Developing in a codespace

In this article

You can work in a codespace using your browser, Visual Studio Code, a JetBrains IDE, or in a command shell.

GitHub CLI JetBrains IDEs (Beta) Visual Studio Code Web browser

Note: Using GitHub Codespaces with JetBrains IDEs is currently in public beta and is subject to change.

About development with GitHub Codespaces

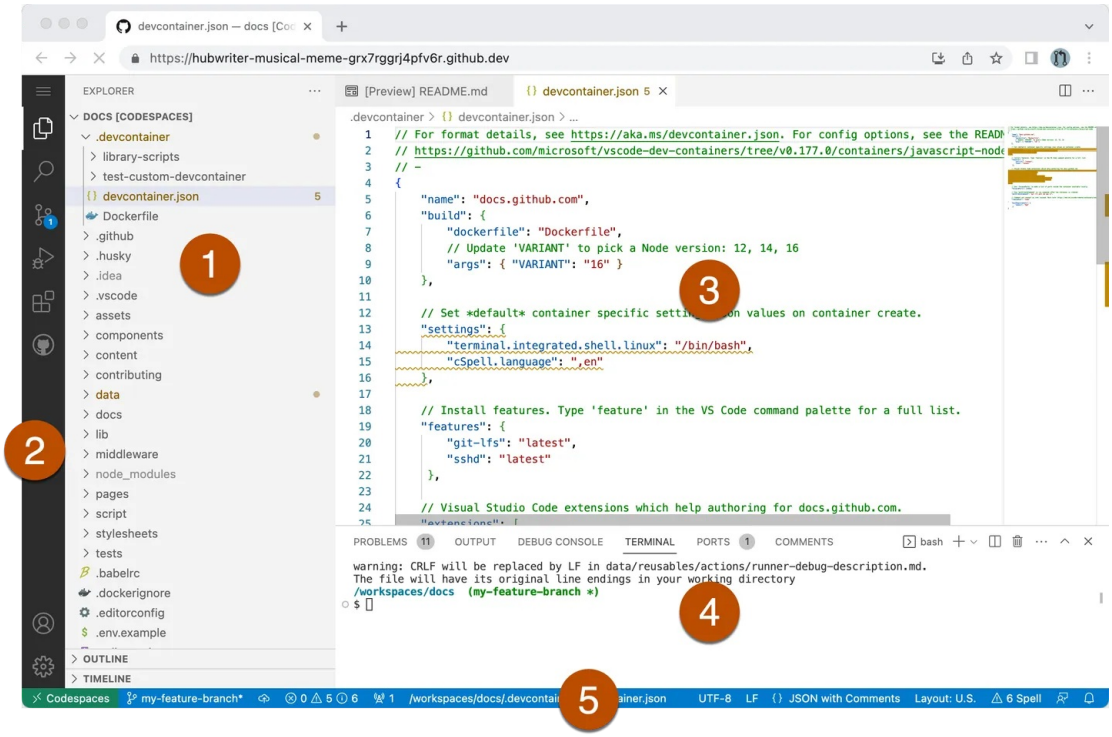
You can develop code in a codespace using your choice of tool:

- A command shell, via an SSH connection initiated using GitHub CLI.
- One of the JetBrains IDEs, via the JetBrains Gateway.
- The Visual Studio Code desktop application.
- A browser-based version of Visual Studio Code.

The tabs in this article allow you to switch between information for each of these ways of working. You're currently on the tab for the web browser version of Visual Studio Code.

Working in a codespace in the browser

Using Codespaces in the browser provides you with a fully featured development experience. You can edit code, debug, use Git commands, and run your application.



The main components of the user interface are:

1. **Side bar** - By default, this area shows your project files in the Explorer.
2. **Activity bar** - This displays the Views and provides you with a way to switch between them. You can reorder the Views by dragging and dropping them.
3. **Editor** - This is where you edit your files. You can right-click the tab for a file to access options such as locating the file in the Explorer.
4. **Panels** - This is where you can see output and debug information, as well as the default place for the integrated Terminal.
5. **Status bar** - This area provides you with useful information about your codespace and project. For example, the branch name, configured ports, and more.

