



Introduction to GitHub Codespaces



Presented by

Tech Skills Transformations LLC

© 2024 Brent C. Laster & Tech Skills Transformations LLC



TECHUPSKILLS ®

All rights reserved



Welcome to the Introduction to GitHub Codespaces Workshop!

3

- Log into GitHub
- Go to <https://github.com/skillrepos/cspaces-intro>
- Fork project
- Follow instructions in README to setup codespace
- When done open *labs.md* file **with Preview**
- Keep copy of *labs.md* open in separate browser tab/window

The screenshot shows a browser window with the URL <https://github.com/skillrepos/cspaces-intro>. The repository name is **cspaces-intro**, described as **Public**. The **Code** tab is selected, showing a list of files:

File	Description	Last Commit
techupskills Update labs.md	bf95b7a · 7 minutes ago	135 Commits
extra	Create registry-compose.yml	5 months ago
images	Add files via upload	8 minutes ago
roar-k8s	Rename mysqlsecret.yaml to my...	5 months ago
LICENSE	Create LICENSE	5 months ago
README.md	Update README.md	1 hour ago
labs.md	Update labs.md	7 minutes ago

The **About** section includes:

- No description, website, or topics provided.
- Readme
- GPL-3.0 license
- Activity
- Custom properties
- 0 stars
- 0 watching
- 1 fork

The **Releases** section states: No releases published.

The **Packages** section states: No packages published.

The **Contributors** section lists three contributors:

- techupskills**
- brentlaster** Brent Laster
- sasbcl** Brent Laster

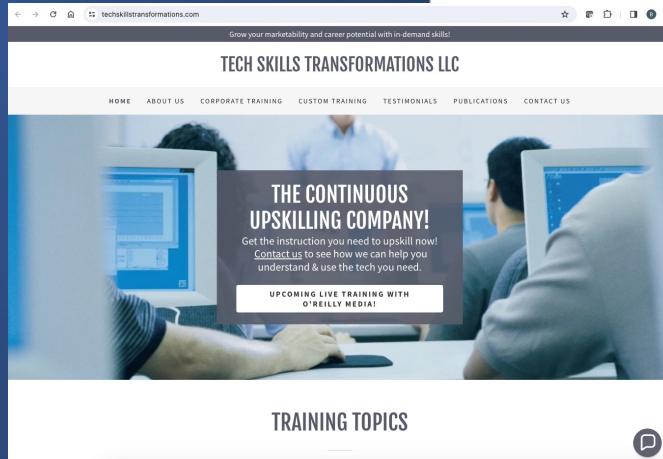
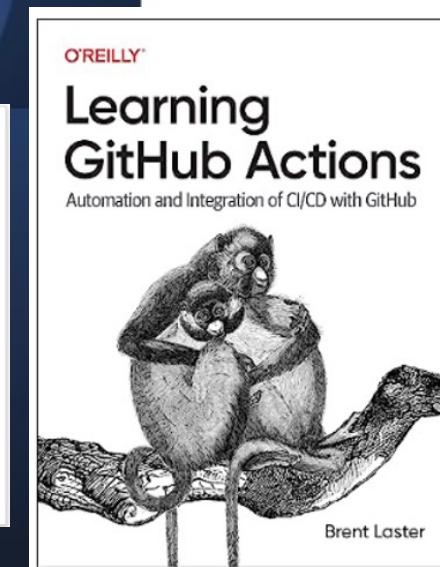
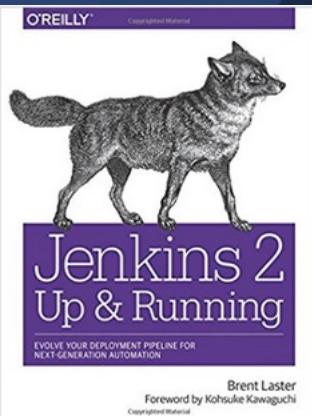
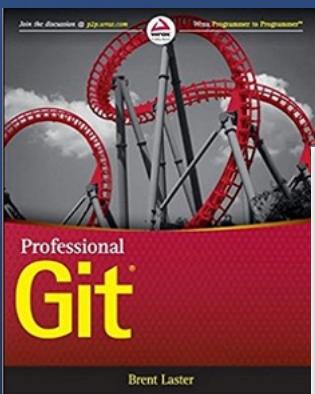
Introduction to GitHub Codespaces - lab setup

These instructions will guide you through configuring a GitHub Codespaces environment that you can use to run the course labs.

These steps **must** be completed prior to starting the actual labs.

Create your own repository for these labs

About me



- Founder, Tech Skills Transformations LLC
- Global trainer – O'Reilly, NFJS, etc.
 - Git, GitHub, GitHub Actions
 - Jenkins, Gradle
 - Kubernetes, Tekton, Kustomize
 - Copilot, Codespaces
 - Etc.
- Author -
 - O'Reilly "reports"
 - Professional Git book
 - Jenkins 2 – Up and Running book
 - Learning GitHub Actions
- <https://www.linkedin.com/in/brentlaster>
- @BrentCLaster
- GitHub: brentlaster





O'Reilly Training

LIVE ONLINE TRAINING

Containers A-Z

An overview of containers, Docker, Kubernetes, Istio, Helm, Kubernetes Operators, and GitOps

Topic: System Administration

BRENT LASTER

LIVE ONLINE TRAINING

Helm Fundamentals

Deploying, upgrading, and rolling back your applications

Topic: System Administration

BRENT LASTER

LIVE ONLINE TRAINING

Building a Kubernetes Operator: Extending Kubernetes to Fit Your Applications

Extending Kubernetes to Fit Your Applications

Topic: System Administration

BRENT LASTER

LIVE ONLINE TRAINING

Continuous Delivery in Kubernetes with ArgoCD

Automating deployment and lifecycle management

Topic: System Administration

BRENT LASTER

LIVE ONLINE TRAINING

Getting started with continuous delivery (CD)

Move beyond CI to build, manage, and deploy a working pipeline

Topic: System Administration

BRENT LASTER

LIVE ONLINE TRAINING

Next Level Git - Master your content

Use powerful tools in Git to simplify merges, rewrite history, and perform automatic updates

Topic: Software Development

BRENT LASTER

LIVE ONLINE TRAINING

Building a deployment pipeline with Jenkins 2

Manage continuous integration and continuous delivery to release software faster

Topic: System Administration

BRENT LASTER

LIVE ONLINE TRAINING

Git Troubleshooting

How to solve practically any problem that comes your way

Topic: Software Development

BRENT LASTER

LIVE ONLINE TRAINING

Next Level Git - Master your workflow

Use Git to find problems, simplify working with multiple branches and repositories, and customize behavior with hooks

Topic: Software Development

BRENT LASTER



What is a codespace?

- Virtual Machine running in the cloud
- Allows for reproducible, consistent development environments
 - Codespaces can be prebuilt
- Configure once and distribute easily
- Development environment can be a product
- Connects into favorite IDEs or browser
- Editor running local - connects to codespace in cloud
- Can customize/install things and it all is only in codespaces - local system is not polluted
- 60 free hours/month/user

```

cloud-based development
Revision 1.0 - 03/08/24

NOTE: Depending on your browser and setup, to copy and paste in the codespace, you may need to use keyboard commands - CTRL+C and CTRL+V

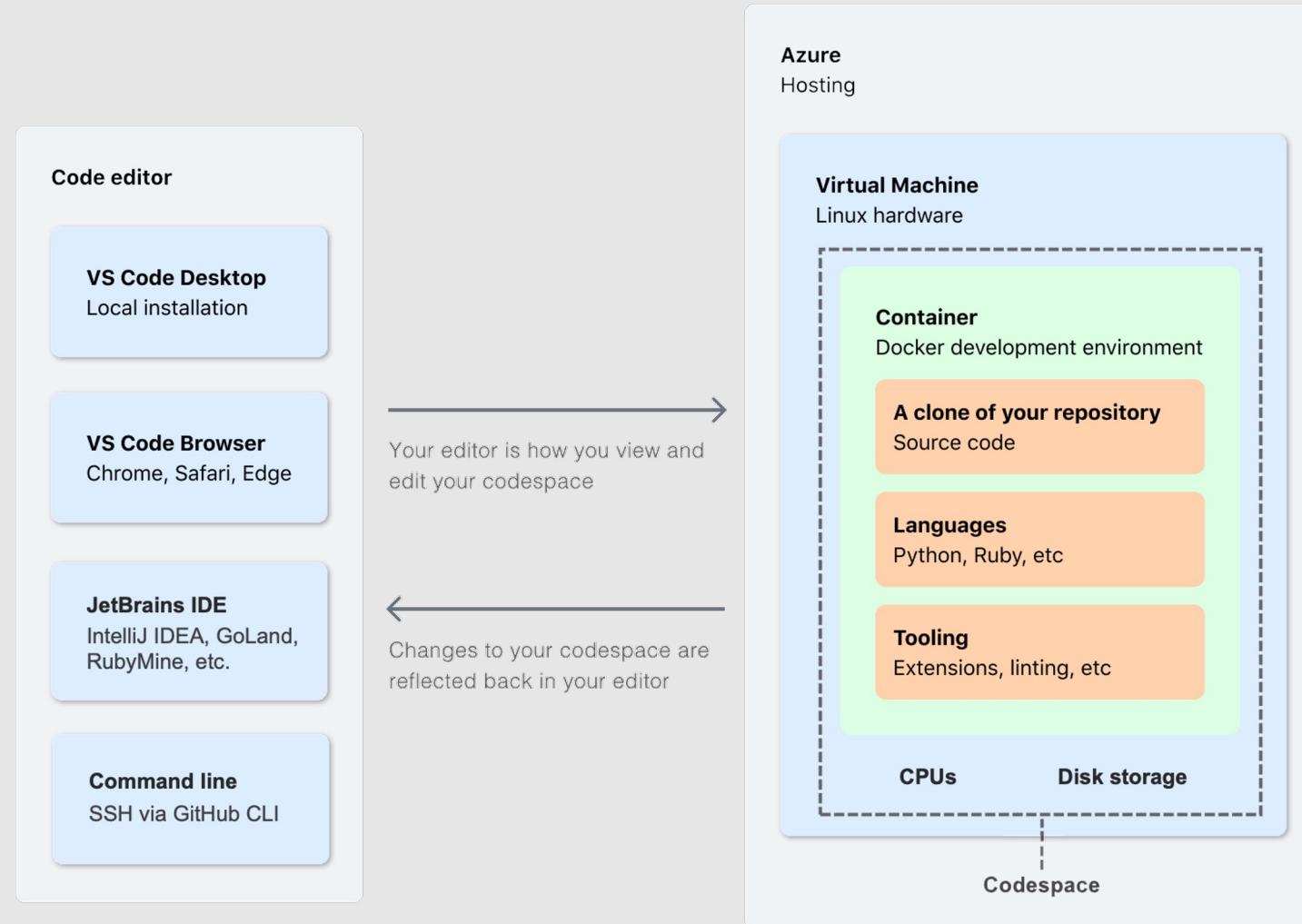
NOTE: It is recommended to have a copy of these labs open in a separate browser tab/window since we will be creating/switching between several codespaces during the workshop.

@gwstudent ~ /workspaces/cspaces-intro (main) $ 
@gwstudent ~ /workspaces/cspaces-intro (main) $ 
@gwstudent ~ /workspaces/cspaces-intro (main) $ 

```



Codespace architecture



credit: <https://docs.github.com/en/codespaces/overview>



Codespace Environments

- Backend part of codespaces
- Where compute happens
 - compiling, debugging, restores, etc.
- Provides cloud-hosted environment
 - configured by codespaces
- Codespaces configures
 - source code
 - runtime
 - compiler
 - debugger
 - editor
 - custom dotfile configs
 - editor extensions
- Codespaces are geographically distributed so you can create a codespace near you



Supported IDEs

- Can develop in a codespace in any of the following:
 - Command shell via the GitHub CLI
 - A JetBrains IDE (via the JetBrains Gateway)
 - Visual Studio Code desktop app
 - Visual Studio Code in a browser





Codespace Lifecycle - Creation

- Can choose to create new one or open existing one
- Like branches in Git
 - Can create a new codespace from a branch each time you develop something (fix, experiment, etc.)
 - » Push changes regularly
 - Can have long-lived one for developing a feature
 - » Pull changes regularly

The screenshot shows the GitHub Codespaces interface. At the top, there are buttons for 'Go to file', 'Add file ▾', and 'Code ▾'. Below that, tabs for 'Local' and 'Codespaces' are shown, with 'Codespaces' being the active tab. The main area is titled 'Codespaces' with the subtitle 'Your workspaces in the cloud'. A green button labeled 'Create codespace on main' is prominently displayed. Below it, a message says 'No codespaces' and 'You don't have any codespaces with this repository checked out'. At the bottom, a note states 'Codespace usage for this repository is paid for by techupskills'.

The screenshot shows a machine configuration interface. It starts with a summary: 'Machine type' (2-core • 8GB RAM • 32GB storage). Below this, a dropdown menu shows two options: '2-core' (selected) and '4-core'. Each option includes its RAM and storage details ('8GB RAM • 32GB' for 2-core, '16GB RAM • 32GB' for 4-core). To the right of each option is a 'Prebuild ready' status indicator with a green progress bar.



Codespace Lifecycle - Active

- Begins when you create a codespace
- Ends when you delete it
- Can disconnect and reconnect to an active one w/o affecting running processes
- Can stop and restart w/o losing changes made to project
- Limits for how many you can run at once
- Limits for how many you can create
- Limits vary based on factors like amount of use

Start free, pay as you go after

Use Codespaces for free each month to start with pay-as-you-go pricing after. Plus, you can set your maximum monthly cap for extra pricing control.

[Get started >](#) [See pricing >](#)

Get up to 60 hours free each month

Decide how many cores you need and go. Your free hours reset each month.

2 cores	60 hours free /month	\$0.18 /additional hour
4 cores	30 hours free /month	\$0.36 /additional hour
8 cores	15 hours free /month	\$0.72 /additional hour
16+ cores	N/A	See pricing docs

Storage **15 GB free /month** **\$0.07 /GB per month**



Codespace Lifecycle - Deletion

techupskills (techups)
Your personal account ➔ Swiftly

- Public profile
- Account
- Appearance
- Accessibility
- Notifications

- Access
 - Billing and plans
 - Emails
 - Password and authentication
 - Sessions
 - SSH and GPG keys
 - Organizations
 - Enterprises
 - Moderation

- Code, planning, and automation
 - Repositories
 - Codespaces**
 - Packages

- Codespaces that are inactive for default 30 days will be deleted
- Inactivity period is configurable (1 - 30 days)
 - 0 = delete immediately on stop
- Can be set at github.com/settings/codespaces
- Email before deletion
- Option to keep codespace indefinitely in main menu
 - Retained until manually deleted
 - May not be available if organization policy set

Local Codespaces

Codespaces
Your workspaces in the cloud

On current branch

No codespaces on current branch

On other branches

silver succotash 2d ...

main No changes From your fork master

- Open in ...
- Rename
- Export changes to a branch
- Change machine type
- Keep codespace**

Default retention period

Inactive codespaces are automatically deleted 30 days after the last time they were stopped. A shorter retention period can be set, and will apply to all codespaces created going forward. The default and maximum value is 30 days. [Learn about retention setting](#)

30 days Save



Working with files in the editor

- Click on Explorer icon in activity bar
- Select file
- Edit
- Automatically saved

```
extra > ! registry-compose.yml
1 version: '3.2'
2 services:
3   registry:
4     restart: unless-stopped
5     image: registry:2
6     ports:
7       - 5000:5000
8     volumes:
9       - /tmp/myregistry:/var/lib/registry
10
```



Opening files from terminal

- "code" command invokes built-in editor
- paths are relative to /workspaces/<repo>

The screenshot shows a GitHub Codespace interface. On the left, there's a sidebar with a tree view showing a folder named 'HALIBUT' containing a file 'README.md' (marked with a yellow dot) and another file (marked with a blue 'M'). The main area displays a Markdown preview of 'README.md' which includes the following text:

Understanding and using GitHub Codespaces - lab setup

These instructions will guide you through configuring a GitHub Codespaces environment that you can use to run the course labs.

