



Middlebury

CSCI 201: Data Structures

Fall 2024

Lecture 12T: Software Development I (Repositories)

Goals for today:

Motivation: is there a good way to save different versions of our codes?



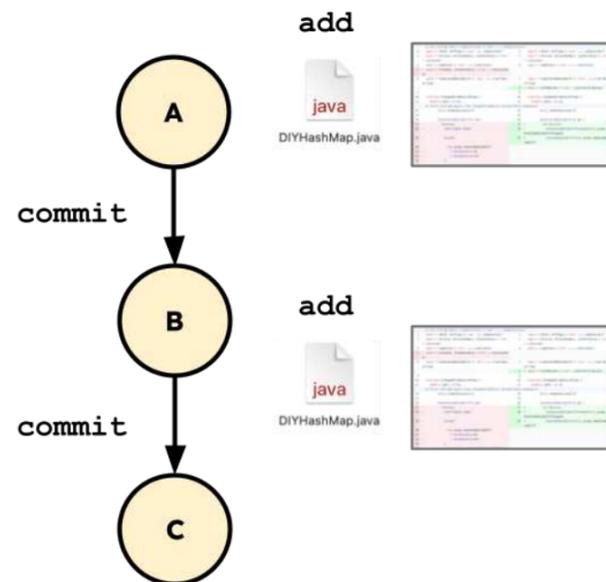
- Use **git** to keep track of different version of your codes.
- Use **GitHub** as a central (**remote**) location for your code repository.
- Save code changes using the **add (stage)** and **commit** steps.
- **Synchronize** commits to/from the remote repository using **push** and **pull**.
- Practice indexing cells in a **grid** (useful for Homework 10).

Useful for electives, personal projects, and in industry.

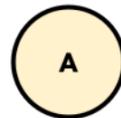
git and GitHub in a nutshell (**git** ≠ **GitHub**).

- **git** is a *version control system*.
- **git** stores your code as a series of **snapshots** represented as a **graph**.
- Making a **commit** takes a snapshot of your code (creating a new node in the graph).
- **GitHub** is a *repository hosting* service: a place to store a central graph.
- When you **clone** a repository, you're copying the graph.
- Your local repository and the remote repository may not always be in sync.

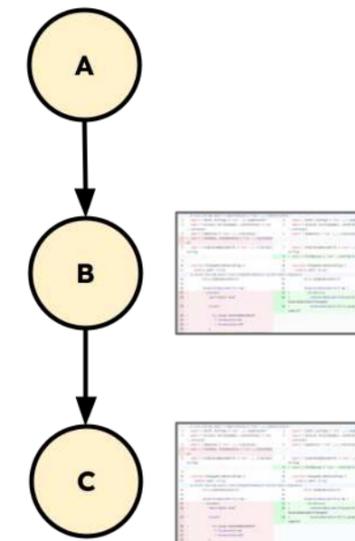
On your computer (or other development machine):



On **GitHub**:

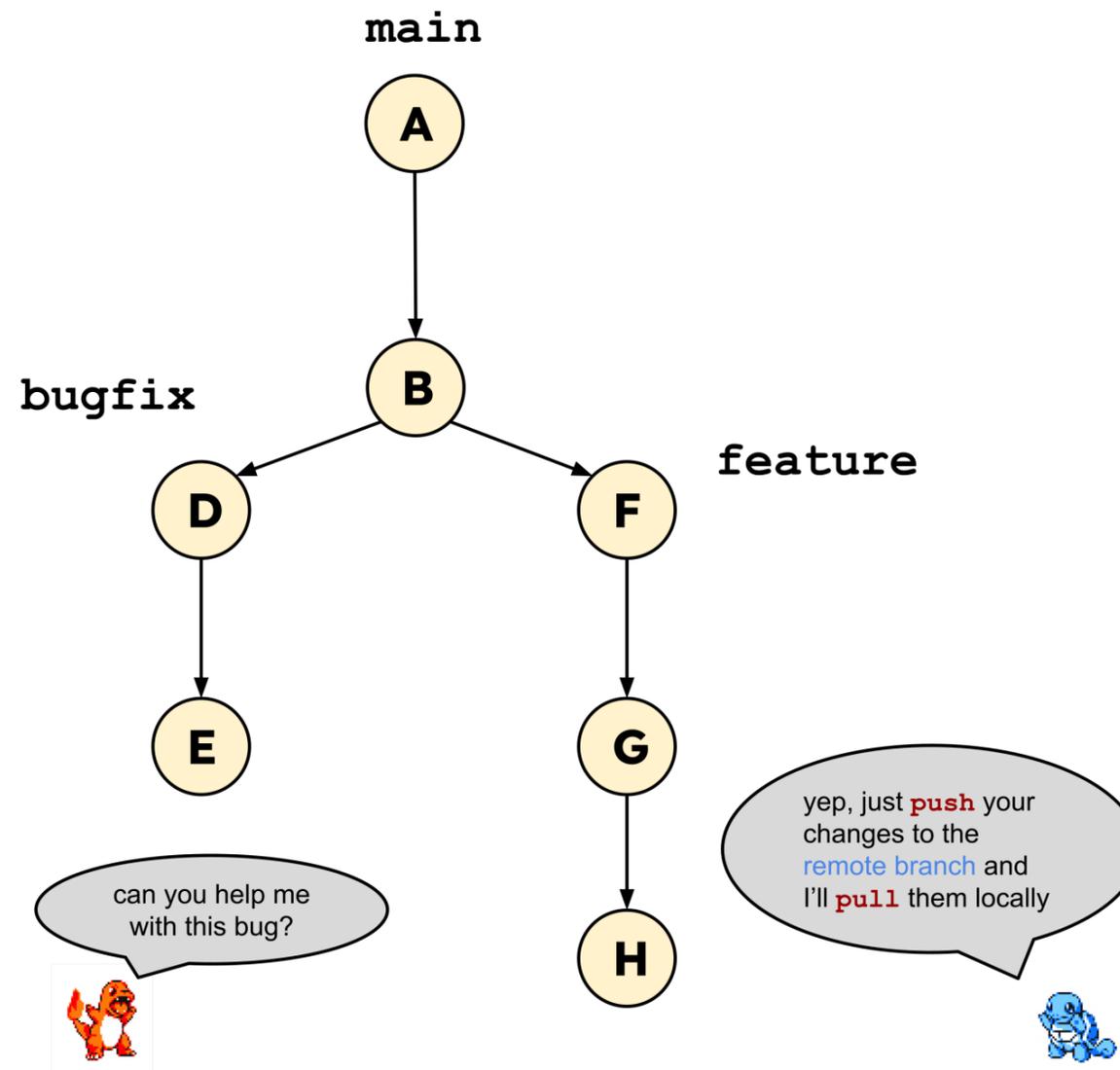


On **GitHub** (after **push**):



