Changing the shell in a codespace

In this article

Changing from the default shell in VS Code
Installing a new shell
Setting the default shell in VS Code
Setting the default shell over SSH
Configuring your shell

You can change your shell in a codespace to keep the setup you're used to.

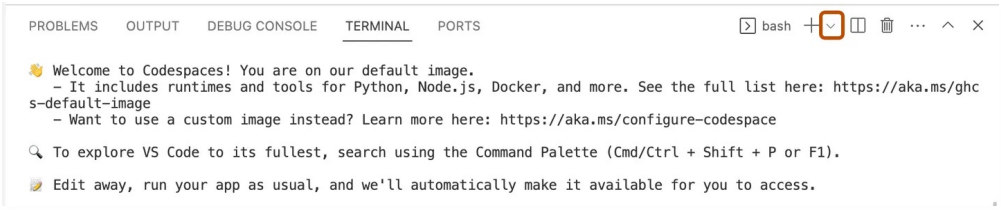
When you're working in a codespace, you can open a new terminal window with a shell of your choice, change your default shell for new terminal windows, or install a new shell. You can also use dotfiles to configure your shell.

Codespaces that use the default codespace image come with the `bash`, `zsh`, and `fish` shells installed. If you open a new codespace in the VS Code web client, or connect to a codespace over SSH, the terminal opens with a `bash` session running by default. In the VS Code desktop application, the default shell depends on your local settings and operating system. For more information, see Terminal Profiles in the VS Code documentation.

Changing from the default shell in VS Code

If you don't want to use the default shell, you can open a new terminal session with a different shell.

1. If you cannot see the integrated terminal in VS Code, press `Ctrl` + `` ` ``.
2. To the right of the + icon for opening a new terminal window, select the dropdown icon.



3. In the dropdown menu, click the name of the shell you want to use.

Installing a new shell

If you want to use a shell that isn't already installed in the base image or dev container configuration for a codespace, you can install a new shell.

If you're using the default codespace image, look for installation instructions for Ubuntu

Linux. If you just want to use a different shell for one session, you can use the command line to install the shell in the codespace you're working in. However, you may lose programs you have installed if you rebuild the container in the codespace. For more information, see "Deep dive into GitHub Codespaces."

A more robust option for installing new shells is to include the installation commands either in a dotfiles repository, or as a lifecycle command such as `postCreateCommand` in a `devcontainer.json` file. You should use a dotfiles repository to install a shell you want to use in all your own codespaces, and a `devcontainer.json` file for a shell that contributors to a specific repository should have installed. For more information, see "Personalizing GitHub Codespaces for your account" and "Introduction to dev containers."

Adding a VS Code terminal profile for a new shell

VS Code automatically detects most standard shells and adds them as a terminal profile, so you can easily open new terminal windows using the shell you have installed.

If the shell you install isn't detected automatically, you can add a new terminal profile to your user settings. This setting is dependent on your operating system, so you should use `linux` for the VS Code web client and your local operating system for the desktop application.

1. To open the Visual Studio Code Command Palette, press `Command` + `Shift` + `P` (Mac) or `Ctrl` + `Shift` + `P` (Windows).
2. Start typing "user settings," then click **Preferences: Open User Settings (JSON)**.
3. In the `settings.json` file, inside the JSON object, add a new property like the following. Replace `OPERATING-SYSTEM` with the relevant operating system (such as `linux`, `windows`, or `osx`) and `SHELL` with the shell you have installed.

   JSON

   ```
   "terminal.integrated.profiles.OPERATING-SYSTEM": {
     "SHELL": {
       "path": "SHELL"
     }
   }
   ```

   For example:

   ```
   "terminal.integrated.profiles.linux": {
    "csh": {
      "path": "csh"
    }
   }
   ```

4. Save the file.

You can use Settings Sync to share these settings across all codespaces you open in the VS Code web client and desktop application. If you're working in the web client, Settings Sync is disabled by default, and you must enable Settings Sync to push changes to your settings or pull in new changes you have made elsewhere. For more information, see "Personalizing GitHub Codespaces for your account."

Setting the default shell in VS Code

You can set a default terminal profile to choose the default shell used for all new terminal

windows you open in VS Code. The default terminal profile is dependent on your operating system, so you can set a default profile for Linux, if you're using the VS Code web client, or for your local operating system, if you're using the desktop application.