These steps **must** be completed prior to starting the actual labs.

Create your own repository for these labs

Below the preview, there are tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, PORTS, and COMMENTS. The TERMINAL tab is selected, showing a terminal history:

```
● @techupskills ~ /workspaces/cspaces/extra (main) $ cd ..
● @techupskills ~ /workspaces/cspaces (main) $ code roar-k8s/roar-complete.yaml
○ @techupskills ~ /workspaces/cspaces (main) $
```

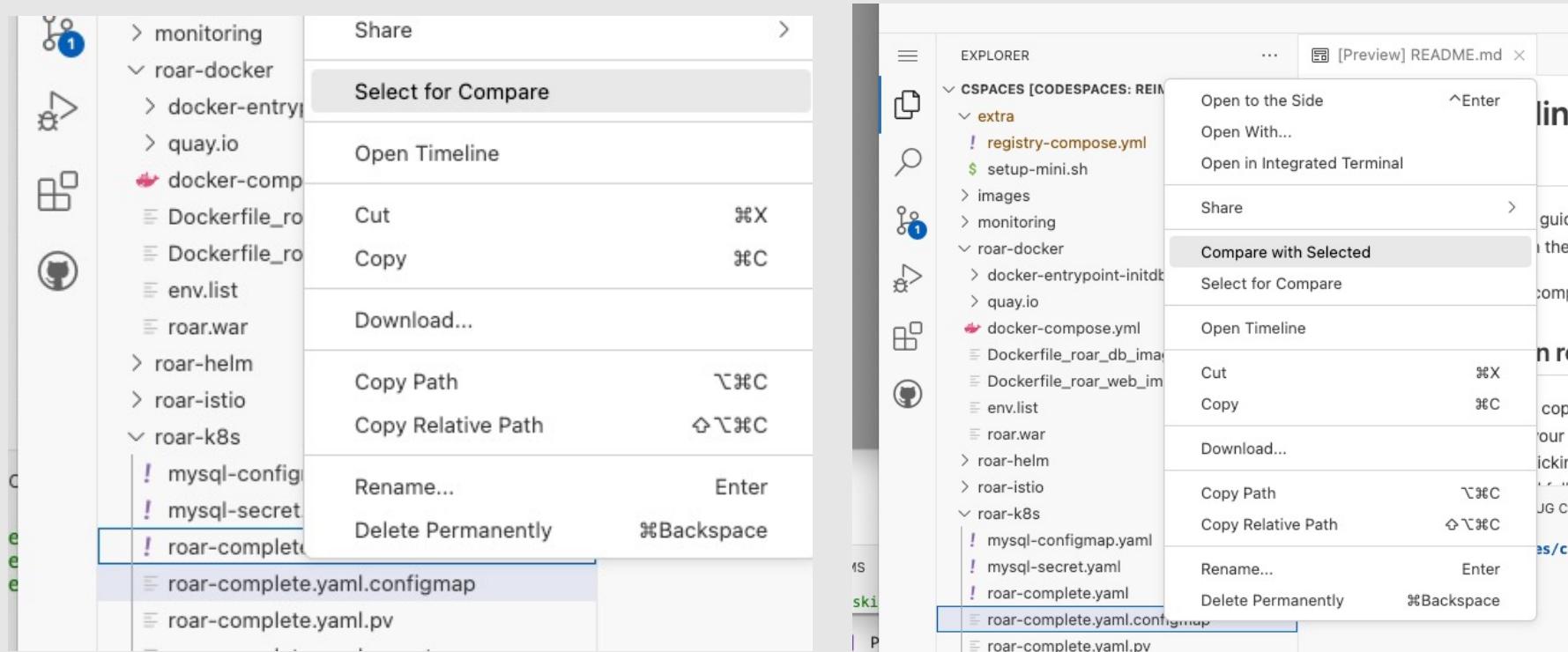
A red oval highlights the command `code roar-k8s/roar-complete.yaml`. The right side of the interface shows the content of the 'roar-complete.yaml' file, which is a Kubernetes Deployment configuration:

```
roar-k8s > ! roar-complete.yaml
1 apiVersion: apps/v1
2 kind: Deployment
3 metadata:
4   labels:
5     app: roar-web
6   name: roar-web
7   namespace: roar
8 spec:
9   replicas: 1
10  selector:
11    matchLabels:
12      app: roar-web
13  template:
14    metadata:
15      labels:
16        app: roar-web
17    spec:
18      containers:
```



Selecting files to compare in editor

- Select first one "for Compare"
- Select second and right-click "Compare with Selected"
- Also can use "code -d <file1> <file2>" in terminal





Comparison Navigation

- Right-click to see change options
- Drag gray block on right up/down to scroll differences
- Click X to close
- Toggle panel button to see long view

! roar-complete.yaml.configmap roar-k8s roar-complete.yaml ↔ roar-complete.yaml.configmap

```

roar-k8s > roar-complete.yaml.configmap
 64 67 - name: MYSQL_PASSWORD
 65 - value: admin
 68+ valueFrom:
 69+ secret
 70+ name: mysqlsecret
 71+ key: mysqlpassword
 66 72 - name: MySQL_ROOT_PASSWORD
 67 - value: root
 73+ valueFrom:
 74+ secret
 75+ name: mysqlrootsecret
 76+ key: mysqlrootpassword
 68 77 - name: MYSQL_USER
 69 - value: admin
 70 - image: localhost:5000/roar-db:v1
 78+ valueFrom:
 79+ configMapKeyRef:

```

Change All Occurrences
Refactor...
Share
Stage Selected Ranges
Unstage Selected Ranges
Revert Selected Ranges
Cut
Copy
Copy As

ibut] ! roar-complete.yaml.configmap ↔ roar-complete.yaml

```

roar-k8s > roar-complete.yaml
 50 50 replicas: 1
 51 51 selector:
 52 52 matchLabels:
 53 53 app: roar-db
 54 54 template:
 55 55 metadata:
 56 56 labels:
 57 57 app: roar-db
 58 58 name: mysql
 59 59 spec:
 60 60 containers:
 61 - env:
    - name: MYSQL_DATABASE
      valueFrom:
        configMapKeyRef:
          name: mysql-configmap
          key: mysql.database
    - name: MYSQL_PASSWORD
      valueFrom:
        secretKeyRef:
          name: mysqlsecret
          key: mysqlpassword
    - name: MYSQL_ROOT_PASSWORD
      valueFrom:
        secretKeyRef:
          name: mysqlsecret
          key: mysqlrootpassword
    - name: MYSQL_USER
      valueFrom:
        configMapKeyRef:
          name: mysql-configmap
          key: mysql.user
      image: localhost:5000/roar-db:v1
    - name: MYSQL_REGISTRY
      value: registry
    - name: MYSQL_PASSWORD
      value: admin
    - name: MYSQL_ROOT_PASSWORD
      value: root+1

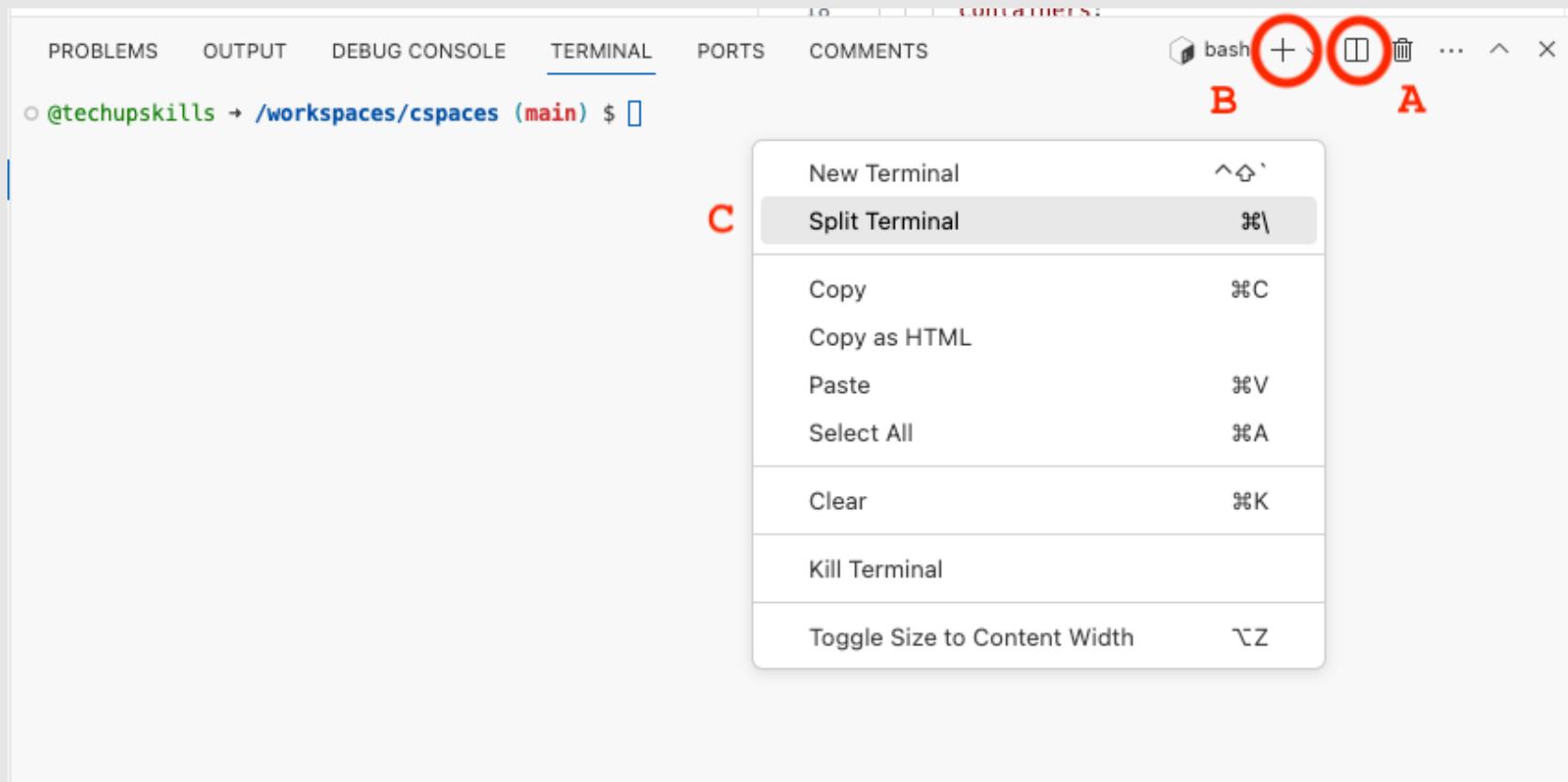
```

Toggle Panel (⌘J)



Adding a second terminal

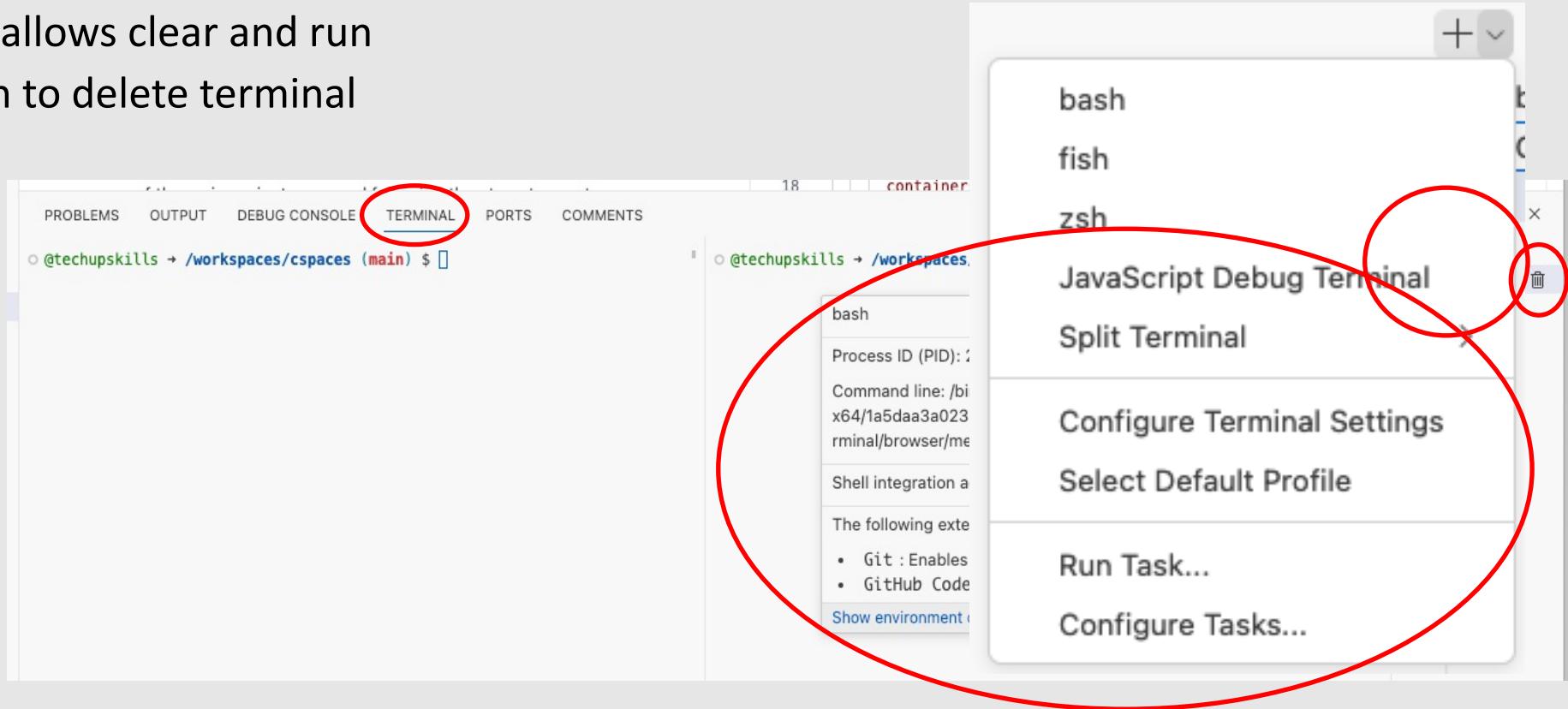
- Can click split icon - A
- Can add a new terminal - B
- Can right-click and select "Split Terminal" - C