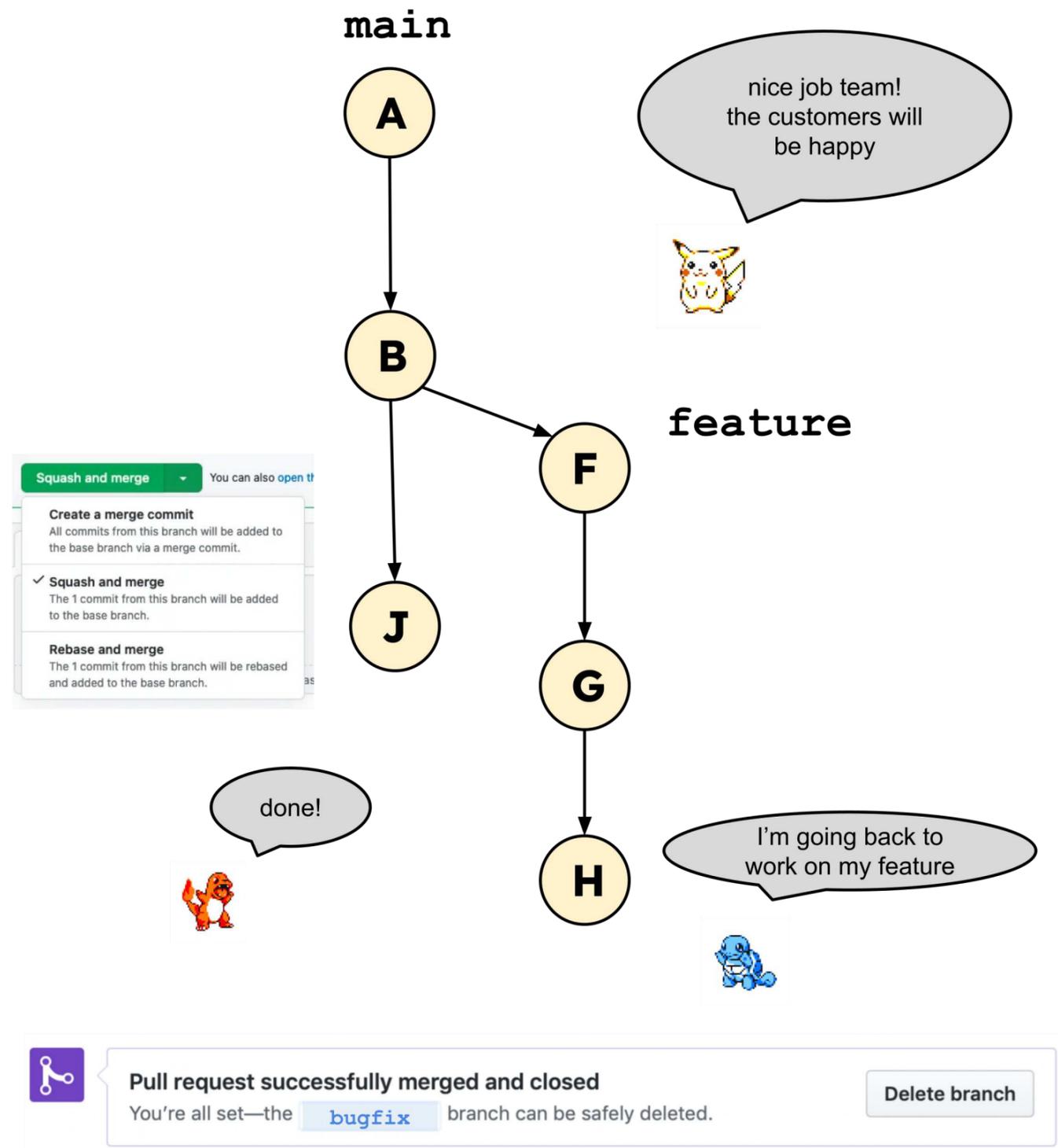
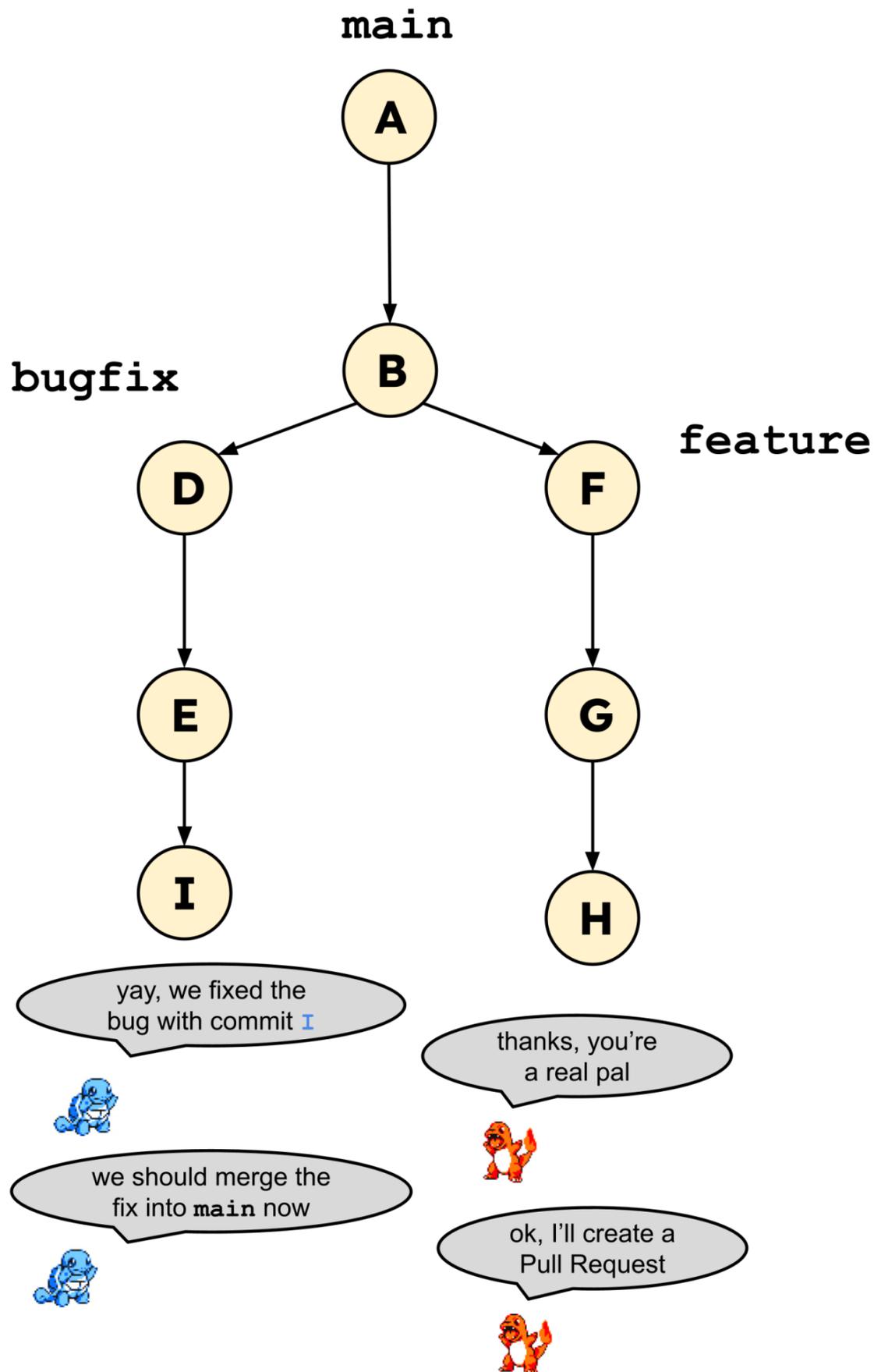
Main concepts: **add**, **commit**, **push**

Usually, you'll work in a *branch*. If you're working alone, you can just work in a single (**main**) branch.



push your commits to the remote repository
pull commits from the remote repository

Working on different branches in a team.

