Troubleshooting prebuilds

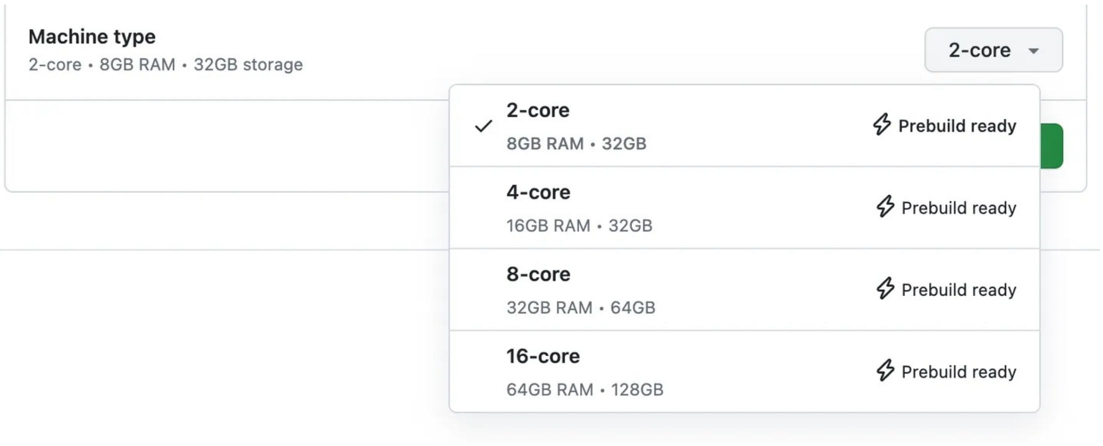
In this article

You can use prebuilds to speed up the creation of codespaces. This article provides troubleshooting steps for common issues with prebuilds.

For more information about GitHub Codespaces prebuilds, see "Prebuilding your codespaces."

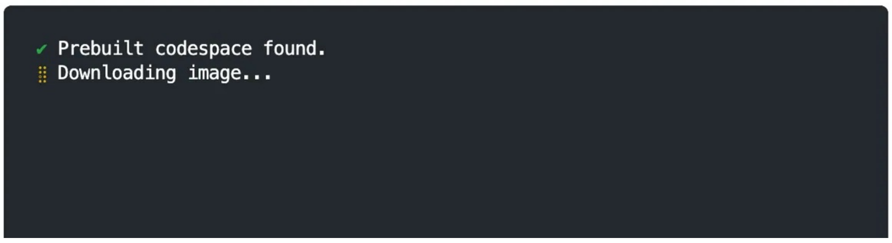
Checking whether a codespace was created from a prebuild?

When you create a codespace, you can choose the type of the virtual machine you want to use. If a prebuild is available for the type of virtual machine, "Prebuild ready" is shown next to it.



If you have your GitHub Codespaces editor preference set to "Visual Studio Code for Web" then the "Setting up your codespace" page will show the message "Prebuilt codespace found" if a prebuild is being used.

Setting up your codespace


```
✔ Prebuilt codespace found.
⣾ Downloading image...
```

Similarly, if your editor preference is "VS Code" then the integrated terminal will contain the message "You are on a prebuilt codespace defined by the prebuild configuration for your repository" when you create a new codespace. For more information, see "Setting your default editor for GitHub Codespaces."

After you have created a codespace you can check whether it was created from a prebuild by running the following GitHub CLI command in the terminal:

Shell

```shell
gh api /user/codespaces/$CODESPACE_NAME --jq .prebuild
```

This returns `true` if the codespace was created using a prebuild.

Alternatively, if GitHub CLI (`gh`) is not installed, you can use the following command, which returns `createFromPrebuild` if the codespace was created from a prebuild:

Shell

```shell
cat /workspaces/.codespaces/shared/environment-variables.json | jq '.ACTION_NAME'
```

Checking prebuild usage

You can check whether a repository is using prebuilds in the "Codespaces" page of the repository's settings.

You can check how much storage space has been consumed by prebuilds in your current billing cycle by reviewing the billing data for your personal or organization account. You can also generate a usage report to see which repositories have been using prebuilds. For more information, see "Viewing your GitHub Codespaces usage."

The "Prebuild Ready" label is sometimes missing

You may notice that sometimes, when you create a new codespace from a prebuild-enabled branch, the "⚡ Prebuild Ready" label is not displayed in the dialog box for choosing a machine type. This means that prebuilds are not currently available.

By default, each time you push to a prebuild-enabled branch, the prebuild is updated. If the push involves a change to the dev container configuration then, while the update is in progress, the "⚡ Prebuild Ready" label is removed from the list of machine types. During this time you can still create codespaces without a prebuild. If required, you can reduce the occasions on which prebuilds are unavailable for a repository by setting the prebuild to be updated only when you make a change to your dev container configuration files, or only on a custom schedule. For more information, see "Configuring prebuilds."