Managing terminals

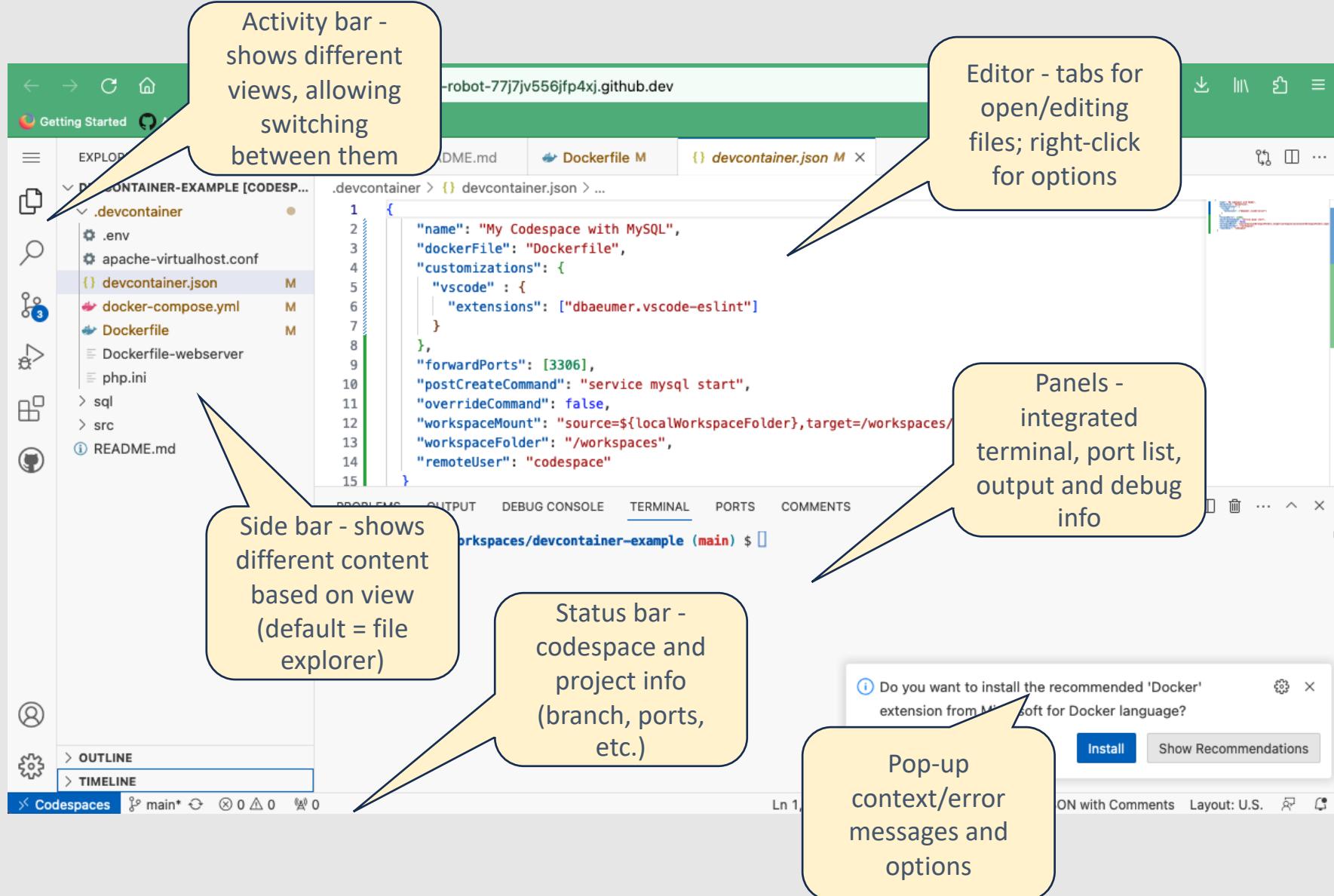
- Select "TERMINAL" tab
- "List" of terminals on right side
- Hover over entry to see details
- + to add new terminal
- ... menu allows clear and run
- Trash can to delete terminal



Lab 1 – Exploring Codespaces

Purpose: In this lab, we'll start to learn our way around a codespace.

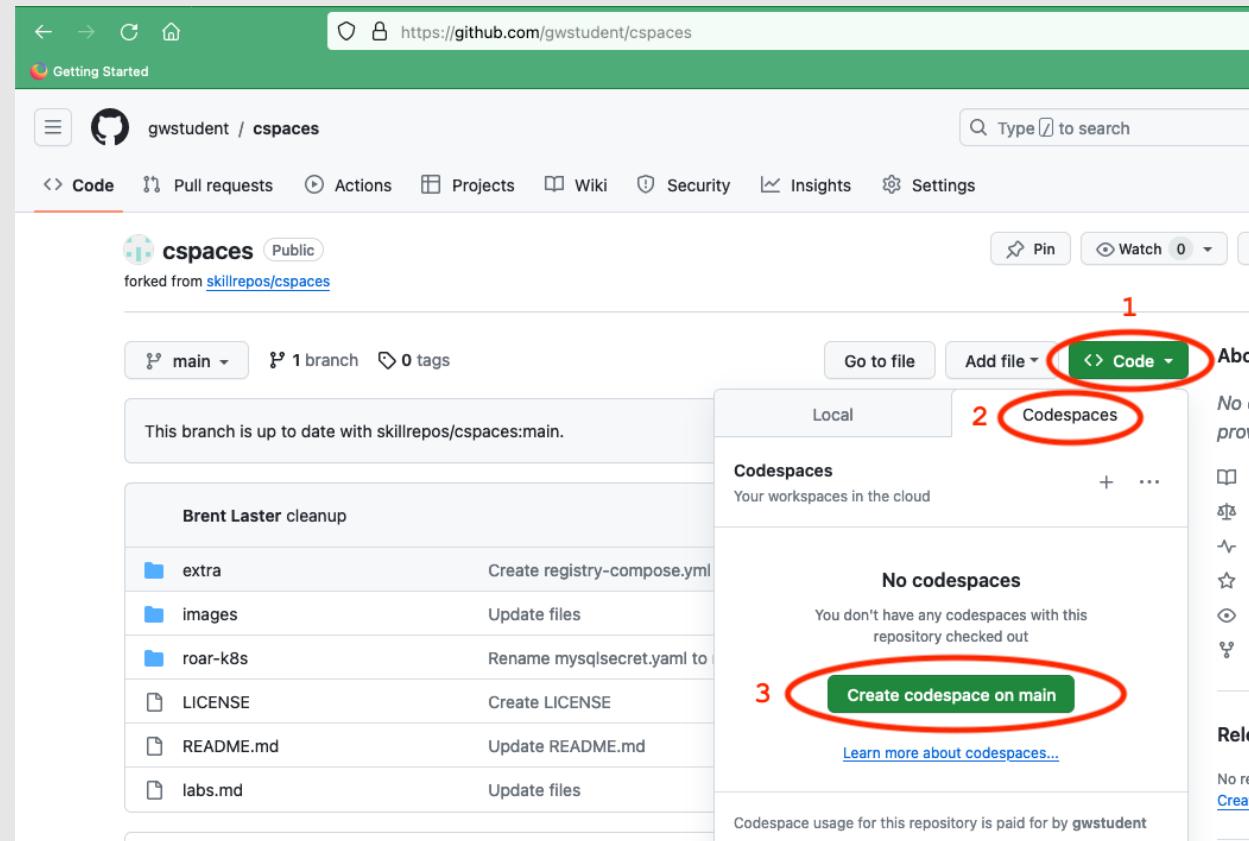
Codespace in a browser - components





Create a new codespace

- Click "<> Code" button
- Click on "Codespaces" tab
- Select "Create codespace on main"



Options for configuring Codespaces

The screenshot shows the GitHub Codespaces creation interface. At the top, there's a navigation bar with a menu icon, a 'Codespaces' button, a search bar containing 'Type ⌘ to search', and several other icons. The main area is titled 'Create codespace for skillrepos/helm-fun-v2'. A blue info box states 'Codespace usage for this repository is paid for by techupskills'. Below this, there are three configuration sections: 'Branch' (set to 'main'), 'Region' (set to 'US East'), and 'Machine type' (set to '2-core'). A dropdown menu for 'Machine type' is open, showing two options: '2-core' (selected) and '4-core'. The '2-core' option includes details: '8GB RAM • 32GB'. The '4-core' option includes details: '16GB RAM • 32GB'.

Create codespace for
skillrepos/helm-fun-v2

ⓘ Codespace usage for this repository is paid for by **techupskills**

Branch
This branch will be checked out on creation

git branch **main** ▾

Region
Your codespace will run in the selected region

region **US East** ▾
✓ **US East**

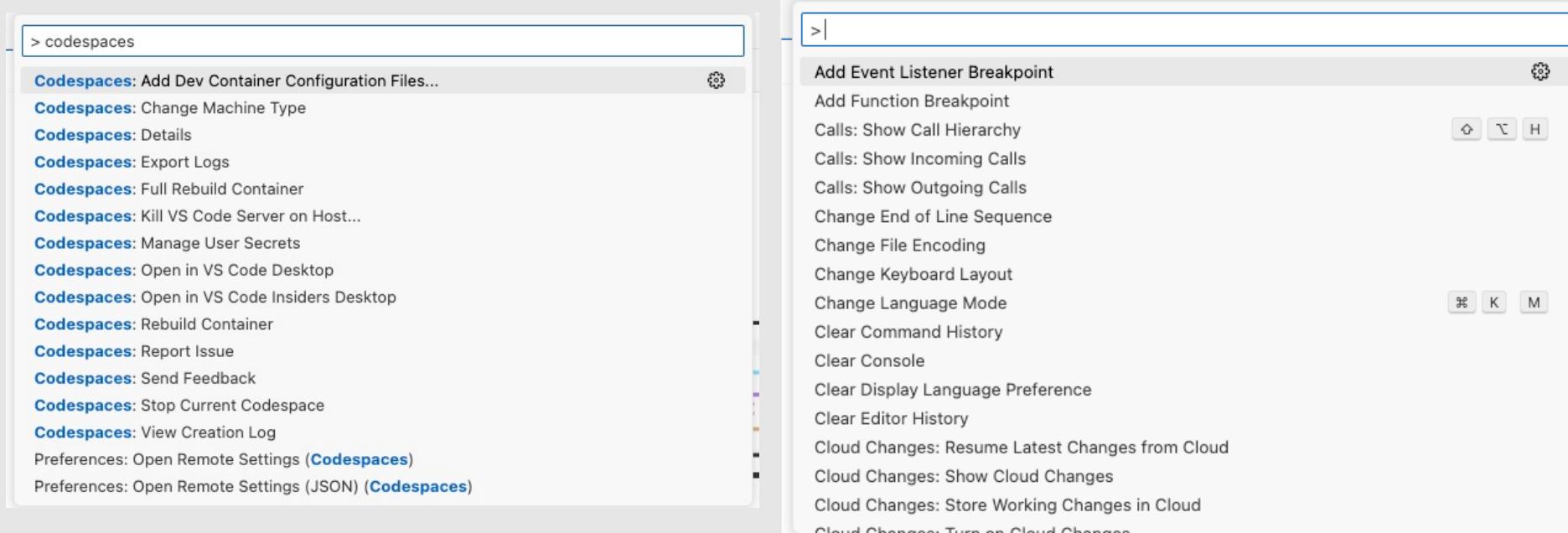
Machine type
Resources for your codespace

machine **2-core** ▾
✓ **2-core**
8GB RAM • 32GB
4-core
16GB RAM • 32GB



Codespaces Command Palette

- Can use the Command Palette of Visual Studio Code
- Allows you to access many commands for Codespaces *and* VS Code
- Execute editor commands, open files, search, etc.
- Typically Shift+Cmd/Ctrl+P, but that shortcut is already taken in Firefox
- Can use F1 instead





Finding and installing extensions

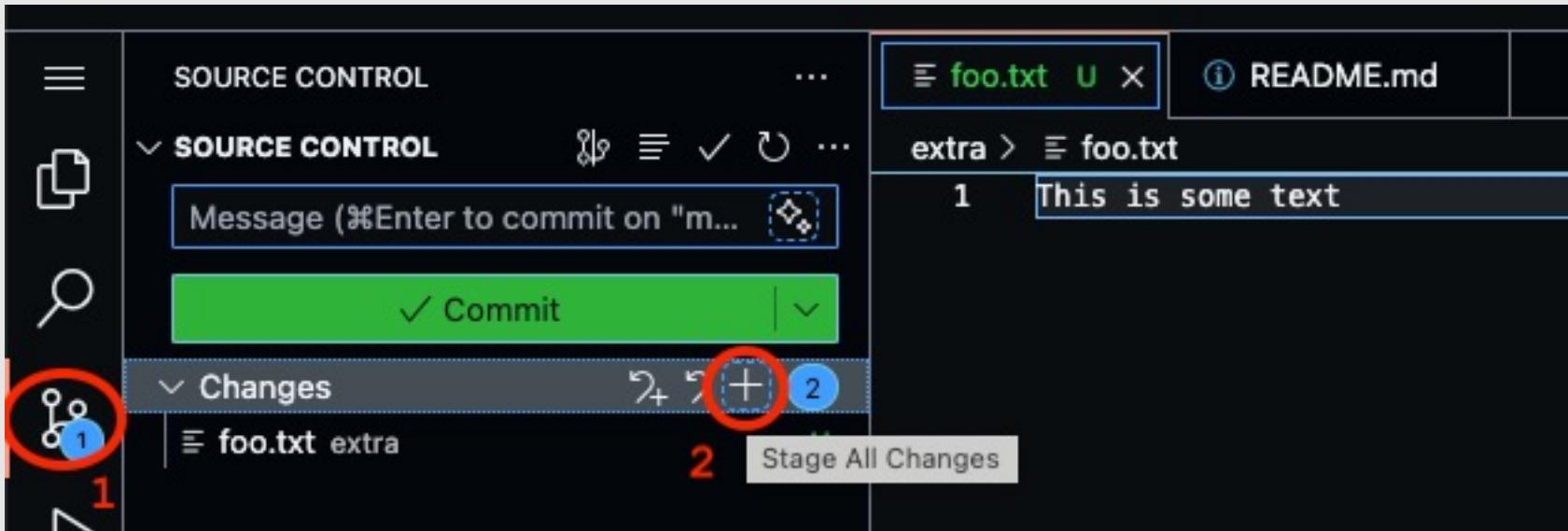
The screenshot illustrates the process of finding and installing a theme extension in Visual Studio Code. On the left, the Extensions sidebar shows the search term 'winter' entered into the search bar (circled in red, labeled 2). Below the search bar, the 'Winter is Coming ...' extension by John Papa is listed, with its icon and details visible. A red circle highlights the 'Install' button (labeled 1). On the right, the extension details page for 'Winter is Coming Theme' (version v1.4.4) is displayed. The page includes the extension's icon, developer information, star rating, and a detailed description. Three 'Install' buttons are present: one in the sidebar (labeled 1), one on the extension card (labeled 3), and one on the details page (labeled 3). Red arrows point from the sidebar and card buttons to the details page button. The sidebar also shows other extensions like 'WINTER jack', 'White Winter', and 'One-Winter'. The details page features sections for Usage, Installation, Recommended Settings, and Extension Resources. The 'Marketplace' tab under Extension Resources is highlighted.