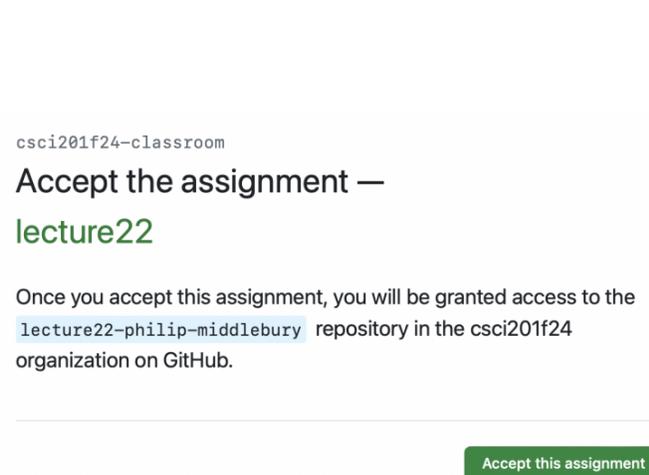


Now, we'll practice with `git` and **GitHub**.

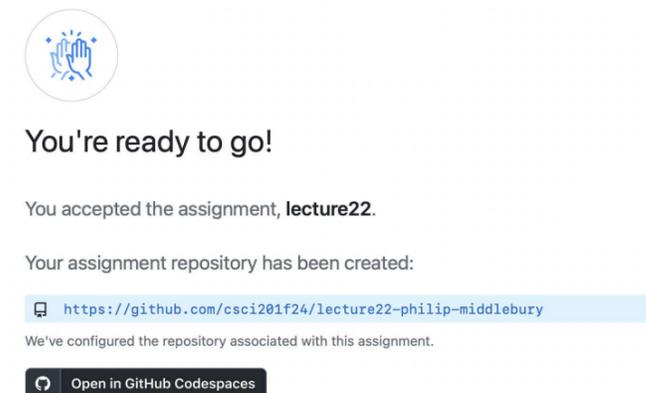
1. If you do not already have an account on **GitHub**, create one here: <https://github.com/signup>.
2. Then click on the `code` link on the course webpage: <https://classroom.github.com/a/bBsdeyav>.



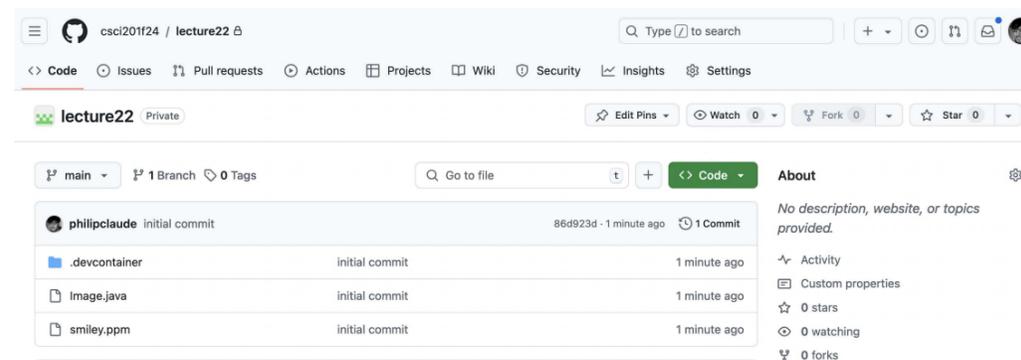
Find your name and join the classroom.



Accept the assignment.