> **Note:** Regardless of your default profile, codespaces opened in the web client always open with a `bash` session running initially.

1. To open the Visual Studio Code Command Palette, press `Command` + `Shift` + `P` (Mac) or `Ctrl` + `Shift` + `P` (Windows).
2. Start typing "user settings," then click **Preferences: Open User Settings (JSON)**.
3. Inside the JSON object, to set the default shell for the relevant operating system, add lines or edit existing lines like the following.

   ```
   "terminal.integrated.defaultProfile.OPERATING-SYSTEM": "SHELL"
   ```

   For example:

   JSON

   ```
   {
      "terminal.integrated.defaultProfile.osx": "zsh",
      "terminal.integrated.defaultProfile.linux": "bash",
      "terminal.integrated.defaultProfile.windows": "PowerShell"
   }
   ```

4. Save the `settings.json` file.

You can use Settings Sync to share these settings across all codespaces you open in the VS Code web client and desktop application. If you're working in the web client, Settings Sync is disabled by default, and you must enable Settings Sync to push changes to your settings or pull in new changes you have made elsewhere. For more information, see "Personalizing GitHub Codespaces for your account."

Setting the default shell over SSH

When you connect to a codespace from the command line over SSH, you connect to a `bash` session in the codespace by default.

If you have enabled a dotfiles repository for GitHub Codespaces, you can change the default shell you connect to by adding a command to an installation script such as `install.sh` in your dotfiles. For more information, see "Using GitHub Codespaces with GitHub CLI" and "Personalizing GitHub Codespaces for your account." For example, the following command changes the default shell to `zsh`.

Shell

```
sudo chsh "$(id -un)" --shell "/usr/bin/zsh"
```

If you want to use a default shell that isn't installed in your codespace by default, or ensure you have the latest version of the shell, you can install the shell first.

Shell

```
sudo apt-get update -y
sudo apt-get install -y csh
sudo chsh "$(id -un)" --shell "/usr/bin/csh"
```

Note: If you create a new codespace (for example by using `gh codespace create`), you must wait sufficient time to ensure the script has finished running before you connect to the codespace over SSH. If the script hasn't finished running, you will connect to a default `bash` session.

When you have connected to the codespace, for most shells, you can use the command `readlink /proc/$$/exe` to check the correct shell is running.

Configuring your shell

With most shells, you have the option of using a configuration file, such as `.bashrc`, to configure the shell with your preferred settings. These settings can include things like aliases and environment variables.

By default, codespaces contain predefined configuration for the shells that come preinstalled. For example, the home directory in a codespace contains `.bashrc` and `.zshrc` files. You can change the contents of these files then use a command like `source ~/.bashrc` to update your shell configuration. However, you will lose any changes to these files if you rebuild the container in a codespace. For more information, see "Deep dive into GitHub Codespaces."

Generally, you should use a dotfiles repository to configure shells with your preferred settings. The setup in your dotfiles applies to all codespaces you create, and persists over rebuilds of the container. For more information, see "Personalizing GitHub Codespaces for your account."

Troubleshooting the `fish` shell

The `fish` shell includes a web-based configuration interface. You can use the `fish_config` command to start a local web server and launch this interface, then do things like change the terminal prompt or view your environment variables.

You can use the web-based interface for `fish` in a codespace. However, the color settings in VS Code's integrated terminal depend on your chosen VS Code theme, and you cannot override these settings by setting a new theme in the `fish_config` interface.

When `fish` starts the local server, the default link that GitHub Codespaces provides to the forwarded port does not work. For example, if you click **Open in Browser** on the popup message, you will be taken to an error page.

To access the web-based interface for `fish_config`:

1. In a terminal running a `fish` session, enter `fish_config`.
2. In the terminal output, use `Command`+click or `Ctrl`+click to open the link to the `web_config` HTML file.

   ```
   $ fish_config
   Web config started at file:///tmp/web_config60rc9tr3.html
   Hit ENTER to stop.
   ```

3. In the `web_config` file, use `Command`+click or `Ctrl`+click to open the link to the forwarded port.

   ```
   <body>
   ```

```
  <p><a href="http://localhost:8000/1b9411c2469e392b96df5e5b28da485b/">Start
the Fish Web config</a></p>
</body>
```