For the best experience with GitHub Codespaces, we recommend using a Chromium-based browser, like Google Chrome or Microsoft Edge. For more information, see "Troubleshooting GitHub Codespaces clients."

Customizing the codespaces for a repository

You can customize the codespaces that are created for a repository by creating or updating the dev container configuration for the repository. You can do this from within a codespace. After you change a dev container configuration, you can apply the changes to the current codespace by rebuilding the Docker container for the codespace. For more information, see "Introduction to dev containers."

Personalizing your codespace

You can use a dotfiles repository and Settings Sync to personalize aspects of the codespace environment for any codespace that you create. Personalization can include shell preferences and additional tools. For more information, see "Personalizing GitHub Codespaces for your account."

Running your app from a codespace

You can forward ports in your codespace to test and debug your application. You can also manage the port protocol and share the port within your organization or publicly. For more information, see "Forwarding ports in your codespace."

Committing your changes

When you've made changes to your codespace, either new code or configuration changes, you'll want to commit your changes. Committing configuration changes to your repository ensures that anyone else who creates a codespace from this repository has the same configuration. Any customization you do, such as adding VS Code extensions, will be available to all users.

For this tutorial, you created a codespace from a template repository, so the code in your codespace is not yet stored in a repository. You can create a repository by publishing the current branch to GitHub.com.

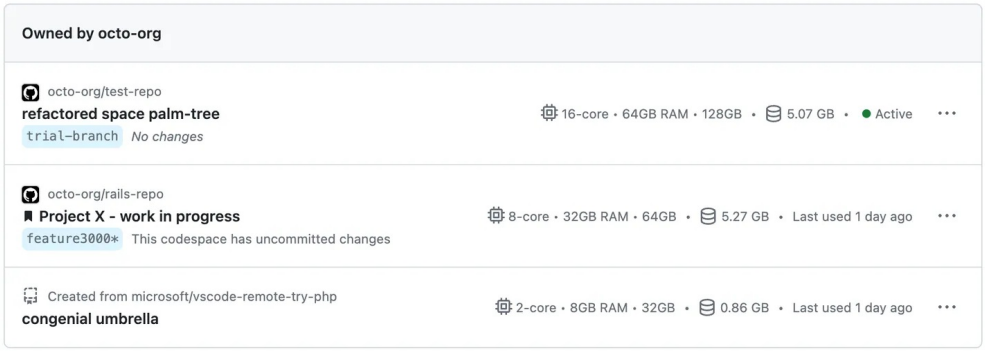
For information, see "Using source control in your codespace."

Using the Visual Studio Code Command Palette

The Visual Studio Code Command Palette allows you to access and manage many features for Codespaces and Visual Studio Code. For more information, see "Using the Visual Studio Code Command Palette in GitHub Codespaces."

Navigating to an existing codespace

1. You can see every available codespace that you have created at github.com/codespaces.
2. Click the name of the codespace you want to develop in.