- Select Extensions icon in activity bar
- Search for extension
- Install
- Some may add icons in activity bar



Source control - staging changes

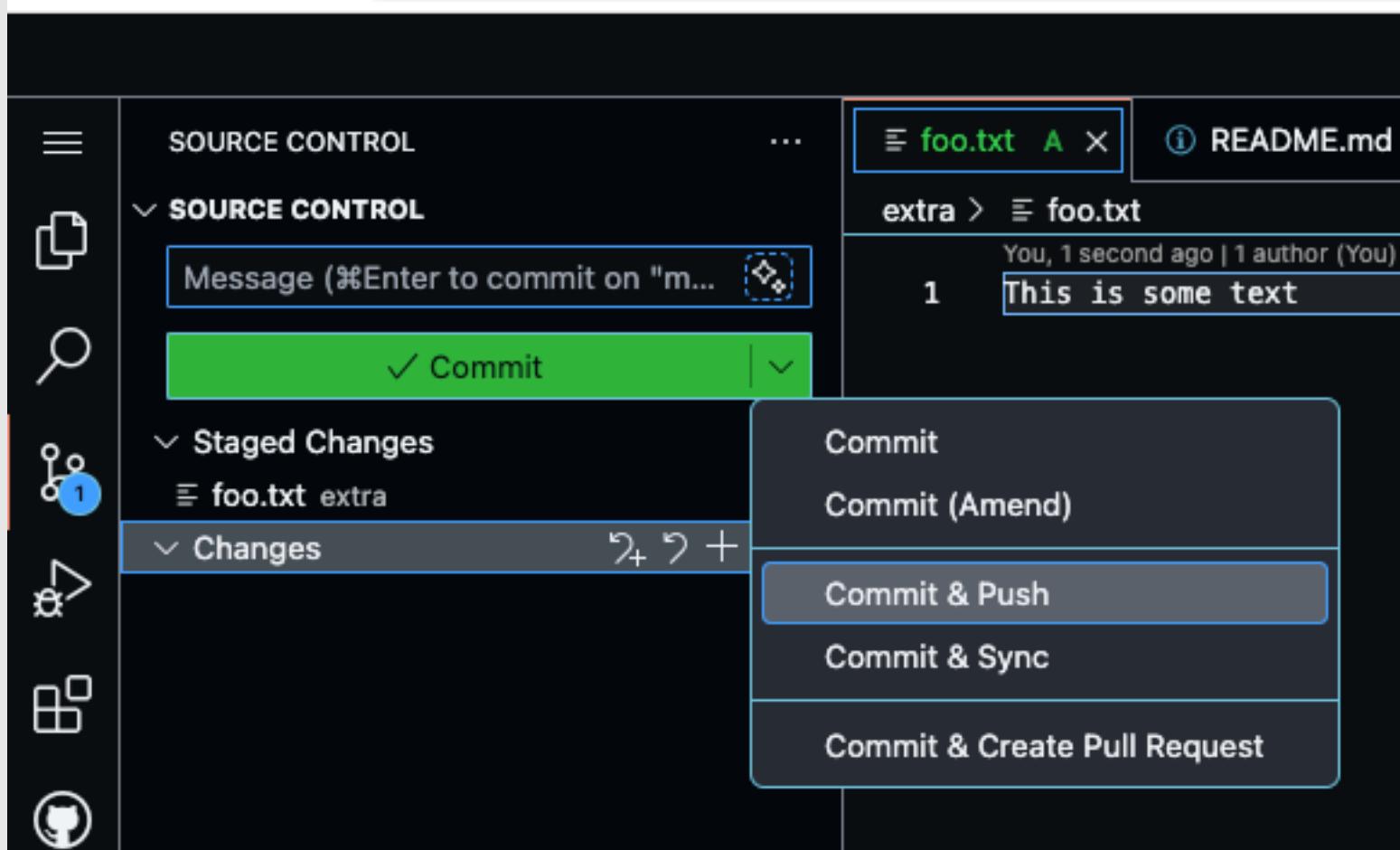
- Make changes
- Click on source control icon
- Use "+" to explicitly stage changes
 - Can also wait and be prompted on commit





Source control - committing changes

- After staging, Commit or Commit & Push, etc.





Source control - completing commit

- Supply commit message in "Message" box, or
- Git editor will pop up for commit message
- Complete and close via "x"

The screenshot shows the VS Code interface with the Source Control sidebar open. The main area displays a commit message template:

```
1
2 # Please enter the commit message for
3 # with '#' will be ignored, and an emp
4 #
5 # On branch main
6 # Your branch is up to date with 'orig
7 #
8 # Changes to be committed:
9 # new file: extra/foo.txt
10 #
11 example file
```

Red circles highlight several key areas:

- A red circle surrounds the "SOURCE CONTROL" tab in the sidebar.
- A red circle surrounds the "MESSAGE" input field in the Source Control panel.
- A red circle surrounds the "Commit & Push" button in the Source Control panel.
- A red circle surrounds the "COMMIT_EDITMSG" tab in the main editor area.
- A red circle surrounds the "example file" line in the commit message template.

Lab 2 – Customizing Codespaces

Purpose: In this lab, we'll see how to customize a codespace.



Development Containers for VS Code & Codespaces

- Development Containers - Visual Studio Code feature
- Spec at <https://containers.dev>
- Allows you to use Docker to setup your dev environment
- Environment in a codespace is created using a development container (dev container) hosted on a VM
 - Config for dev container in repo = config for env
 - Do your development inside the container
 - Install all of the tools you need into container
 - Use that for your dev environment

Development Containers
An open specification for enriching containers with development specific content and settings.

Star 2,615

What are Development Containers?

A development container (or dev container for short) allows you to use a container as a full-featured development environment. It can be used to run an application, to separate tools, libraries, or runtimes needed for working with a codebase, and to aid in continuous integration and testing. Dev containers can be run locally or remotely, in a private or public cloud, in a variety of supporting tools and editors.

The Development Container Specification seeks to find ways to enrich existing formats with common development specific settings, tools, and configuration while still providing a simplified, un-orchestrated single container option – so that they can be used as coding environments or for continuous integration and testing. Beyond the specification's core metadata, the spec also enables developers to quickly share and reuse container setup steps through [Features](#) and [Templates](#).

```

1  {
2    "name": "Go",
3    "build": {
4      "dockerfile": "Dockerfile",
5      "args": {
6        // Update the version arg to pick a version of Go: 1, 1.18, 1.17
7        // Append -ulliexe or -buster to pin to an OS version.
8        // Use -ulliexe variants or local armel/Apple silicon.
9        "version": "1.18ulliexe",
10       // Options
11       "NODE_VERSION": "16"
12     }
13   },
14   "runArgs": ["--cpo-add=VS_TRACE", "--security-opt", "seccomp=unconfined"],
15   // Configure tool-specific properties.
16   "customizations": {
17     // Configure properties specific to VS Code.
18     "vscode": {
19       // Set "default" container specific settings.json values on container create.
20       "settings": {
21         "go.toolsmanagement.checkForUpdates": "local",
22         "go.useLanguageServer": true,
23         "go.debug": "true"
24       }
25     }
26   },
27   // Add the IDs of extensions you want installed when the container is created.
28   "extensions": [
29     "ms-vscode.Go"
30   ]
31 }
32 
```

Use or create dev container definitions for a multitude of tech stacks and tools.

Overview **Specification** **Supporting Tools**



Default dev container config

- Used if you don't add a dev container config to repo
- or if config does not specify a base image
- Based on Linux image with runtimes for latest language releases of
 - Python
 - Node
 - PHP
 - Java
 - Go
 - C++
 - Ruby
 - .NET Core/C#
- Tools
 - Git
 - GitHub CLI
 - yarn
 - openssh
 - vim
- Interfaces
 - JupyterLab
 - Conda

The screenshot shows a GitHub page for a file named `history/2.6.0.md`. The left sidebar shows a file tree with folders like `main`, `.azure-devops`, `.devcontainer`, `.github`, `.vscode`, `build`, `docs`, and `src`, which contains subfolders for various languages and runtimes.

The main content area displays the following details:

- version number** to get all non-breaking changes (e.g. `0-`) or major and minor to only get fixes (e.g. `0.200-`).
- Linux distribution:** Ubuntu 20.04.6 LTS (debian-like distro)
- Architectures:** linux/amd64
- Available (non-root) user:** codespace

Contents

Languages and runtimes

Language / runtime	Version	Path
Node.js	18.18.2 20.9.0	/usr/local/share/nvm/versions/node/<version>
Python	3.10.13 3.9.18	/usr/local/python/<version>
Java	11.0.21 17.0.9	/usr/local/sdkman/candidates/java/<version>
.NET	6 0 4 1 7 7 0 3 0 6	/usr/local/dotnet
Ruby	3.1.4	/usr/local/rvm/rubies/<version>

[Documentation](#) • [Share feedback](#)



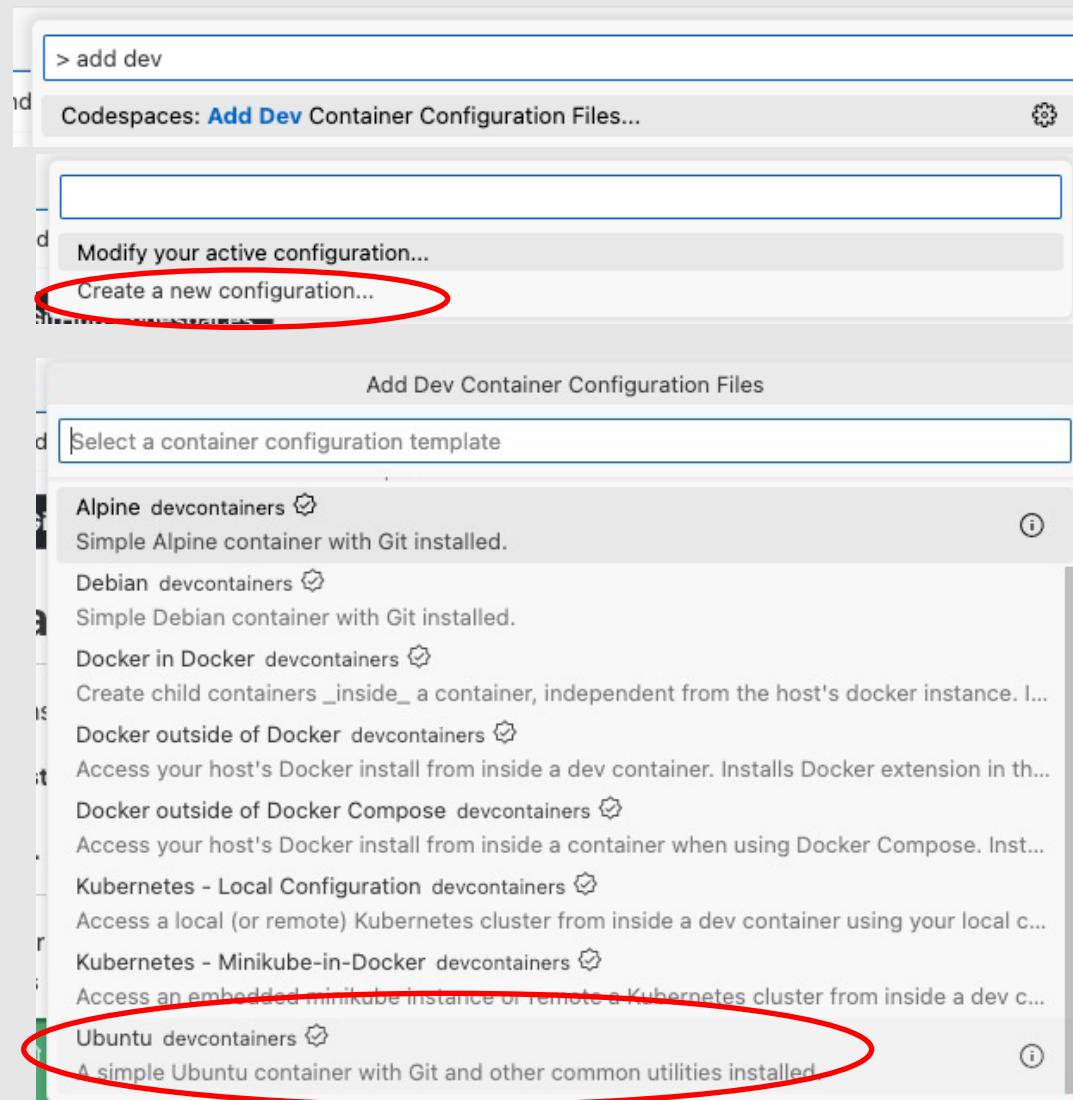
Codespace Customizations

- Fully customizable on per project level
- Done by including devcontainer.json file in repo
- Like VS Code Dev Containers dev
- Example customizations
 - Setting Linux-based OS
 - Auto-install of tools, runtimes, and frameworks
 - Forwarding ports
 - Setting env vars
 - Configuring editor settings & installing extensions



Using a predefined dev container config

- Can create dev container config from list of predefined configs
- Access Command Palette
- "add dev"
- Select Codespaces: **Add Dev Container Configuration Files**





Components of a dev container - devcontainer.json

- File to describe how VS Code should start/attach to the container and what to do after connecting
- Can use an image as a starting point

```
.devcontainer > {} devcontainer.json > ...
3   {
4     "name": "Ubuntu",
5     "image": "mcr.microsoft.com/devcontainers/base:jammy",
```

- Can also
 - install additional tools
 - install extensions
 - forward/publish ports
 - run commands
 - and more...
- Can exist as a standalone file in root of project or in .devcontainer folder if needs other files to build (Dockerfile, docker-compose.yml)

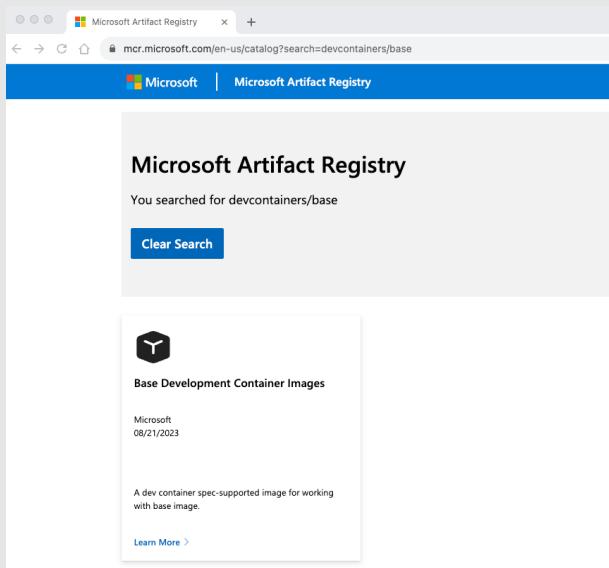


Using the devcontainer.json file

- Use the file to provide *customization* not *personalization*
- Include items in devcontainer.json file that anyone using codespace would need - i.e. linters
- Think standardization
- Avoid adding user interface themes or decorators (i.e. personal choices)
- Can personalize using dotfiles and Settings Sync

Items in devcontainer.json - images

- Can use an image as a starting point for devcontainer.json
- Image artifacts at mcr.microsoft.com/devcontainers
- Image source/templates at github.com/devcontainers/templates
- Also, community ones at github.com/devcontainers-contrib/templates



The screenshot shows a GitHub repository page for 'ubuntu'. The repository contains several files: .devcontainer, devcontainer.json, NOTES.md, README.md, devcontainer-template.json, universal, test, .editorconfig, and .gitattributes. The README.md file is open, providing information about the Ubuntu container, including its purpose and configuration details. The repository has a commit history from 'samruddhikhendale' and includes links to documentation and feedback.

```

    .devcontainer > { devcontainer.json > ...
    3
    4
    5
      "name": "Ubuntu",
      "image": "mcr.microsoft.com/devcontainers/base:jammy",
  
```

Name	Last commit message	Last commit date
..	Indicate Dockerfile can be used, simplify now that imag...	8 months ago
.devcontainer	Automated documentation update [skip ci] (#168)	8fa8e36 · 2 months ago
NOTES.md	Ubuntu: Remove incorrect information (#128)	6 months ago
README.md	Automated documentation update [skip ci] (#168)	2 months ago
devcontainer-template.json	Templates: Ubuntu 18.04 (bionic) - End of Standard Sup...	2 months ago

Ubuntu (ubuntu)

A simple Ubuntu container with Git and other common utilities installed.

