URL" field on your app's settings page.

You should see output that looks like this, where `WEBHOOK_PROXY_URL` is your webhook proxy URL:

```
Forwarding WEBHOOK_PROXY_URL to http://localhost:3000/api/webhook
Connected WEBHOOK_PROXY_URL
```

3. In a second terminal window, navigate to the directory where your clone of github/github-app-js-sample is stored.
4. Run `npm run server`. Your terminal should say, `Server is listening for events at: http://localhost:3000/api/webhook`.

Step 7: Test your app

Now that your server is running and receiving forwarded webhooks events, test your app by opening a pull request.

1. Open a pull request on the repository you created in Step 5: Install your app. For more information, see "Creating a pull request."
2. Navigate to your webhook proxy URL on smee.io. You should see a `pull_request` event. This indicates that GitHub successfully sent a pull request event when you created a pull request.
3. In the terminal where you ran `npm run server`, you should see something like "Received a pull request event for #1" where the integer after the `#` is the number of the pull request that you opened.
4. In the timeline of your pull request, you should see a comment from your app. The comment uses the contents of the `message.md` file in your cloned repository.
5. In both terminal windows, enter `Ctrl` + `C` to stop your server and stop listening for forwarded webhooks.

Next steps

Now that you have an app, you might want to expand your app's code, deploy your app, and make your app public.

Modify the app code

Fork the github/github-app-js-sample repository and modify the code to respond to different webhook events or to make different API requests. For more information about the code, see "Building a GitHub App that responds to webhook events."

Remember to update your app's permissions if your app needs additional permissions for the API requests that you want to make or the webhook events you want to receive. For more information, see "Choosing permissions for a GitHub App."

Deploy your app

This tutorial used your computer or codespace as a server. Once the app is ready for production use, you should deploy your app to a dedicated server. For example, you can use Azure App Service.

Once you have a server, update the webhook URL in your app settings. You should not use Smee.io to forward your webhooks in production.

You will also need to update the `port` and `host` constants in the code. For more information, see "Building a GitHub App that responds to webhook events."

You should never publicize your app's private key or webhook secret. This tutorial stored your app's credentials in a gitignored `.env` file. When you deploy your app, you should choose a secure way to store the credentials and update your code to get the value accordingly. For example, you can store the credentials in an environment variable on the server where your app is deployed. You can also use a secret management service like Azure Key Vault.

Share your app

If you want to share your app with other users and organizations, make your app public. For more information, see "Making a GitHub App public or private."

Follow best practices

You should aim to follow best practices with your GitHub App. For more information, see "Best practices for creating a GitHub App."

Legal

© 2023 GitHub, Inc. Terms Privacy Status Pricing Expert services Blog