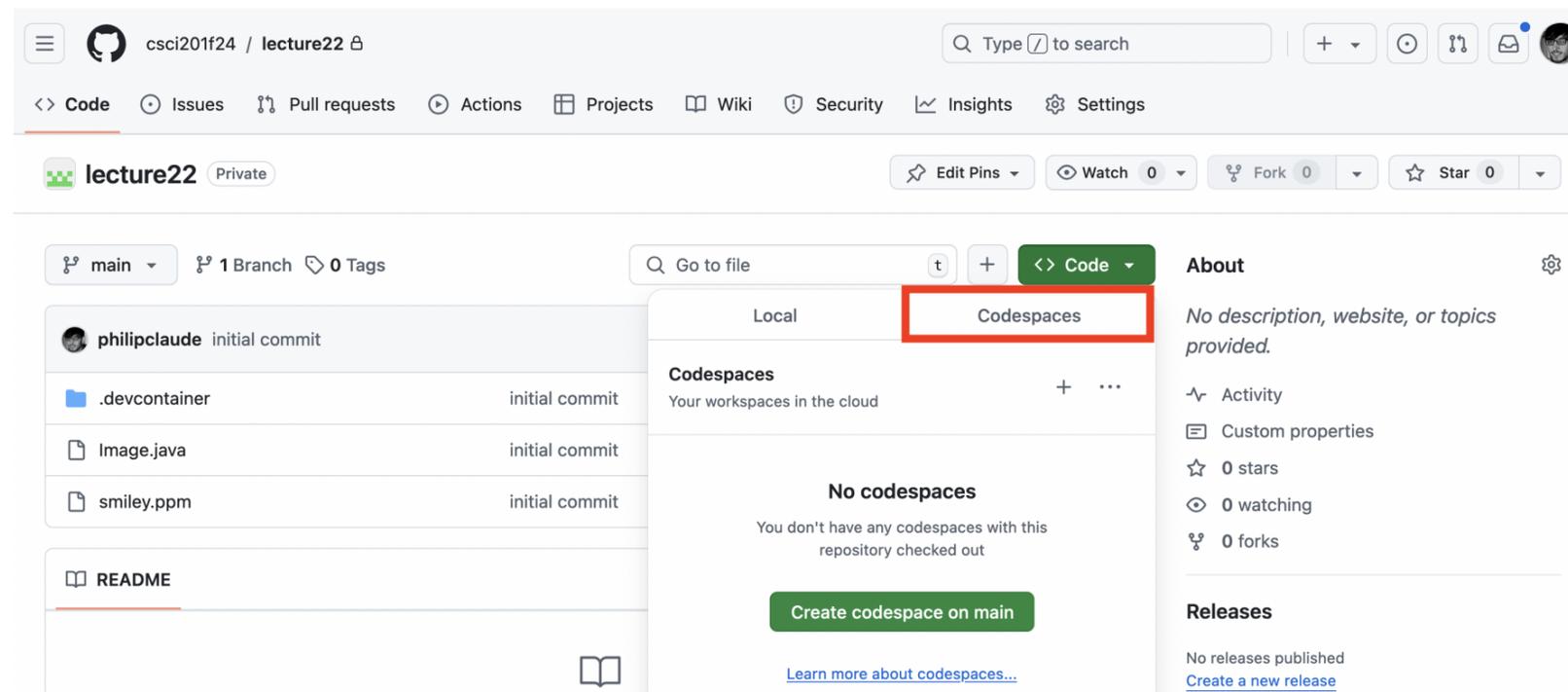
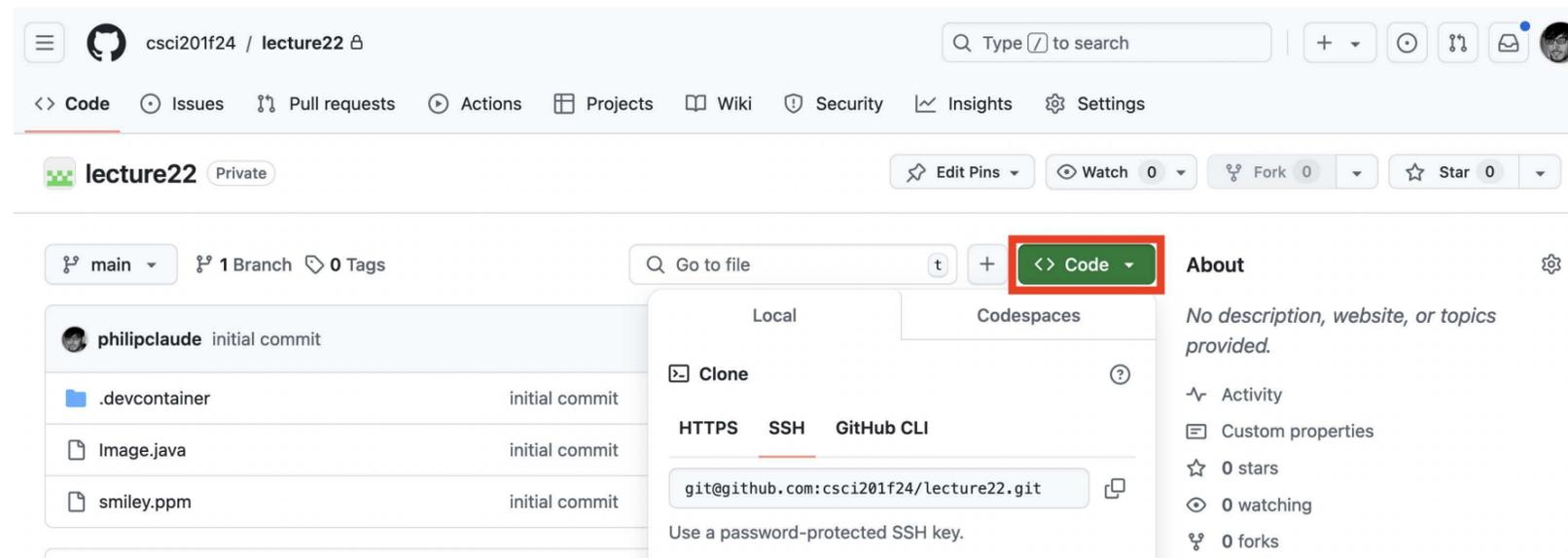
Click on the link to open the repository.



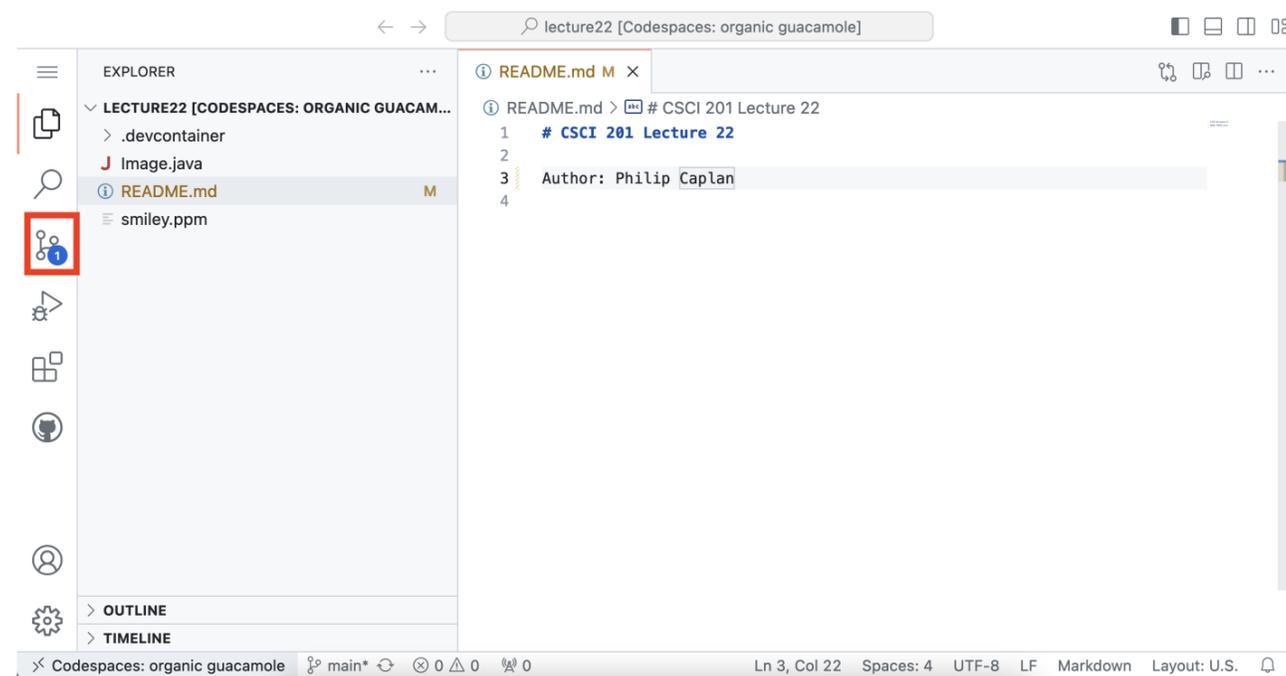
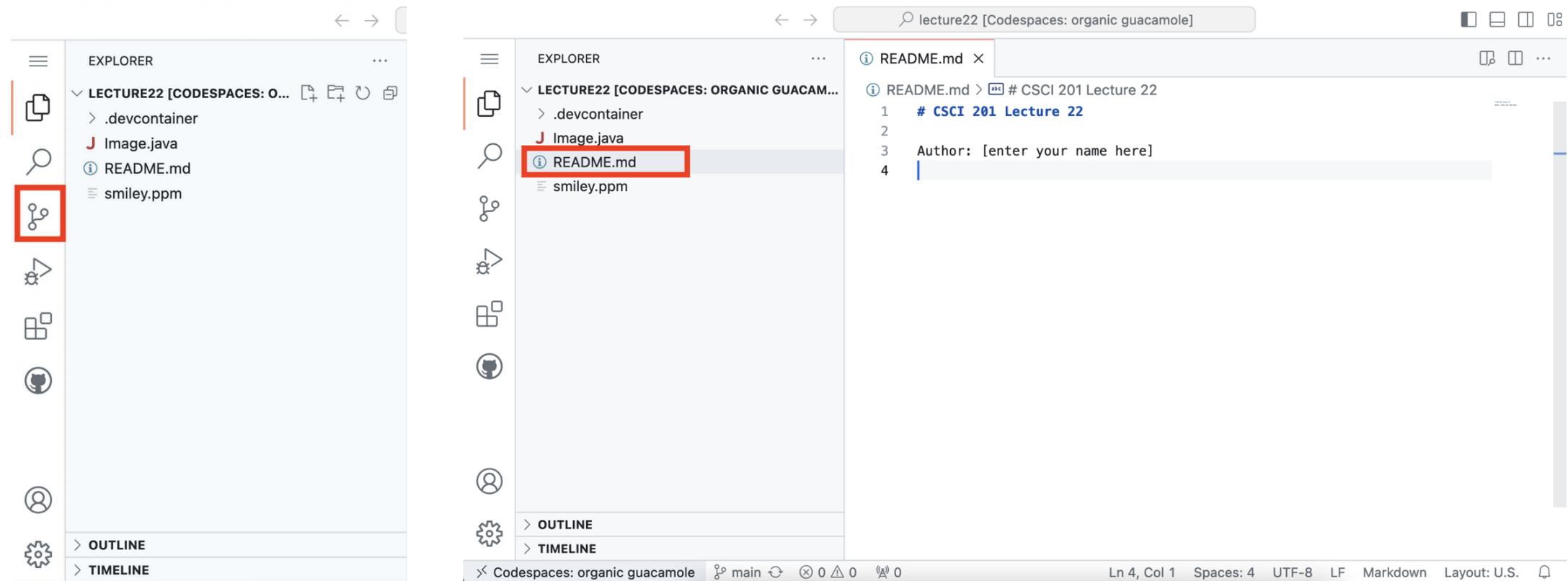
You should then see this.