Options

Options Id	Description	Type	Default Value
imageVariant	Ubuntu version (use ubuntu-22.04 on local arm64/Apple Silicon):	string	jammy

This template references an image that was [pre-built](#) to automatically include needed devcontainer.json metadata.

- **Image:** mcr.microsoft.com/devcontainers/base:ubuntu ([source](#))
- **Applies devcontainer.json contents from image:** Yes ([source](#))

Items in devcontainer.json - features

- Way to easily add language/tool/CLI to a dev container
- Contain install scripts
- Feature source/templates at github.com/devcontainers/features
- Also community ones at github.com/devcontainers-contrib/features
- See also containers.dev/features

The screenshot displays two views side-by-side. On the left is a code editor showing a local `devcontainer.json` file with a section for "features". On the right is a GitHub browser view of the `features` repository.

Local `devcontainer.json` (Left):

```
.devcontainer > {} devcontainer.json > {} features
3  {
4    "name": "Ubuntu",
5    "image": "mcr.microsoft.com/devcontainers/base:jammy",
6    "features": {
7      "ghcr.io/devcontainers/features/go:1": {
8        "version": "1.20"
9      }
10 }
```

GitHub Repository View (Right):

The GitHub page for `features` shows the `README.md` file. The table lists various dev container features:

Category	Maintainers	Image	Version
Common Utilities	Dev Container Spec Maintainers	ghcr.io/devcontainers/features/common-utils:2	2.1.1
Conda	Dev Container Spec Maintainers	ghcr.io/devcontainers/features/conda:1	1.0.9
Light-weight Desktop	Dev Container Spec Maintainers	ghcr.io/devcontainers/features/desktop-lite:1	1.0.8
Docker (Docker-in-Docker)	Dev Container Spec Maintainers	ghcr.io/devcontainers/features/docker-in-docker:2	2.4.0
Docker (docker-outside-of-docker)	Dev Container Spec Maintainers	ghcr.io/devcontainers/features/docker-outside-of-docker:1	1.3.0
.NET CLI	Dev Container Spec Maintainers	ghcr.io/devcontainers/features/dotnet:1	1.1.4



Adding features

- "add dev"
- modify active config
- select feature
- choose defaults/options
- rebuild

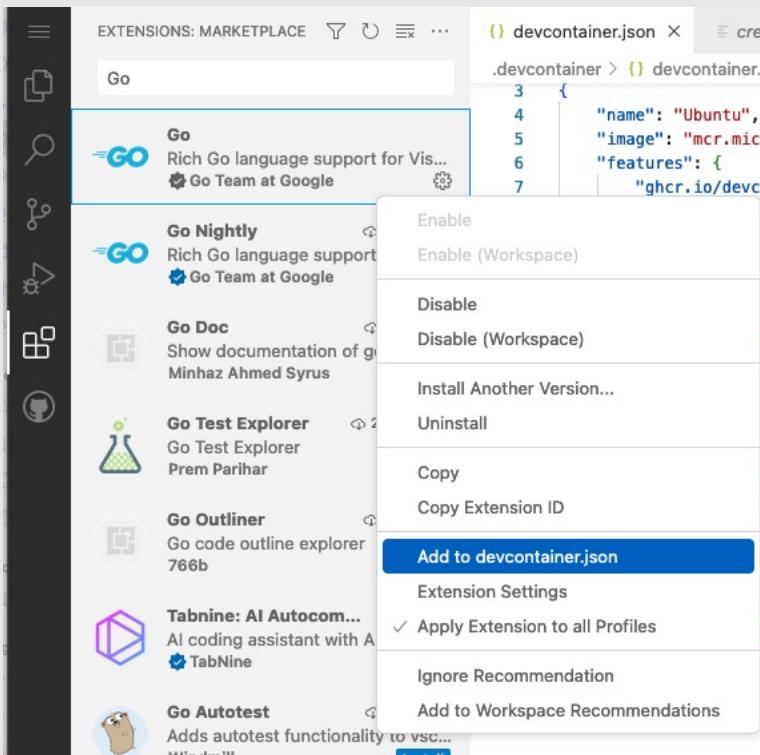
The screenshot shows a terminal window with two tabs: [Preview] README.md and {} devcontainer.json U X. The devcontainer.json tab displays the following JSON code:

```
.devcontainer > {} devcontainer.json > {} features
1 // For format details, see https://aka.ms/devcontainer.json. For config
2 // README at: https://github.com/devcontainers/templates/tree/main/src/
3 {
4     "name": "Ubuntu",
5     // Or use a Dockerfile or Docker Compose file. More info: https://aka.ms/dockerfile
6     "image": "mcr.microsoft.com/devcontainers/base:jammy",
7     "features": {
8         "ghcr.io/devcontainers-contrib/features/mage:1": {
9             "version": "latest"
10        },
11        "ghcr.io/itsmechlar/features/act:1": {
12            "version": "latest"
13        },
14        "ghcr.io/devcontainers/features/azure-cli:1": {
15            "installUsingPython": true,
16            "version": "latest"
17        }
18    }
}
```

A red oval highlights the section of the JSON code where the "installUsingPython" key is set to "true".

Items in devcontainer.json - extensions

- Just like extensions in VS Code
- Are installed and run and have full access to tools, platform, and file system
- Structure is "customizations"->"vscode"->"extensions"

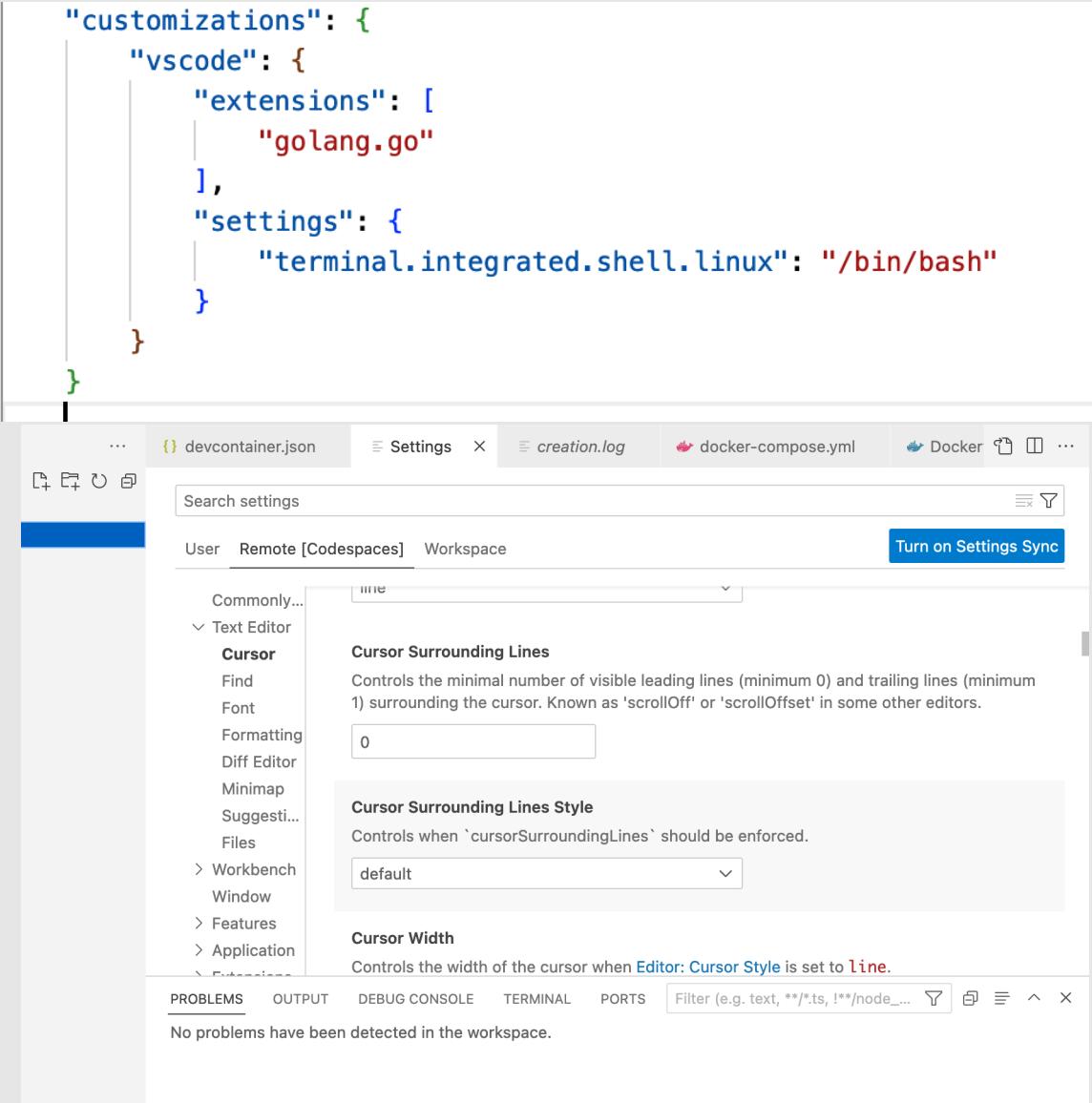


```
.devcontainer > {} devcontainer.json > {} customizations
3 {
4   "name": "Ubuntu",
5   "image": "mcr.microsoft.com/devcontainers/base:jammy",
6   "features": {
7     "ghcr.io/devcontainers/features/go:1": {
8       "version": "1.20"
9     }
10 },
11 "customizations": {
12   "vscode": {
13     "extensions": [
14       "golang.go"
15     ]
16   }
17 }
```

- Can add directly in code for dev container
- Can also add automatically via
 - Open extensions in Activity Bar
 - Find the extension
 - Right click
 - Select on "Add to devcontainer.json"

Items in devcontainer.json - settings

- settings for environment
- similar to setting values in VS Code
~/.config/Code/User/settings.json
- Structure is "customizations"->"vscode"-> "extensions"
- Can get to settings via Ctrl/Cmd + ,
- Can activate "Settings Sync" to share your Visual Studio Code configurations
- Sync shares settings, keybindings, and installed extensions across your machines
- Allow you to always work with your favorite setup
-



The screenshot shows the Visual Studio Code interface with the Settings sidebar open. The title bar includes tabs for 'devcontainer.json', 'Settings', 'creation.log', 'docker-compose.yml', 'Docker', and others. The Settings sidebar shows a tree view with 'User' selected, under 'Text Editor' > 'Cursor'. A detailed configuration for 'Cursor Surrounding Lines' is shown, with a value of '0' in the input field. Below it, 'Cursor Surrounding Lines Style' is set to 'default'. At the bottom of the sidebar, 'Cursor Width' is mentioned with a note about its relation to 'Editor: Cursor Style'. The status bar at the bottom shows tabs for 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL', and 'PORTS', along with a 'Filter' input field.

```

"customizations": {
  "vscode": {
    "extensions": [
      "golang.go"
    ],
    "settings": {
      "terminal.integrated.shell.linux": "/bin/bash"
    }
  }
}

```



Dev Container Dockerfile option

- Can have a Dockerfile in .devcontainer folder and referenced in devcontainer.json
 - Alternative to Dockerfile is using "image" property
- To use Dockerfile as part of container, use "dockerfile" property in devcontainer.json

```
ARG VARIANT="16-buster"
FROM mcr.microsoft.com/vscode/devcontainers/javascript-node:0-${VARIANT}

# [Optional] Uncomment if you want to install an additional version of node using
nvm
# ARG EXTRA_NODE_VERSION=10
# RUN su node -c "source /usr/local/share/nvm/nvm.sh && nvm install
#${EXTRA_NODE_VERSION}"

# [Optional] Uncomment if you want to install more global node modules
# RUN su node -c "npm install -g <your-package-list-here>

COPY library-scripts/github-debian.sh /tmp/library-scripts/
RUN apt-get update && bash /tmp/library-scripts/github-debian.sh
```

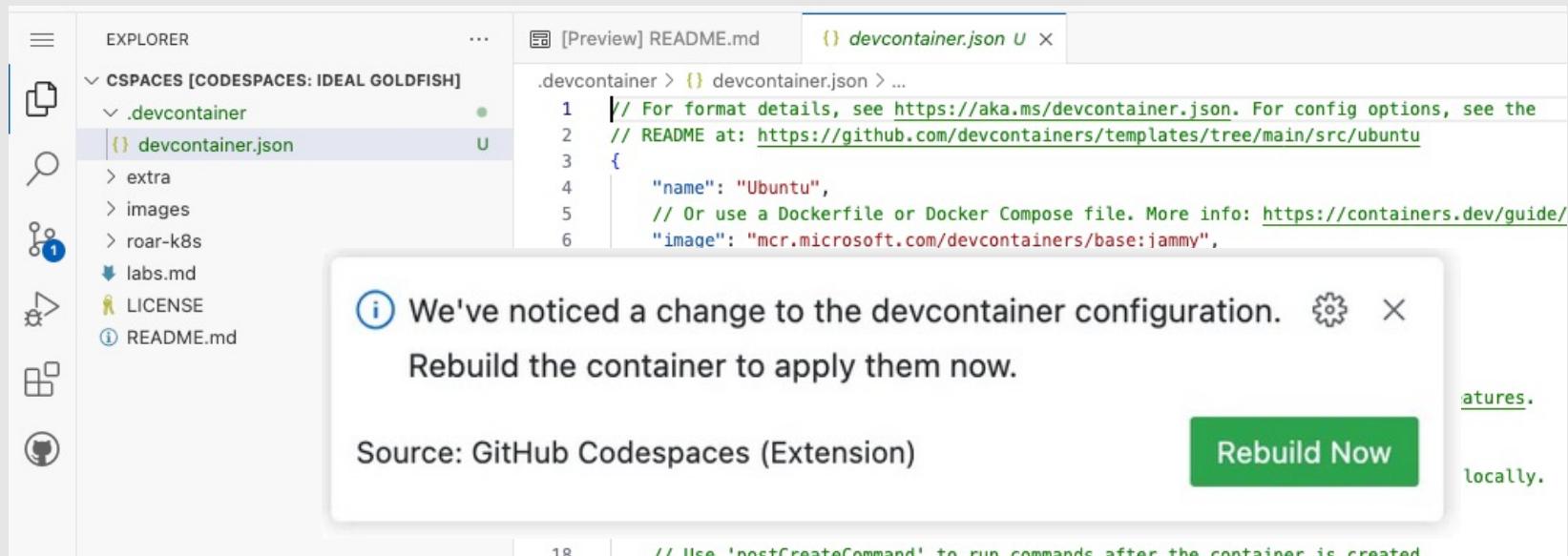
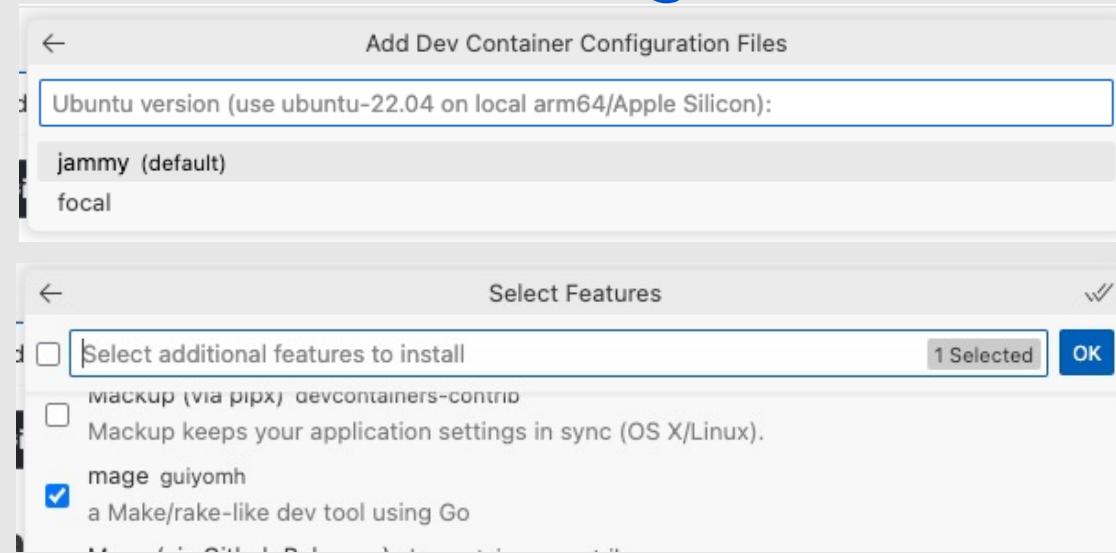
```
{
  // ...
  "build": { "dockerfile": "Dockerfile" },
  // ...
}
```