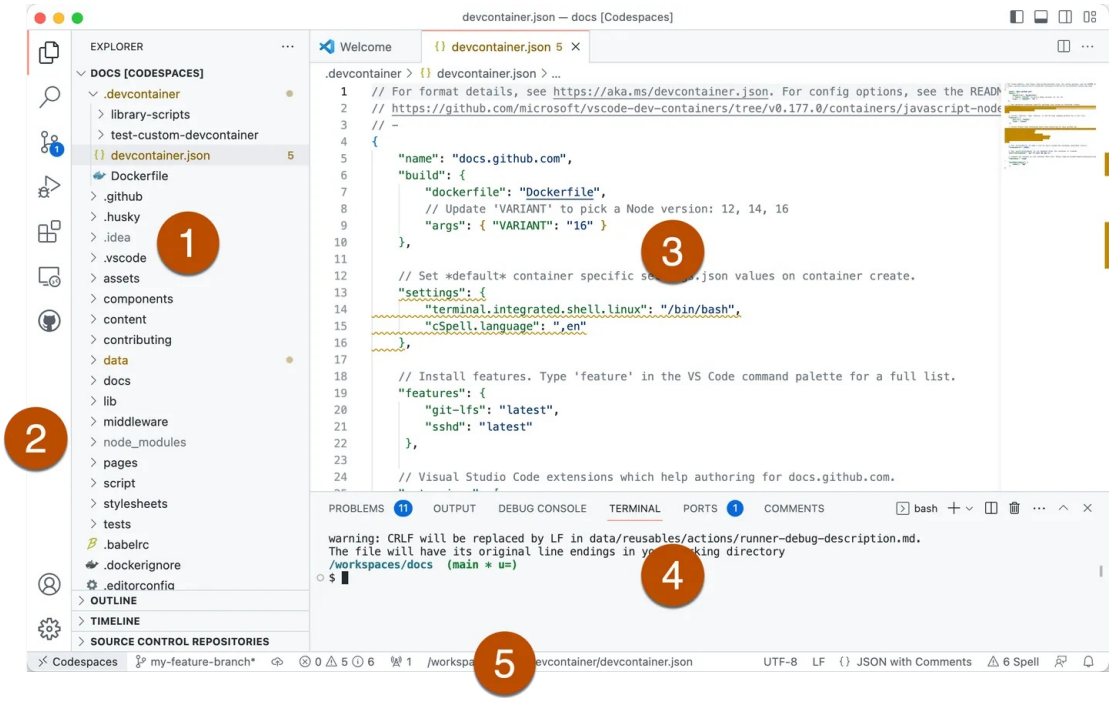


Alternatively, you can see any of your codespaces for a specific repository by navigating to that repository and selecting <> **Code**. The dropdown menu will display all active codespaces for a repository.

The tabs in this article allow you to switch between information for each of these ways of working. You're currently on the tab for Visual Studio Code.

Working in a codespace in VS Code

GitHub Codespaces provides you with the full development experience of Visual Studio Code. You can edit code, debug, and use Git commands while developing in a codespace with VS Code. For more information, see the VS Code documentation.



The main components of the user interface are:

1. **Side bar** - By default, this area shows your project files in the Explorer.
2. **Activity bar** - This displays the Views and provides you with a way to switch between them. You can reorder the Views by dragging and dropping them.
3. **Editor** - This is where you edit your files. You can right-click the tab for a file to access options such as locating the file in the Explorer.
4. **Panels** - This is where you can see output and debug information, as well as the default place for the integrated Terminal.
5. **Status bar** - This area provides you with useful information about your codespace and project. For example, the branch name, configured ports, and more.

For more information on using VS Code, see the User Interface guide in the VS Code documentation.

You can connect to your codespace directly from VS Code. For more information, see "Using GitHub Codespaces in Visual Studio Code."

For troubleshooting information, see "Troubleshooting GitHub Codespaces clients."

Customizing the codespaces for a repository

You can customize the codespaces that are created for a repository by creating or updating the dev container configuration for the repository. You can do this from within a codespace. After you change a dev container configuration, you can apply the changes to the current codespace by rebuilding the Docker container for the codespace. For more information, see "Introduction to dev containers."

Personalizing your codespace

You can use a dotfiles repository and Settings Sync to personalize aspects of the codespace environment for any codespace that you create. Personalization can include shell preferences and additional tools. For more information, see "Personalizing GitHub Codespaces for your account."

Running your app from a codespace

You can forward ports in your codespace to test and debug your application. You can also manage the port protocol and share the port within your organization or publicly. For more information, see "Forwarding ports in your codespace."

Committing your changes

When you've made changes to your codespace, either new code or configuration changes, you'll want to commit your changes. Committing configuration changes to your repository ensures that anyone else who creates a codespace from this repository has the same configuration. Any customization you do, such as adding VS Code extensions, will be available to all users.