Our "local" computer will be a virtual machine in the cloud.

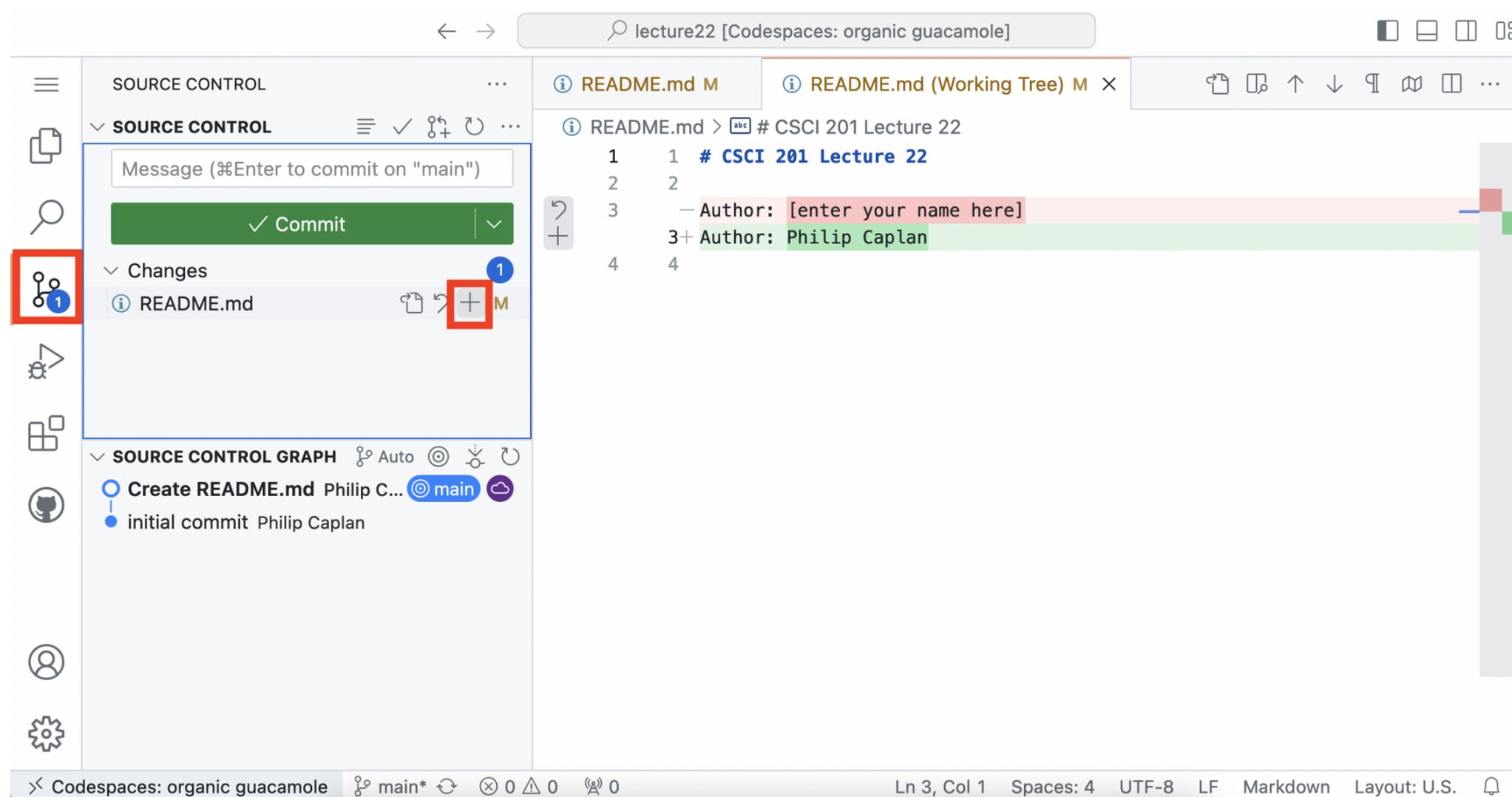
We'll work on our code in a **GitHub Codespace** (which will have VS Code integrated).



The *Source Control* button can be used to do **git** commands.



Inspect the changes by clicking on a file in the "Changes" section.



Then click on the + button to "add" (or "stage") your changes.

Add a message (information about the changes) and then click the "Commit" button.