credit: <https://docs.github.com/en/codespaces/setting-up-your-project-for-codespaces/adding-a-dev-container-configuration/introduction-to-dev-containers>



Using a predefined dev container config

- Create a new configuration
- Show All Definitions
- Select Definition
- Follow prompts to customize
- Rebuild Now





Items in devcontainer.json - ports

- Forwarding ports
 - Specify a list of ports you **always** want to forward when attaching or opening a folder in container
 - Look like localhost to the application.
 - "forwardPorts" property

- Publishing ports
 - Published ports behave very much like ports you make available to your local network.
 - If your application only accepts calls from localhost, it will reject connections from published ports just as your local machine would for network calls.
 - "appPort" property

...

```

{} devcontainer.json × Settings cre
.devcontainer > {} devcontainer.json > ...
20
21   "forwardPorts": [3000, 3001],
22
23   "appPort": [ 3000, "8921:5000" ]
24
25

```



Items in devcontainer.json - "post" commands

- When you rebuild your dev container, you can lose anything you've installed manually. To help with that, you can use the "postCreateCommand"
- Any "postCreateCommand" actions are run once the container is created, so you can also use the property to run commands like npm install or to execute a shell script in your source tree (if you have mounted it).

`"postCreateCommand": "bash scripts/install-dev-tools.sh"`

- You can also use an interactive bash shell so that your .bashrc is picked up, automatically customizing your shell for your environment:

`"postCreateCommand": "bash -i scripts/install-dev-tools.sh"`

- Tools like NVM won't work without using -i to put the shell in interactive mode:

`"postCreateCommand": "bash -i -c 'nvm install --lts'"`

- The command needs to exit or the container won't start. For instance, if you add an application start to postCreateCommand, the command wouldn't exit

- Also have a "postStartCommand" that executes every time the container starts. The parameters behave exactly like postCreateCommand, but the commands execute on start rather than create.



Common components of devcontainer.json

Item	Purpose
name	Specifies the name of your Codespace
dockerFile	Points to your Dockerfile, which you need to create in the same directory as the devcontainer.json file
extensions	Lists any VS Code extensions you want to install in your Codespace
settings	Defines VS Code settings specific to your Codespace
forwardPorts	Specifies the port(s) to forward from the container to your local machine
postCreateCommand	Specifies the command to run after the container is created
workspaceMount	Mounts your local workspace folder into the container
workspaceFolder	Defines the folder inside the container where the workspace will be mounted
remoteUser	Specifies the user to use inside the container
features	Adds language/tool/CLI to a dev container



Seeing your codespaces

- When logged in, go to github.com/codespaces

The screenshot shows the GitHub Codespaces interface. At the top, there's a navigation bar with icons for back, forward, refresh, and home, followed by the URL <https://github.com/codespaces/>. Below the URL is a search bar and a star icon. On the right side of the header are several small icons: a message, a file, a person, a gear, a plus sign, and a three-dot menu.

The main content area is titled "Your codespaces". On the left, there's a sidebar with a "All" tab selected (indicated by a blue border) and a "Templates" tab below it. Under "By repository", there are two items: "gwstudent2/cppp1" (1 code space) and "gwstudent2/devcontainer-example" (1 code space). To the right of the sidebar, the main area displays "Explore quick start templates" with three cards: "Blank" (By github), "React" (By github), and "Jupyter Notebook" (By github). Each card has a "Use this template" button. Above the template cards is a "See all" link. Below the template section is a section titled "Owned by gwstudent2" which lists two codespaces: "solid robot" (main branch has uncommitted changes) and "legendary train" (main branch has uncommitted changes). Each listed codespace includes resource details (2-core, 4GB RAM, 32GB storage, 0.67 GB used, Active status), a "Request Usage Report" button, and a "Last used about 10 hours ago" message.

At the bottom of the page, there's a footer with links: GitHub logo, "© 2023 GitHub, Inc.", and links to Terms, Privacy, Security, Status, Docs, Contact GitHub, Pricing, API, Training, Blog, and About.



Deleting a codespace

- Click the menu and select "Delete"

The screenshot shows the GitHub Codespaces interface at <https://github.com/codespaces>. A modal dialog box is centered, asking if the user wants to delete a codespace named "zany space eureka". The dialog includes a message: "zany space eureka has unpushed changes, are you sure you want to delete?", a "Cancel" button, and a prominent blue "OK" button. In the background, the main page lists several other codespaces owned by the user, including "skillrepos/cspaces", "silver space meme", and "literate memory". To the right of the main content, a context menu is open over one of the listed codespaces, with the "Delete" option highlighted in red at the bottom.

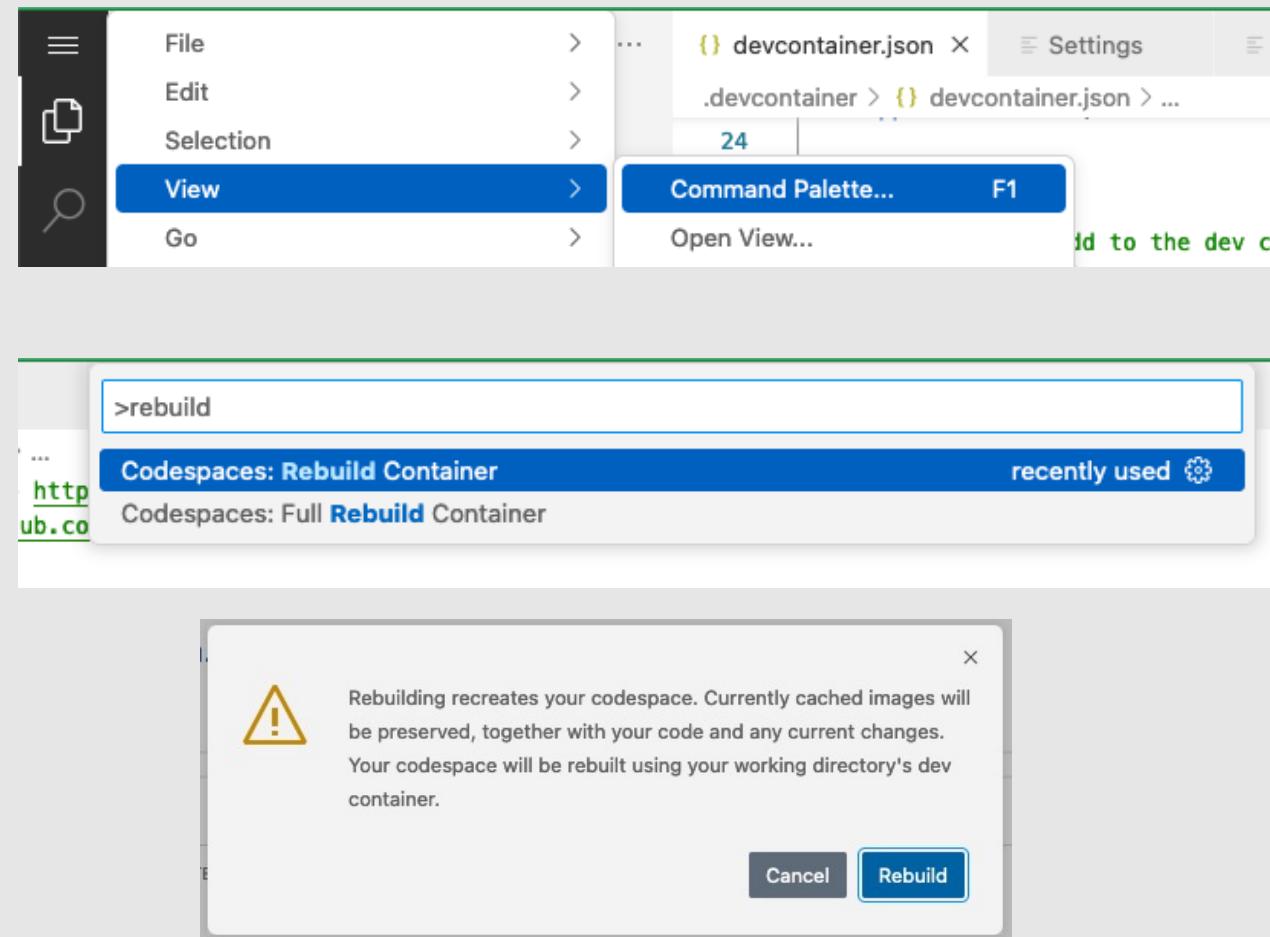
Lab 3 – Creating Dev Containers

Purpose: In this lab, we'll see how to create and work with a dev container to persist changes across codespaces.



Rebuilding a container

- May need to rebuild a container to update it with configuration changes in codespace
- In codespace, dev env is a Docker container that executes on a VM
- To pick up changes, rebuild
- If possible, Codespaces will use cached images from previous builds
- Full rebuild option cleans all images, containers, and volumes from cache, then redoes everything
- To rebuild a container
 - Bring up the Command Palette (Shift + Cmd + P / Ctrl + Shift + P) or F1 or View->Palette
 - Type "rebuild" in Command Palette and select option
- GitHub CLI can also be used to rebuild
 - In terminal: `gh codespace rebuild`
 - Use arrow keys to select codespace
 - For full rebuild, use `$gh codespace rebuild --full`



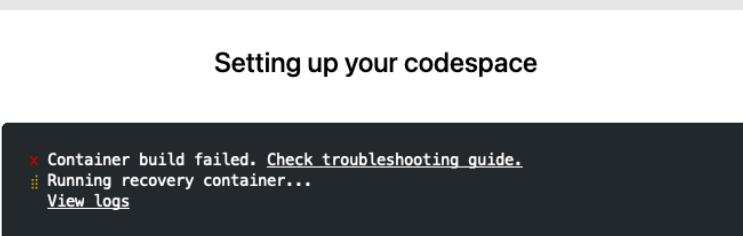
```
@gwstudent2 ~ /workspaces/codespaces-blank $ gh codespace rebuild
? Choose codespace: [Use arrows to move, type to filter]
> github/codespaces-blank (main): bug-free potato
```



Failed container builds

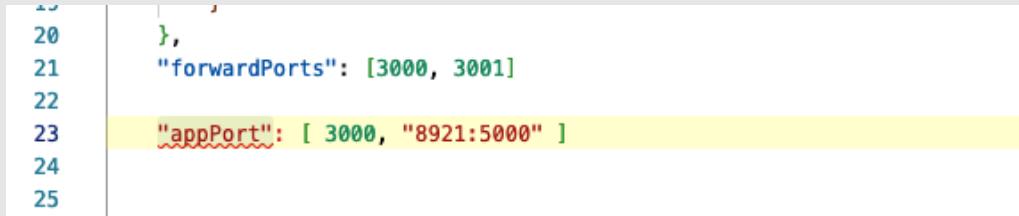
50

- If container build fails, first indication is in "setting up" window
- After startup, error dialog
- Then can choose to view Creation Log
- Errors also visible in PROBLEMS tab