For this tutorial, you created a codespace from a template repository, so the code in your codespace is not yet stored in a repository. You can create a repository by publishing the current branch to GitHub.com.

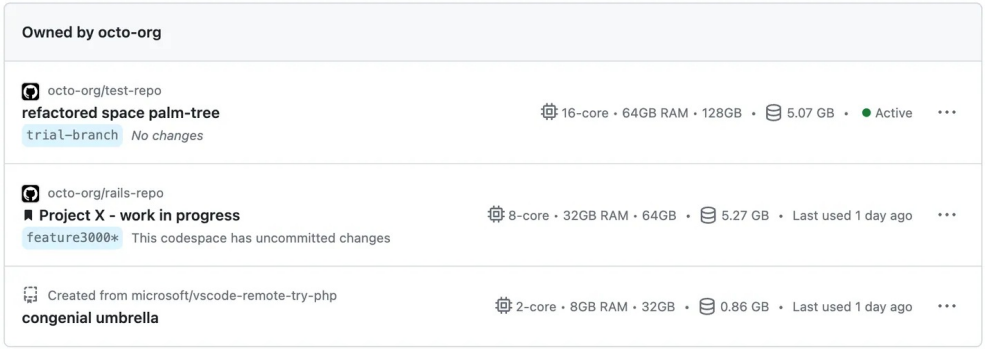
For information, see "Using source control in your codespace."

Using the Visual Studio Code Command Palette

The Visual Studio Code Command Palette allows you to access and manage many features for Codespaces and Visual Studio Code. For more information, see "Using the Visual Studio Code Command Palette in GitHub Codespaces."

Navigating to an existing codespace

1. You can see every available codespace that you have created at github.com/codespaces.
2. Click the name of the codespace you want to develop in.



Alternatively, you can see any of your codespaces for a specific repository by navigating to that repository and selecting <> **Code**. The dropdown menu will display all active codespaces for a repository.

The tabs in this article allow you to switch between information for each of these ways of working. You're currently on the tab for JetBrains IDEs.

Working in a codespace in a JetBrains IDE

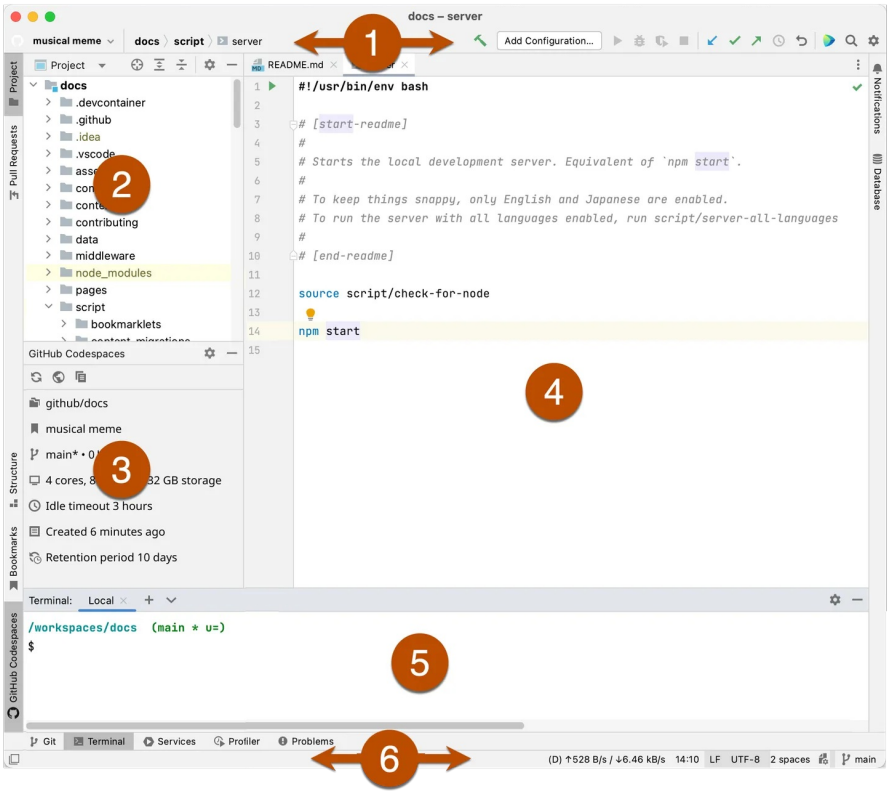
To use GitHub Codespaces with a JetBrains IDE you must have already installed JetBrains Gateway. For information about installing JetBrains Gateway, see the JetBrains website.