The screenshot shows a code editor interface for a repository named "lecture22 [Codespaces: organic guacamole]". The left sidebar is divided into two main sections: "SOURCE CONTROL" and "SOURCE CONTROL GRAPH".

In the "SOURCE CONTROL" section, a text input field contains the message "added my name to the readme". Below this field is a green button with a checkmark and the text "Commit". This entire area is highlighted with a red rectangular box. Below the commit button, there are sections for "Staged Changes" (showing 1 change for README.md) and "Changes" (showing 0 changes).

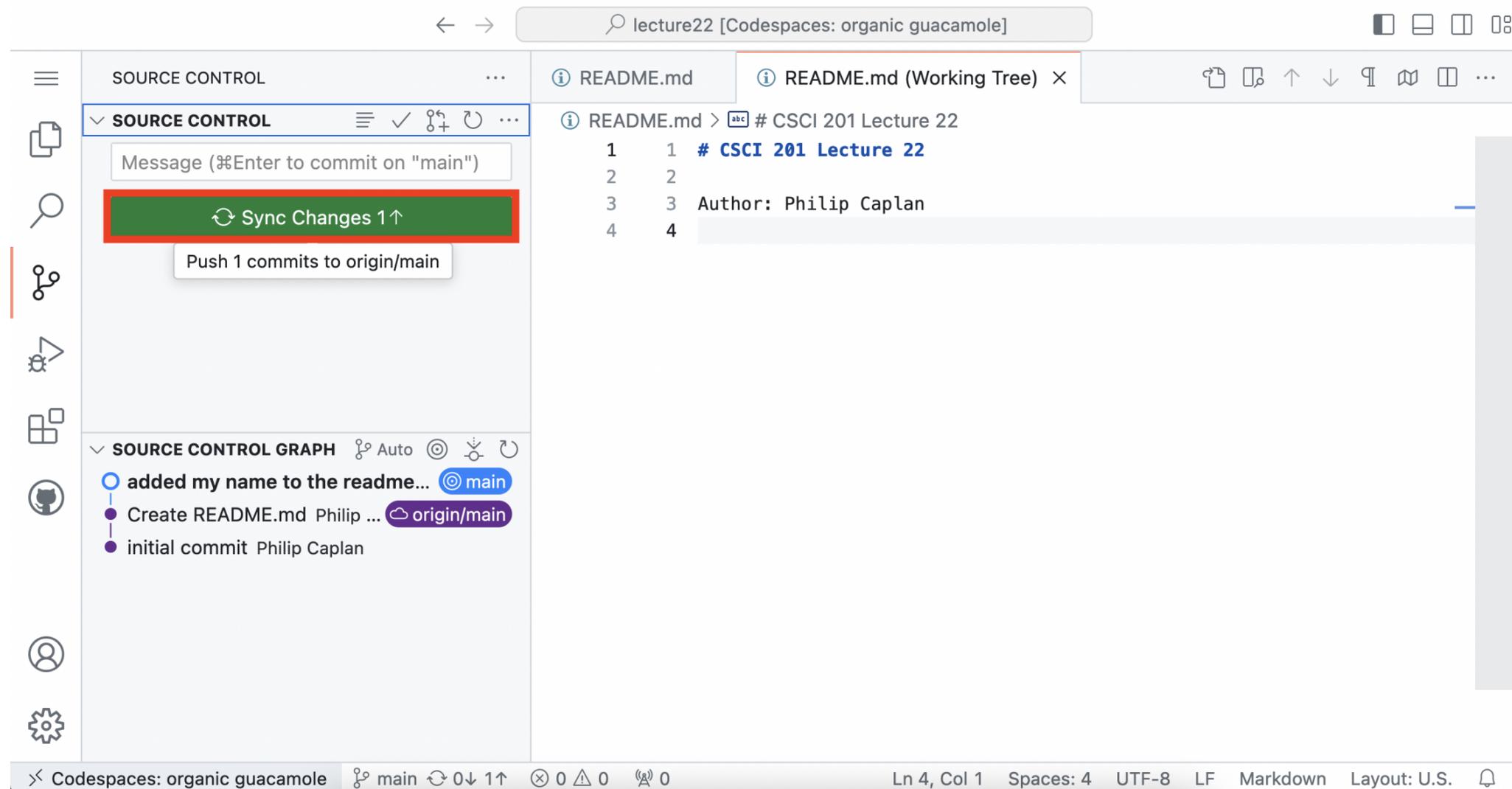
The "SOURCE CONTROL GRAPH" section shows a commit history with two entries: "Create README.md Philip C..." on the "main" branch and "initial commit Philip Caplan".

The main editor area displays the content of the selected "README.md" file in the "Working Tree". The content is as follows:

```
1 1 # CSCI 201 Lecture 22
2 2
3 3 Author: Philip Caplan
4 4
```

The status bar at the bottom of the editor shows the current file path "Codespaces: organic guacamole", branch "main+", and various status icons. On the right side of the status bar, it displays "Ln 4, Col 1", "Spaces: 4", "UTF-8", "LF", "Markdown", and "Layout: U.S.".

Finally, **push** the changes to the remote repository (on **GitHub**) by clicking on "Sync Changes".



Now the **GitHub** repository has the commit we just made.

We can also click on the commit message to get more info about the changes.

The screenshot shows the GitHub repository page for 'csci201f24 / lecture22'. The commit history table lists the following commits:

Commit Message	Author	Time
added my name to the readme	philipclaude	3 minutes ago
initial commit		13 minutes ago
initial commit		13 minutes ago
initial commit		13 minutes ago

The README content is:

```
CSCI 201 Lecture 22
Author: Philip Caplan
```

The screenshot shows the GitHub commit page for 'Commit c372327'. The commit message is 'added my name to the readme'. The diff view shows the change in README.md:

```
@@ -1,3 +1,3 @@
1 # CSCI 201 Lecture 22
2
3 - Author: [enter your name here]
3 + Author: Philip Caplan
```