If your branch is not specifically enabled for prebuilds it may still benefit from prebuilds if it was branched from a prebuild-enabled branch. However, if the dev container configuration is changed on your branch, so that it's not the same as the configuration on the base branch, prebuilds will no longer be available on your branch.

Here are things to check if the "Prebuild Ready" label is not displayed for a particular branch:

- Confirm that a prebuild configuration exists for this branch. If you’re not a repository administrator, you'll need to reach out to one to confirm this.
- Confirm that the prebuild configuration includes your region.
- Check whether a change to the dev container configuration was pushed to the prebuild-enabled branch recently. If so, you will typically have to wait until the prebuild workflow run for this push completes before prebuilds are available again.
- If no configuration changes were recently made, go to the **Actions** tab of your repository, click **Codespaces Prebuilds** in the workflows list, and check that prebuild workflow runs for the branch are succeeding. If latest runs of a workflow failed, and one or more of these failed runs contained changes to the dev container configuration, then there will be no available prebuilds for the associated branch.

Some resources cannot be accessed in codespaces created using a prebuild

If the `devcontainer.json` configuration file for a prebuild configuration specifies that permissions for access to other repositories are required, then the repository administrator is prompted to authorize these permissions when they create or update the prebuild configuration. If the administrator does not grant all of the requested permissions there's a chance that problems may occur in the prebuild, and in codespaces created from this prebuild. This is true even if the user who creates a codespace based on this prebuild *does* grant all of the permissions when they are prompted to do so.

Troubleshooting failed workflow runs for prebuilds

Increasing the GitHub Actions spending limit

Prebuilds are created and updated using GitHub Actions. Your prebuild workflows will fail if you have used all of your GitHub Actions minutes and have reached your spending limit. If this occurs you can increase your GitHub Actions spending limit to allow the workflows to run. For more information, see "Managing your spending limit for GitHub Actions."

Authorizing access permissions

If the `devcontainer.json` configuration file for a prebuild configuration is updated to specify that permissions for access to other repositories are required, and a repository administrator has not been prompted to authorize these permissions for the prebuild configuration, then the prebuild workflow may fail. Try updating the prebuild configuration, without making any changes. If, when you click **Update**, the authorization page is displayed, check that the requested permissions are appropriate and, if so, authorize the request. For more information, see "Managing prebuilds" and "Managing access to other repositories within your codespace."

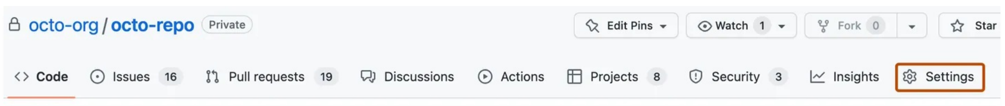
If the workflow runs for a prebuild configuration are failing, you can temporarily disable the prebuild configuration while you investigate. For more information, see "Managing prebuilds."

Preventing out-of-date prebuilds being used

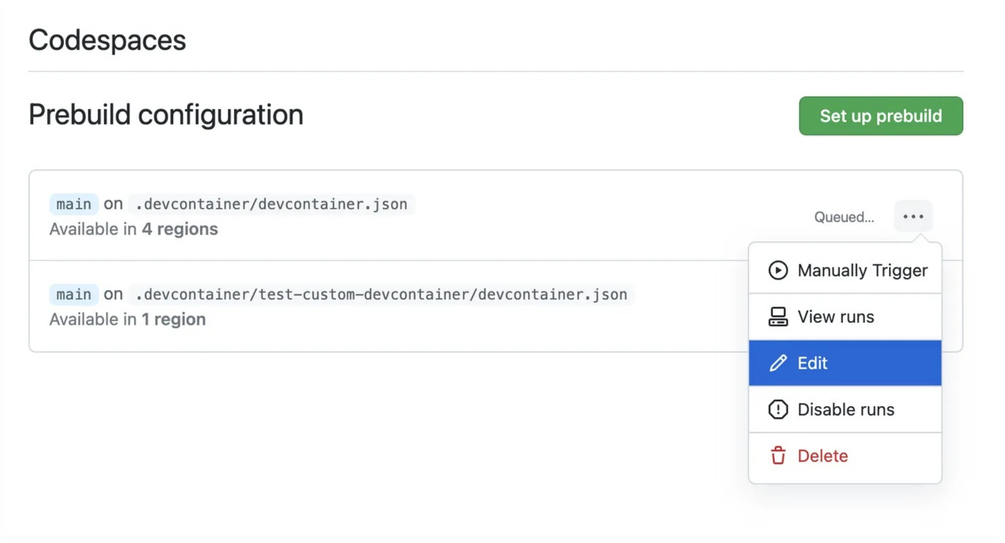
By default, if the latest prebuild workflow has failed, then a previous prebuild for the same combination of repository, branch, and `devcontainer.json` configuration file will be used to create new codespaces. This behavior is called prebuild optimization.