You can work in a codespace using your choice of JetBrains IDE. After creating a codespace, you can use the JetBrains Gateway application to open the codespace in your preferred IDE.

You can edit code, debug, and use Git commands while developing in a codespace with your JetBrains IDE. For more information about the various JetBrains IDEs, see the JetBrains documentation.

IntelliJ IDEA user interface

Within the GitHub Codespaces documentation we use IntelliJ IDEA as a representative JetBrains IDE. Different JetBrains IDEs may have different layouts.



The main components of the user interface are:

1. **Navigation bar** - This displays the path to the currently selected file or directory. Use the buttons to the right of the navigation bar to perform various actions, including building, running, or debugging the project, or running Git commands to commit and push your changes.

2. **Project tool window** - This shows you the structure of your project and allows you to open files in the editor.

3. **GitHub Codespaces tool window** - This is displayed by clicking the GitHub Codespaces plugin in the bar to the left of the tool window. It displays information about your codespace, including its display name and machine type. The buttons at the top of this tool window allow you to:

   - Refresh the details in the tool window for the active codespace
   - Display the "Your codespaces" web page
   - View the codespace creation logs

4. **Editor** - This is where you edit your files. You can right-click the tab for a file to access options such as moving the tab to a new window.

5. **Terminal** - This is displayed by clicking **Terminal** in the tool window bar at the bottom of the main window (just above the status bar). The integrated terminal

allows you to perform command-line tasks without having to switch to a dedicated terminal application.

6. **Status bar** - Hover over the icon at the left of the status bar to see a list of tools. Click the icon to hide or show the tool window bars. The right side of the status bar shows information about the project, including the current Git branch.

For more information about the IntelliJ IDEA user interface, see the JetBrains documentation for IntelliJ IDEA.

Customizing the codespaces for a repository

You can customize the codespaces that are created for a repository by creating or updating the dev container configuration for the repository. You can do this from within a codespace. After you change a dev container configuration, you can apply the changes to the current codespace by rebuilding the Docker container for the codespace. For more information, see "Introduction to dev containers."

Personalizing your codespace

You can use a dotfiles repository to personalize aspects of the codespace environment for any codespace that you create. For more information, see "Personalizing GitHub Codespaces for your account."

Committing your changes

Once you've made changes to your codespace, either new code or configuration changes, you'll want to commit and push your changes. Pushing changes to a repository ensures that anyone else who creates a codespace from this repository has the same configuration. This also means that any customization you do, to modify the configuration of codespaces created for a repository, will be available to everybody who uses the repository.

For more information, see "Using source control in your codespace."

Further reading

- "Using GitHub Codespaces in your JetBrains IDE"
- "Using the GitHub Codespaces plugin for JetBrains"
- "Troubleshooting GitHub Codespaces clients"

The tabs in this article allow you to switch between information for each of these ways of working. You're currently on the tab for GitHub CLI.

Working in a codespace in a command shell

> To learn more about GitHub CLI, see "About GitHub CLI."

You can use GitHub CLI to create a new codespace, or start an existing codespace, and then SSH to it. Once connected, you can work on the command line using your preferred command-line tools.

After installing GitHub CLI and authenticating with your GitHub account you can use the command `gh codespace [<SUBCOMMAND>...] --help` to browse the help information. Alternatively, you can view the same reference information at

https://cli.github.com/manual/gh_codespace.

For more information, see "Using GitHub Codespaces with GitHub CLI."