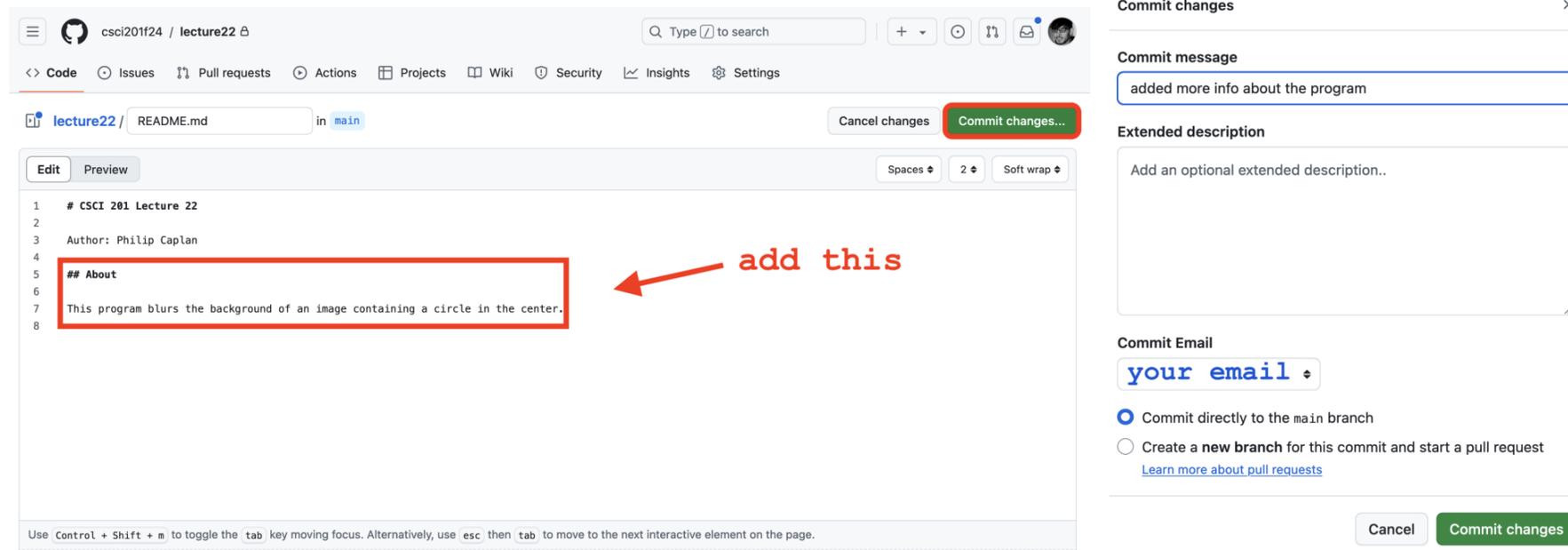
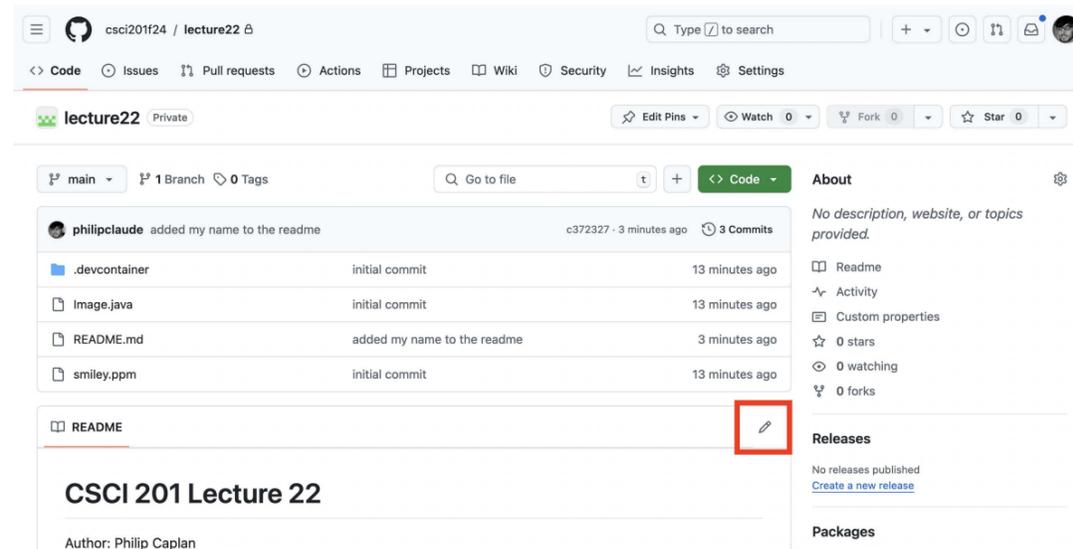
Our problem for today: a bit of image processing (to practice indexing grid cells).



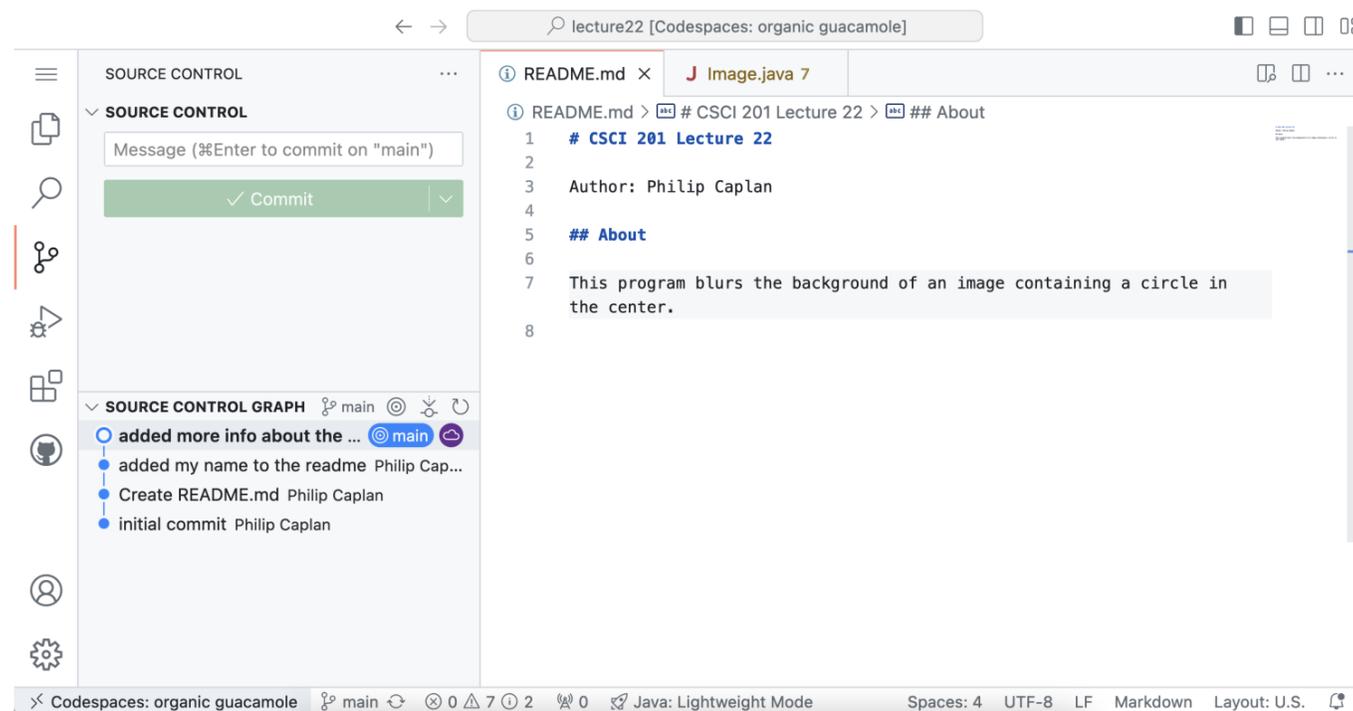
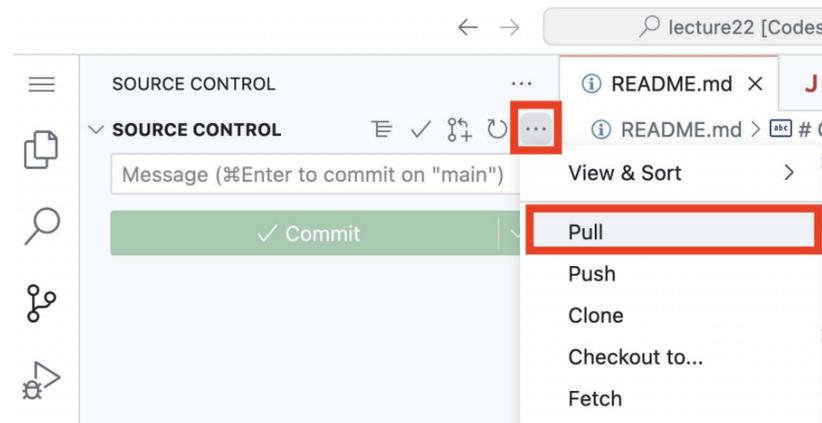
- Smiley wants to blur the background of each frame (in a video chat) so it doesn't look like a ski hill.
- We'll help Smiley by writing a program to blur the background pixels.