We recommend keeping prebuild optimization enabled, because it helps ensure that codespaces can still be created quickly if an up-to-date prebuild is not available. However, as a repository administrator, you can disable prebuild optimization if you run into problems with prebuilt codespaces being behind the current state of the branch. If you disable prebuild optimization, codespaces for the relevant combination of repository, branch, and `devcontainer.json` file will be created without a prebuild if the latest prebuild workflow has failed or is currently running.

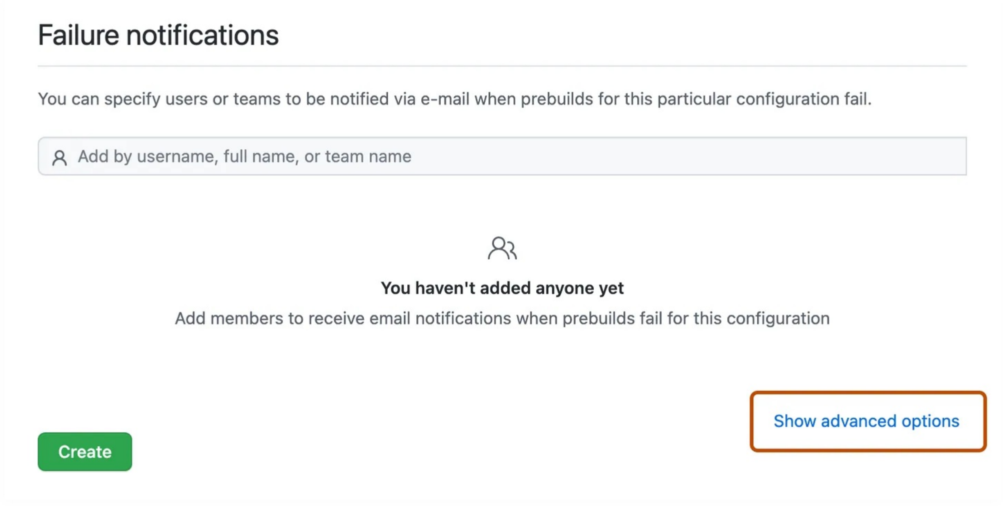
1. On GitHub.com, navigate to the main page of the repository.
2. Under your repository name, click **Settings**. If you cannot see the "Settings" tab, select the ··· dropdown menu, then click **Settings**.



3. In the "Code & automation" section of the side bar, click **Codespaces**.
4. To the right of the affected prebuild configuration, select the ellipsis (**...**), then click **Edit**.



5. Scroll to the bottom of the "Edit configuration" page and click **Show advanced options**.


Failure notifications

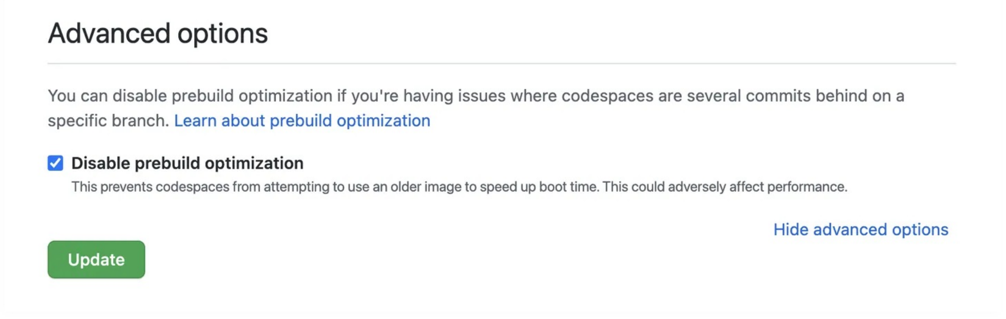
You can specify users or teams to be notified via e-mail when prebuilds for this particular configuration fail.

Add by username, full name, or team name

You haven't added anyone yet

Add members to receive email notifications when prebuilds fail for this configuration

6. If you're sure you want to disable the default setting, select **Disable prebuild optimization**.


Advanced options

You can disable prebuild optimization if you're having issues where codespaces are several commits behind on a specific branch. Learn about prebuild optimization

☑ Disable prebuild optimization

This prevents codespaces from attempting to use an older image to speed up boot time. This could adversely affect performance.

7. To save your change, click **Update**.

Further reading

- "Configuring prebuilds"
- "Managing prebuilds"