About GitHub Codespaces prebuilds

In this article

Overview

The prebuild process

About pushing changes to prebuild-enabled branches

GitHub Codespaces prebuilds help to speed up the creation of new codespaces for large or complex repositories.

You create and configure prebuilds in your repository's settings. Repository-level settings for GitHub Codespaces are available for all repositories owned by personal accounts.

For repositories owned by organizations, repository-level settings for GitHub Codespaces are available for organizations on GitHub Team and GitHub Enterprise plans. To access the settings, the organization or its parent enterprise must have added a payment method and set a spending limit for GitHub Codespaces. For more information, see "Choosing who owns and pays for codespaces in your organization" and "GitHub's plans."

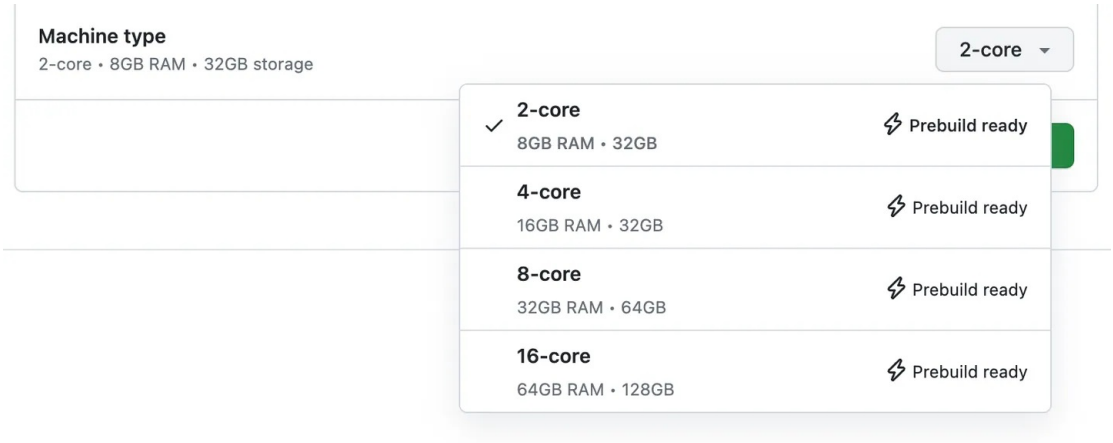
Overview

A prebuild assembles the main components of a codespace for a particular combination of repository, branch, and `devcontainer.json` configuration file. It provides a quick way to create a new codespace. For complex and/or large repositories in particular, you can create a new codespace more quickly by using a prebuild.

If it currently takes more than 2 minutes to create a codespace for a repository, you are likely to benefit from using prebuilds. This is because, with a prebuild, any source code, editor extensions, project dependencies, commands, and configurations have already been downloaded, installed, and applied before you create a codespace.

By default, whenever you push changes to your repository, GitHub Codespaces uses GitHub Actions to automatically update your prebuilds.

When prebuilds are available for a particular branch of a repository, a particular dev container configuration file, and for your region, you'll see the "Prebuild ready" label in the list of machine type options when you create a codespace. If a prebuild is still being created, you will see the "Prebuild in progress" label. For more information, see "Creating a codespace for a repository."



When you create a codespace from a template on the "Your codespaces" page, GitHub may automatically use a prebuild to speed up creation time. For more information on templates, see "Creating a codespace from a template."

Note: Each prebuild that's created consumes storage space that will either incur a billable charge or, for repositories owned by your personal GitHub account, will use some of your monthly included storage. For more information, see "About billing for GitHub Codespaces."

The prebuild process

To create a prebuild, you set up a prebuild configuration. When you save the configuration, a GitHub Actions workflow runs to create each of the required prebuilds; one workflow per prebuild. Workflows also run whenever the prebuilds for your configuration need to be updated. This can happen at scheduled intervals, on pushes to a prebuild-enabled repository, or when you change the dev container configuration. For more information, see "Configuring prebuilds."

When a prebuild configuration workflow runs, GitHub creates a temporary codespace, performing setup operations up to and including any `onCreateCommand` and `updateContentCommand` commands in the `devcontainer.json` file. No `postCreateCommand` commands are run during the creation of a prebuild. For more information about these commands, see the `devcontainer.json` reference in the VS Code documentation. A snapshot of the generated container is then taken and stored.

As with other GitHub Actions workflows, running a prebuild configuration workflow will either consume some of the GitHub Actions minutes included with your account, if you have any, or it will incur charges for GitHub Actions minutes. Storage of codespace prebuilds is billed in the same way as storage of active or stopped codespaces. For more information, see "About billing for GitHub Codespaces."

When you create a codespace from a prebuild, GitHub downloads the existing container snapshot from storage and deploys it on a fresh virtual machine, completing the remaining commands specified in the dev container configuration. Since many operations have already been performed, such as cloning the repository, creating a codespace from a prebuild can be substantially quicker than creating one without a prebuild. This is true where the repository is large and/or `onCreateCommand` commands take a long time to run.

About pushing changes to prebuild-enabled branches

By default, each push to a branch that has a prebuild configuration results in a GitHub-managed GitHub Actions workflow run to update the prebuild. The prebuild workflow has a concurrency limit of one workflow run at a time for a given prebuild configuration,

unless changes were made that affect the dev container configuration for the associated repository. For more information, see "Introduction to dev containers." If a run is already in progress, the workflow run that was queued most recently will run next, after the current run completes.

With the prebuild set to be updated on each push, it means that if there are very frequent pushes to your repository, prebuild updates will occur at least as often as it takes to run the prebuild workflow. That is, if your workflow run typically takes one hour to complete, prebuilds will be created for your repository roughly hourly, if the run succeeds, or more often if there were pushes that change the dev container configuration on the branch.

For example, let's imagine 5 pushes are made, in quick succession, against a branch that has a prebuild configuration. In this situation:

- A workflow run is started for the first push, to update the prebuild.
- If the 4 remaining pushes do not affect the dev container configuration, the workflow runs for these are queued in a "pending" state.

  If any of the remaining 4 pushes change the dev container configuration, then the service will not skip that one and will immediately run the prebuild creation workflow, updating the prebuild accordingly if it succeeds.
- Once the first run completes, workflow runs for pushes 2, 3, and 4 will be canceled, and the last queued workflow (for push 5) will run and update the prebuild.