Setting up your codespace

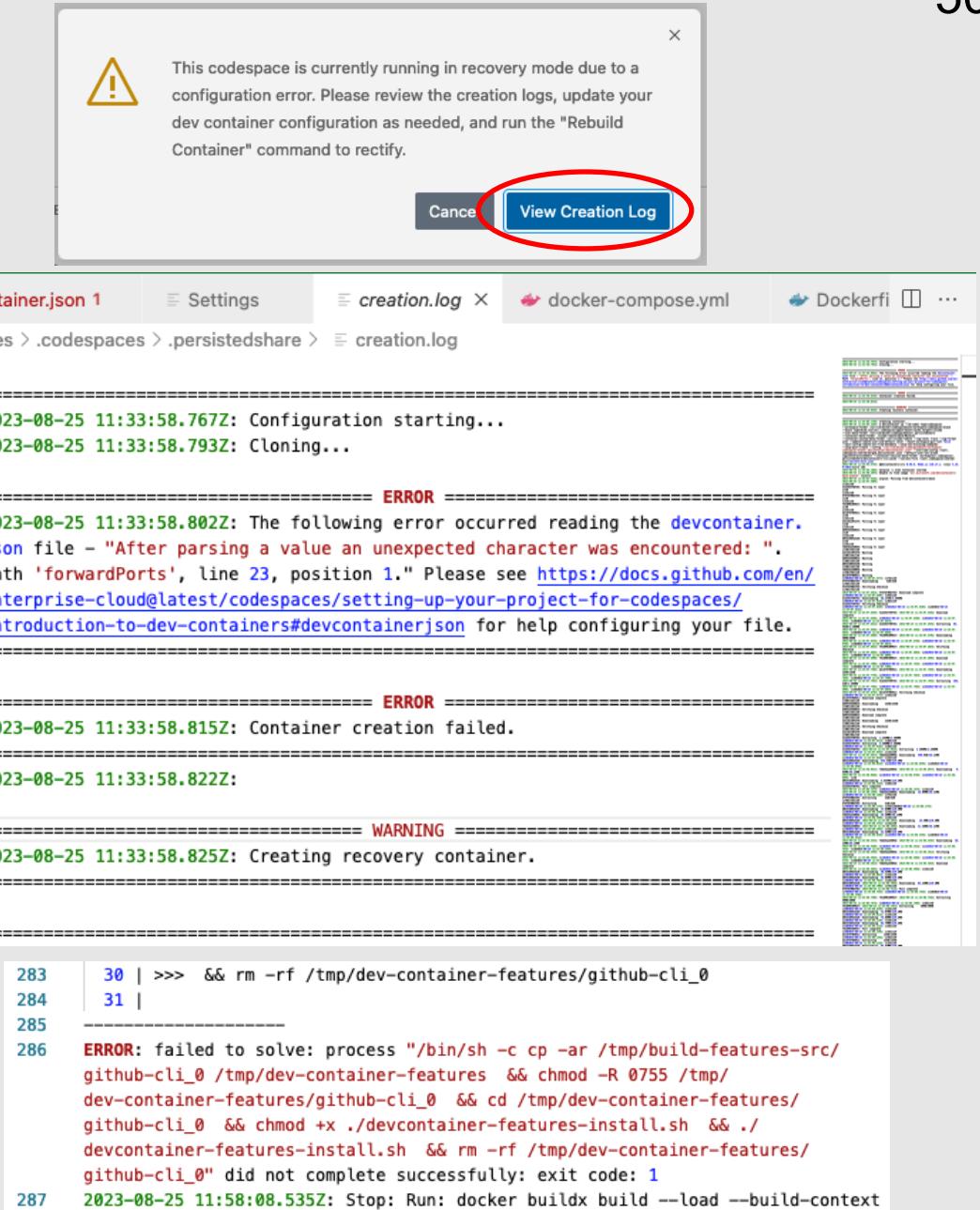
Container build failed. [Check troubleshooting guide.](#)
Running recovery container...
[View logs](#)



```
19
20
21
22
23 "appPort": [ 3000, "8921:5000" ]
24
25
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

devcontainer.json .devcontainer 1
Expected comma jsonc(514) [Ln 23, Col 2]
Expected comma



This codespace is currently running in recovery mode due to a configuration error. Please review the creation logs, update your dev container configuration as needed, and run the "Rebuild Container" command to rectify.

Canc View Creation Log

{ devcontainer.json 1 Settings creation.log × docker-compose.yml Dockerfi ...

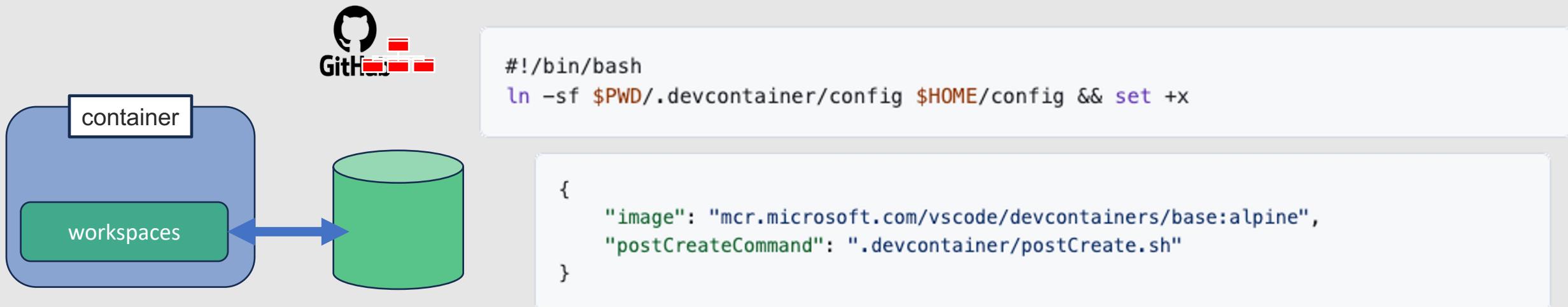
workspaces > .codespaces > .persistedshare > creation.log

```
1
2 =====
3 2023-08-25 11:33:58.767Z: Configuration starting...
4 2023-08-25 11:33:58.793Z: Cloning...
5
6 ====== ERROR =====
7 2023-08-25 11:33:58.802Z: The following error occurred reading the devcontainer.json file - "After parsing a value an unexpected character was encountered: ". Path 'forwardPorts', line 23, position 1." Please see https://docs.github.com/en/enterprise-cloud@latest/codespaces/setting-up-your-project-for-codespaces/introduction-to-dev-containers#devcontainerjson for help configuring your file.
8
9
10 ====== ERROR =====
11 2023-08-25 11:33:58.815Z: Container creation failed.
12
13 2023-08-25 11:33:58.822Z:
14
15 ====== WARNING =====
16 2023-08-25 11:33:58.825Z: Creating recovery container.
17
18
19
283 30 | >>> && rm -rf /tmp/dev-container-features/github-cli_0
284 31 |
285
286 286 ERROR: failed to solve: process "/bin/sh -c cp -ar /tmp/build-features-src/github-cli_0 /tmp/dev-container-features && chmod -R 0755 /tmp/dev-container-features/github-cli_0 && cd /tmp/dev-container-features/github-cli_0 && chmod +x ./devcontainer-features-install.sh && ./devcontainer-features-install.sh && rm -rf /tmp/dev-container-features/github-cli_0" did not complete successfully: exit code: 1
287 2023-08-25 11:58:08.853Z: Stop: Run: docker buildx build --load --build-context
```



Persisting data across rebuilds

- When codespace is created, /workspaces is mounted into container as persistent directory
- GitHub repo is cloned into /workspaces/
- Any changes made inside this directory are preserved on codespace starts, stops, rebuilds
- Includes any editing, adding, deleting of files in /workspaces
- Separate from /workspaces, codespace contains a Linux dir structure dependent on dev container used
- Other items preserved on start and stop
- To preserve files outside of /workspaces over a rebuild, create a directory and symlink into /workspaces
- Could even put shell script as postCreateCommand in devcontainer.json file



Credit: <https://docs.github.com/en/codespaces/developing-in-a-codespace/rebuilding-the-container-in-a-codespace#persisting-data-over-a-rebuild>



Seeing codespace templates

- When logged in, go to github.com/codespaces

The screenshot shows the GitHub Codespaces interface. At the top, there's a navigation bar with icons for back, forward, refresh, and home, followed by the URL <https://github.com/codespaces/>. Below the URL is a green header bar with the text "Getting Started" and "Actions - brentlaster...". The main content area is titled "Your codespaces". On the left, there's a sidebar with "All" selected and a "Templates" section. Under "Templates", there are two items: "gwstudent2/cppp1" and "gwstudent2/devcontainer-example". The main content area has a heading "Explore quick start templates" with three cards: "Blank" (By github), "React" (By github), and "Jupyter Notebook" (By github). Each card has a "Use this template" button. A red box highlights this "Explore quick start templates" section. Below this, there's a section titled "Owned by gwstudent2" with two items: "solid robot" and "legendary train". Each item has a small icon, a repository name, a branch name, a status message ("This codespace has uncommitted changes" or "Request Usage Report"), and some resource details like "2-core" and "4GB RAM". At the bottom of the page, there's a footer with links to GitHub's terms, privacy, security, status, docs, contact, pricing, API, training, blog, and about pages.



Using a template to create a codespace

The screenshot shows the GitHub Codespaces interface. On the left, the Explorer sidebar displays the project structure for 'CODESPACES-REACT [CODESPACES: FUZZY HA...]' containing files like .devcontainer, .vscode, node_modules, public, src (with App.css, App.js, App.test.js, index.css, index.js, logo.svg, reportWebVitals.js, setupTests.js), .gitignore, LICENSE, package-lock.json, package.json, and README.md. The main workspace shows the 'App.js' file open in a code editor:

```
src > JS App.js > ...
1 import './App.css';
2
3 function App() {
4   return (
5     <div className="App">
6       <header className="App-header">
7         
8         <p>
9           GitHub Codespaces <span className="heart"> ❤ </span>
10        </p>
11        <p className="small">
12          Edit <code>src/App.js</code> and save to <a href="#">Preview</a>
13        </p>
14        <a
15          className="App-link"
16          href="https://reactjs.org"
17          target="_blank"
18          rel="noopener noreferrer"
19        >
20          Learn React
21        </a>
22      </div>
23    )
```

The terminal output at the bottom shows:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS 1
Compiled successfully!
You can now view codespaces-react in the browser.
Local: http://localhost:3000
On Your Network: http://172.16.5.4:3000
Note that the development build is not optimized.
To create a production build, use npm run build.
webpack compiled successfully
```

A browser tab titled 'Simple Browser' shows the running application at <https://fuzzy-halibut-4q5w5p4wg5h5q94-3000.app.github.dev>. The page features a GitHub Octocat logo and the text "GitHub Codespaces ❤ React". A link "Learn React" is present. The bottom right corner of the browser window says "Edit src/App.js and save to reload."



Forwarding a port

PROBLEMS	OUTPUT	DEBUG CONSOLE	TERMINAL	PORTS 6	COMMENTS	
Port	Forwarded Address	Running Process			Visibility	Origin
● Application (5000)	https://studious-u...	/usr/local/python/3.10.13/bin/python /home/codespace/.pytho...			Private	User Forwarded
○ 5992	https://studious-umbrella-jg...				Private	Auto Forwarded
○ 34319	https://studious-umbrella-jg...				Private	Auto Forwarded
○ 34521	https://studious-umbrella-jg...				Private	Auto Forwarded
○ 41243	https://studious-umbrella-jg...				Private	Auto Forwarded
● 43985	https://studious-umbrella-jg...	Code Extension Host (514)			Private	Auto Forwarded
○ 44101	https://studious-umbrella-jg...				Private	Auto Forwarded
Add Port						

- PORTS tab
- Add Port button
- If known port will be added with relevant data

Port	Forwarded Address	Running Process	Visibility	Origin
● Application (5000)	https://studious-u...	/usr/local/python/3.10.13/bin/python /home/codespace/.pytho...	Private	User Forwarded
○ 5992	Port labeled Application. Remote port localhost:5000 forwarded to local address https://studious-umbrella-jgv9j4jjx45cq7r4-5000.app.github.dev/.			Private
○ 34319	https://studious-umbrella-in...		Private	Auto Forwarded
○ 34521	https://studious-umbrella-in...		Private	Auto Forwarded



Working with forwarded ports

The screenshot shows the VS Code interface with the "PORTS" tab selected, displaying a list of forwarded ports. A context menu is open over the first port entry.

Port	Forwarded Address	Running Process	Visibility
Application (5000)	https://studious-u...	/usr/local/python/3.10.13/bin/python /home/codespace/.pytho...	Private

Annotations highlight specific actions:

- Change port label**: Points to the "Set Port Label" option in the context menu.
- Stop forwarding port**: Points to the "Stop Forwarding Port" option in the context menu.
- Copy local addr**: Points to the "Copy Local Address" option in the context menu.
- Preview in editor**: Points to the "Preview in Editor" button in the main interface.
- Open in browser**: Points to the "Open in Browser" button in the main interface.

The bottom right corner shows a preview window displaying a GitHub Codespaces Flask demo page.

Context Menu Options (Open in Browser view):

- Open in Browser
- Preview in Editor
- Set Port Label
- Set Label and Update devcontainer.json
- Copy Local Address ⌘C
- Port Visibility Private
- Change Port Protocol HTTP
- HTTPS
- Stop Forwarding Port ⌘Backspace
- Forward a Port



Exporting codespaces to a branch

- If you created the codespace from a repository where you have write access, you can export changes to a new branch in the repo
- The name of the new branch will be the permanent name of your codespace prefixed by the string codespace-, for example [codespace-zany-space-eureka-p74p45vww63rq4q](#)

The screenshot shows a GitHub repository named "cspaces" which is public. The repository has 2 branches and 0 tags. A message indicates that the "zany space eureka" branch is 10 commits behind the "main" branch. The "Codespaces" tab is selected, showing a list of workspaces in the cloud. A context menu is open for the "zany space eureka" codespace, with the "Export changes to a branch" option highlighted.

cspaces Public

2 branches 0 tags

This branch is 10 commits behind main.

brentlaster Update devcontainer.json

.devcontainer Update devcontainer.json

extra Create registry-compose.yml

images Add files via upload

Edit Pins Unwatch 3

Go to file Add file Code

Local Codespaces

Codespaces Your workspaces in the cloud

On current branch

zany space eureka 1w ...

main No changes

Open in ... →

Rename

Export changes to a branch

Change machine type

Keep codespace

Delete



Codespaces publish to new repo

- If code is based **on a template** (not associated with existing repo...)
- Click on source control icon in Activity Bar
- Click on "Publish to GitHub" button
- Brings up command pallet with action
- Follow prompts

The screenshot illustrates the GitHub Codespaces publishing process. At the top, a browser window shows the GitHub Dev Container configuration for a repository named 'codespaces-demo1'. A red circle highlights the 'Publish to GitHub' button in the activity bar. Below it, another red circle highlights the 'Publish to GitHub public repository' option in the dropdown menu. In the middle, a modal dialog titled 'Select which files should be included in the repository.' shows a list of files, with a checkbox next to '.devcontainer' being selected. At the bottom, the GitHub repository page for 'codespaces-demo1' is shown, displaying the commit history for 'first commit' and a success message: 'Successfully published the "gwstudent2/codespaces-demo1" repository to GitHub.'

Lab 4 - Templates and Ports

Purpose: In this lab, we'll see how to create a codespace from a template and also work with ports.



Personalizing GitHub Codespaces

- Settings Sync - Synchronize your VS Code settings between the desktop app and VS Code web client
- Dotfiles - use a *dotfiles* repo to specify scripts, shell preferences, and other configurations
- Settings->Codespaces
- Defaults config that applies to every codespace for a repo

The screenshot shows the GitHub Codespaces settings interface. The left sidebar lists various account and repository management options, with 'Codespaces' selected. The main content area is divided into several sections: 'Dotfiles' (with an option to automatically install dotfiles), 'Codespaces secrets' (which is currently empty), 'GPG verification' (with an enabled checkbox for GPG signing), and 'Settings Sync' (with an enabled checkbox for VS Code Settings Sync). The top right corner of the browser window shows a 'Paused' status.



Settings sync

- Allows you to sync "look and feel" across interfaces
- Can select Manage (bottom icon) and sign in

The screenshot shows the VS Code interface with a floating 'Settings Sync' dialog box. The dialog box contains a list of items to sync (Settings, Keyboard Shortcuts, User Snippets, User Tasks, UI State, Extensions, Profiles) and three bullet points: 'Personalize your VS Code', 'Choose the look you want', and 'One shortcut to access everything'. At the bottom of the dialog is a 'Backup and Sync Settings' button. The main VS Code window shows a terminal tab with some command-line output.

VS Code Settings Sync is requesting additional permissions

VS Code Settings Sync

By accepting, your codespace will be able to push and pull configurations from VS Code Settings Sync.

Please make sure you trust this repository before continuing.

See the [Codespaces and VS Code Settings Sync](#) documentation for more information.

Authorize



Dotfiles

- Files and folders starting with . that control config
- Can store dotfiles in GitHub repo
- Point codespaces to dotfiles repo
 - <https://github.com/settings/codespaces>
- Codespace will get dotfile and clone them into repo
- Dotfiles repo looks for typical startup file to setup the env (**install.sh, install, bootstrap.sh, script/bootstrap, setup.sh setup, script/setup**)
- If found, runs one of those
- If not found, will symlink any files starting with ":"
- NOTE: Must be "dotfiles" repo for current logged in user

gwstudent2 / dotfiles

Code Pull requests Actions Projects Wiki Security Insights

dotfiles Public

forked from skillrepos/dotfiles

Pin Watch 0

gwstudent (gwstudent) Your personal account Go to your personal profile

Public profile Account Appearance Accessibility Notifications

Dotfiles

Automatically install dotfiles
Codespaces can automatically install your dotfiles into every codespace you create. [Learn how to set up your dotfiles for Codespaces.](#)

Select repository ▾

gwstudent2 Rename .bashrc_aliases to .bash_al...	7df0f5b 4 minutes ago	7 commits
.bash_aliases	Rename .bashrc_aliases to .bash_aliases	4 minutes ago
.zshrc	Create .zshrc	36 minutes ago
LICENSE	Initial commit	1 hour ago
README.md	Initial commit	1 hour ago

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

● @gwstudent2 → /workspaces/greetings (master) \$ ls -la ~/.zshrc
lrwxrwxrwx 1 codespace codespace 55 Dec 3 18:24 .zshrc → /workspaces/.codespaces/.persistedshare/dotfiles/.zshrc
○ @gwstudent2 → /workspaces/greetings (master) \$ █



Codespace Secrets

- Can create secrets specifically for codespaces
- Define which repos have access to the secrets (org, repo, etc.)
- Reference secrets as environment variables
- Secret values may be referenced in codespaces as env variables

```

TERMINAL ... bash + v ⚡ ... ^ X

👉 Welcome to Codespaces! You are on a custom image defined in your devcontainer.json file.

🔍 To explore VS Code to its fullest, search using the Command Palette (Cmd/Ctrl + Shift + P)

📝 Edit away, then run your build command to see your code running in the browser.

@gwstudent2 ~ /workspaces/codespaces-demo1 (main) $ echo $MY_SECRET_INFO
value: 123
@gwstudent2 ~ /workspaces/codespaces-demo1 (main) $

```

github.com/settings/codespaces

Secret updated.

gwstudent2 (gwstudent2)
Your personal account

Dotfiles

Automatically install dotfiles
Codespaces can automatically install your dotfiles into every codespace you create. [Learn how to set up your dotfiles for Codespaces.](#)

Codespaces secrets [New secret](#)

Development environment secrets are environment variables that are **encrypted**. They are available to any codespace you create using repositories with access to that secret.

MY_SECRET_INFO	Updated now
Available to 2 repositories.	
Update	Delete

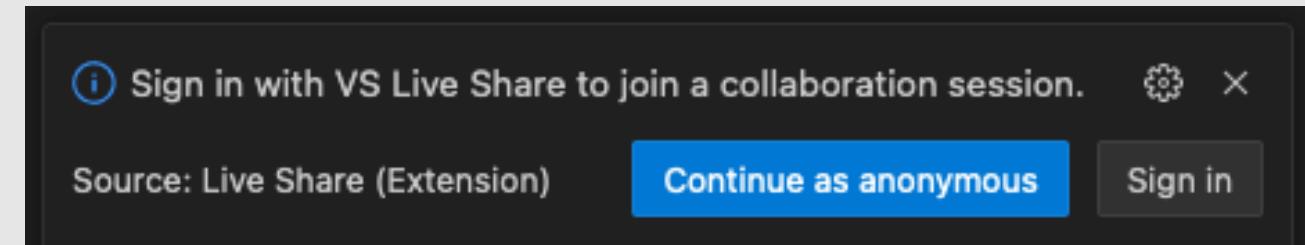
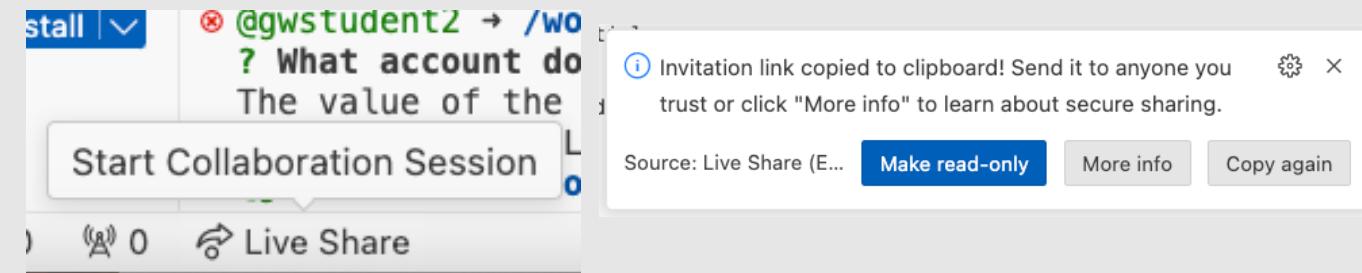
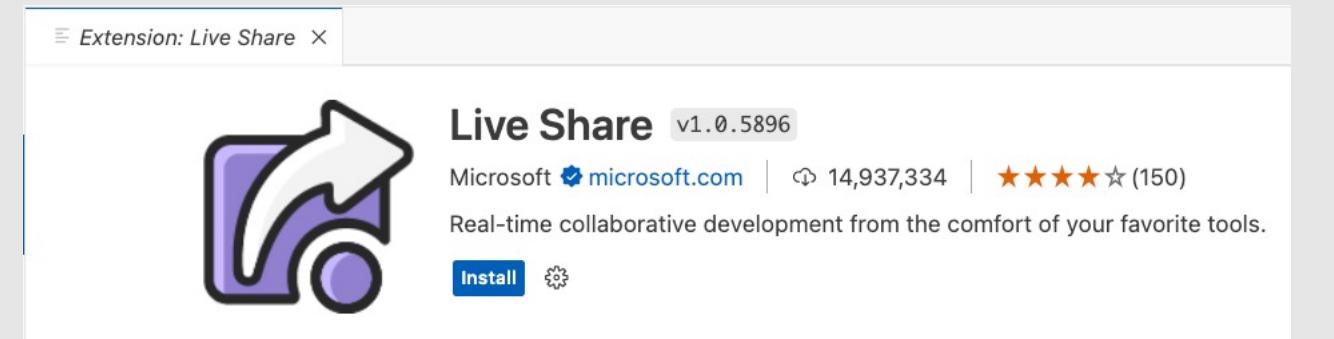
GPG verification

Codespaces can have GPG commit signing capabilities so that GitHub can verify that commits made in the codespace come from a trusted source. When enabled, this setting will be applied to your list of trusted repositories.



Live sharing

- Done via Live Share extension
- Allows for quick collaboration w/o needing to sync code or configure same dev environment
- Shared context of workspace in each editor
- Actions and changes from either participant are instantly reflected
- Can debug as well



Lab 5 - Live sharing and collaboration

Purpose: In this lab, we'll see how to share a codespace and collaborate with it.



Codespace Prebuilds

- Pre-build functionality

- automatically pre-assembles pieces needed for codespaces for repo+branch+devcontainer.json
- push to branch w/ prebuild runs GH Action to rebuild image
- uses space to store prebuilt image that will be billable or come out of existing bucket

The screenshot shows the GitHub Codespaces interface. The top navigation bar includes links for Code, Pull requests, Actions, Projects, Wiki, Security, and Insights. On the left, a sidebar menu lists General, Access, Collaborators, Moderation options, Code and automation, Branches, Tags, Rules, Actions, Webhooks, Environments, and Codespaces. The Codespaces section is currently selected. The main content area displays the 'Prebuild configuration' section, which contains a message stating 'There are no prebuilds config'. It explains that prebuild configurations speed up Codespaces by executing all the tasks required to build. A 'Learn more about setting up prebuilds' link and a 'Set up prebuilds' button are present. At the bottom right of the content area is a 'Create' button.

The screenshot shows the 'Codespaces / New prebuild configuration' dialog. It includes a warning message about storage consumption and billable charges. The 'Configuration' section allows selecting a branch. The 'Access and cost control' section includes 'Prebuild triggers' (set to 'Every push'), 'Region availability' (set to 'US East'), and a checkbox for 'Reduce prebuild availability to only specific regions'. The 'Template history' section shows 1 version retained. The 'Failure notifications' section has a search field for users to receive email notifications. The 'You haven't added anyone yet' section provides instructions to add members for notifications. A 'Show advanced options' link is at the bottom right.



Checking expiration date

- If close to expiring, can see at github.com/codespaces
- Ones marked as keep have bookmark icon
- If marked as keep, there is an "unkeep" option

Owned by brentlaster

Created from [github/codespaces-flask](#)

silver space meme

4-core • 16GB RAM • 32GB • 0.9 GB • Last used 5 days ago

skillrepos/cspaces

zany space eureka

main No changes

2-core • 8GB RAM • 32GB • 0.82 GB • Last used 6 days ago

brentlaster/copilot-dd

literate memory

main No changes

2-core • 8GB RAM •

- Open in ... →
- Rename
- Export changes to a branch
- Change machine type
- Unkeep codespace**
- Delete



Managing codespaces with gh CLI

- GitHub CLI has "codespace" command
- commands such as *create, list, stop*, etc.
- Can select options via arrow keys if multiple (easier than full command line options)

```
developer@Bs-MacBook-Pro calc3 % gh codespace
Connect to and manage codespaces

USAGE
  gh codespace [flags]

AVAILABLE COMMANDS
  code:      Open a codespace in Visual Studio
  cp:        Copy files between local and rem...
  create:    Create a codespace
  delete:   Delete codespaces
  edit:     Edit a codespace
  jupyter:  Open a codespace in JupyterLab
  list:     List codespaces
  logs:    Access codespace logs
  ports:   List ports in a codespace
  rebuild: Rebuild a codespace
  ssh:     SSH into a codespace
  stop:    Stop a running codespace
  view:   View details about a codespace
```

```
developer@Bs-MacBook-Pro calc3 % gh codespace list
NAME          DISPLAY NAME      REPOSITORY      BRANCH      STATE      CREATED AT
opulent-tribb...  opulent tr...  gwstudent2...  main*       Shutdown  about 3 da...
ideal-goldfis...  ideal gold...  skillrepos...  main*       Available about 23 h...
```

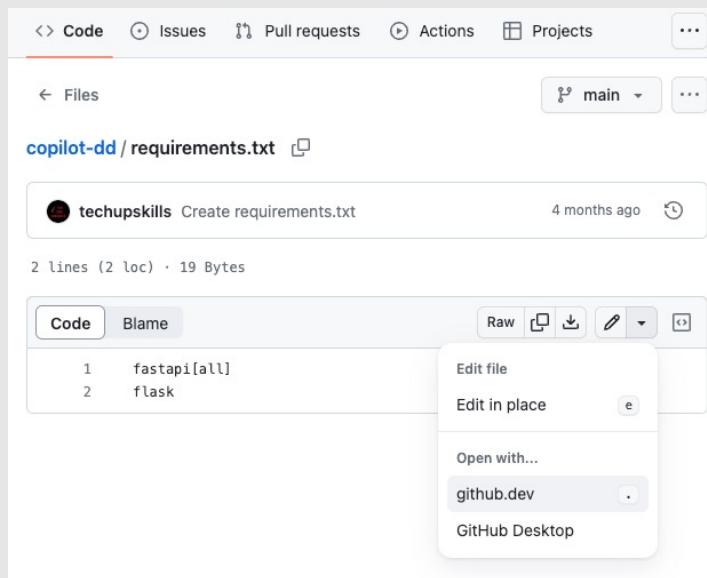
```
developer@Bs-MacBook-Pro calc3 % gh codespace rebuild
? Choose codespace: [Use arrows to move, type to filter]
> skillrepos/cspaces (main*): ideal goldfish
gwstudent2/caz-class-v2 (main*): opulent tribble
```



GitHub.dev editor vs Codespaces

- GitHub.dev editor

- lightweight editor that runs entirely in your browser
- navigate files and repos
- make and commit code changes
- open any repo, fork, or PR in editor
- Includes search syntax highlighting, and source control view



	github.dev	GitHub Codespaces
Cost	Free.	Free monthly quota of usage for personal accounts. For information on pricing, see " About billing for GitHub
Available		
Start	Instantly	Instantly
Comes with	IDE, Git, and build tools	IDE, Git, and build tools
Terminal access	Yes	Yes
Extensions	Yes	Yes



That's all - thanks!

69

Contact: training@getskillsnow.com

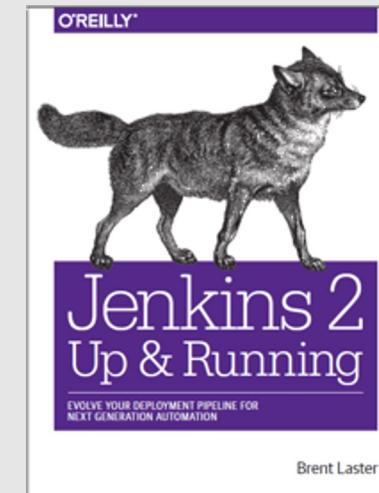
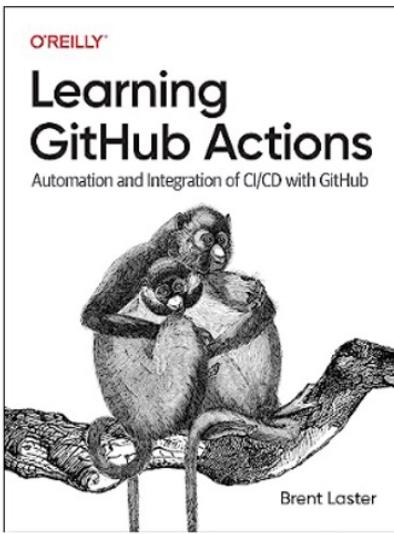
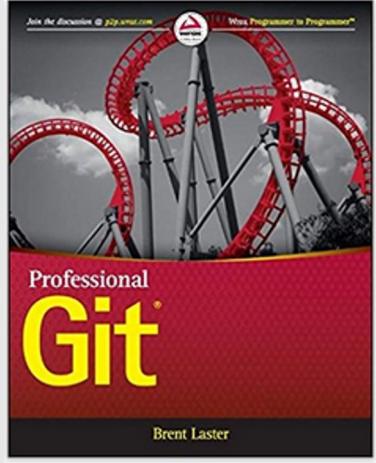
techskilltransformations.com
getskillsnow.com

Professional Git 1st Edition

by Brent Laster (Author)

★★★★★ 7 customer reviews

[Look inside](#) ↴



The screenshot shows the homepage of techskilltransformations.com. The header includes the company name "TECH SKILLS TRANSFORMATIONS LLC" and a navigation menu with links to HOME, ABOUT US, CORPORATE TRAINING, CUSTOM TRAINING, TESTIMONIALS, PUBLICATIONS, and MORE. A central banner features the text "THE CONTINUOUS UPSKILLING COMPANY!" and a call to action: "Get the instruction you need to upskill now! [Contact us](#) to see how we can help you understand & use the tech you need." Below the banner, there is a section titled "UPCOMING LIVE TRAINING WITH O'REILLY MEDIA!" and a "TRAINING TOPICS" section. A small circular icon with a speech bubble is visible in the bottom right corner.