But first, let's practice with **committing** changes and **pulling**.

We can also do this directly in **GitHub**!

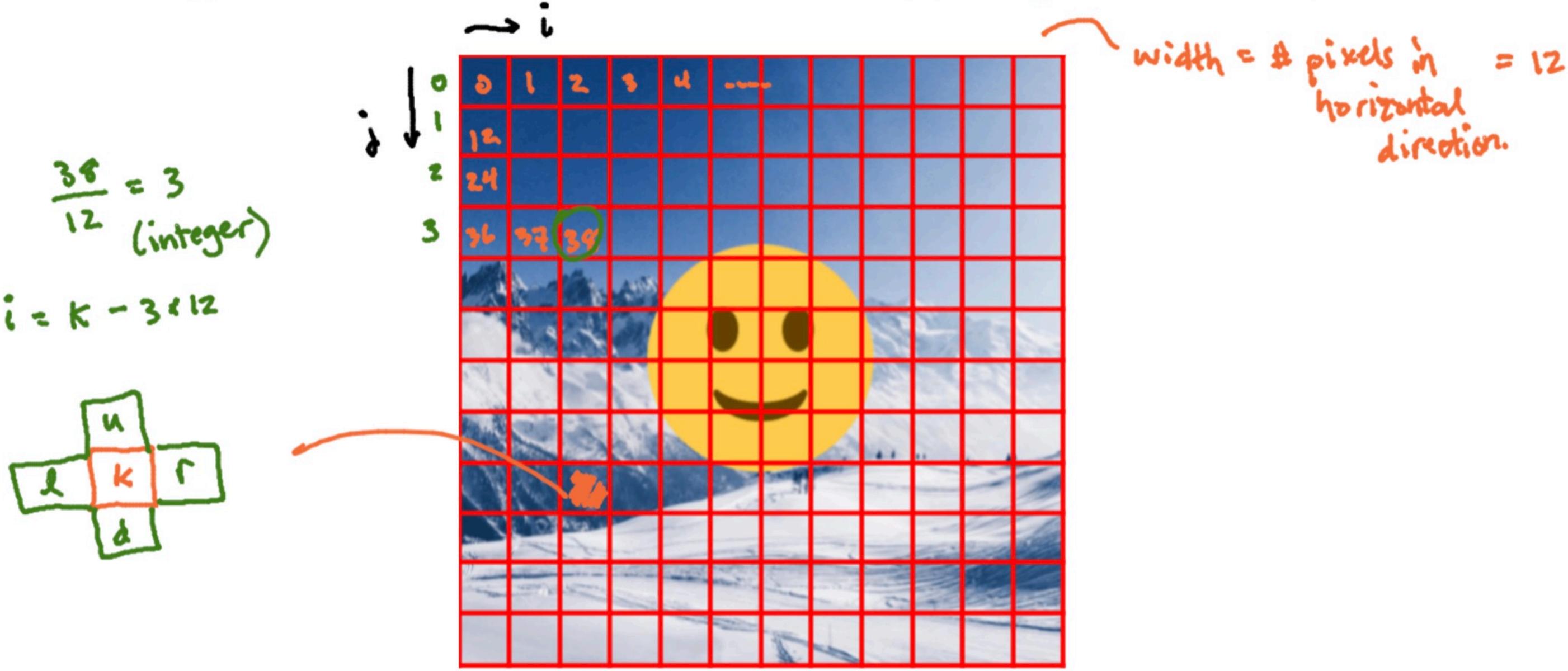


Now retrieve the changes in your Codespace.



Now we know how to **add** (stage), **commit**, **push** and **pull**.

Here, pixels are stored in a one-dimensional array (**ArrayList<Pixel>**).



- Every pixel outside of the circle will be set to the average of the surrounding pixels (right, left, above, below).
- To determine if a pixel is outside the smiley face circle, we need the row and column indices of the pixel.
- How to retrieve **i** (column) and **j** (row) indices from pixel index **k**?
- How to retrieve the surrounding pixels (right, left, above, below)?
- We will not modify pixels on the image boundary. How to detect these and skip them?



Possible implementation of the **blurBackground** method.

```
1 public void blurBackground(int faceRadius, int numIterations) {
2     int cx = width / 2;
3     int cy = height / 2;
4
5     for (int iter = 0; iter < numIterations; iter++) {
6         for (int k = 0; k < pixels.size(); k++) {
7             int j = k / width;
8             int i = k - j * width;
9
10            if (i == 0 || i + 1 == width || j == 0 || j + 1 == height) {
11                continue;
12            }
13
14            int radiusSquared = (i - cx) * (i - cx) + (j - cy) * (j - cy);
15            if (radiusSquared < faceRadius * faceRadius) {
16                continue;
17            }
18            Pixel pixel = pixels.get(k); // or j * width + i
19            Pixel r = pixels.get(j * width + i + 1); // or k + 1
20            Pixel d = pixels.get((j + 1) * width + i); // or k + width
21            Pixel l = pixels.get(j * width + i - 1); // or k - 1
22            Pixel u = pixels.get((j - 1) * width + i); // or k - width
23            pixel.average(r, d, l, u);
24        }
25    }
26 }
```

Now, **add** (click the **+** button for **Image.java**), **commit** (adding a message), then sync (**push**).

It's a good idea to stop your codespace.

The screenshot shows the GitHub Codespace interface for a repository named 'organic guacamole'. The interface includes a file explorer on the left with files like '.devcontainer', 'Image.java', 'README.md', and 'smiley.ppm'. A central panel shows the 'Codespaces' section with a table of active workspaces. The 'organic guacamole' workspace is active on the 'main' branch. A context menu is open over the workspace, with the 'Stop codespace' option highlighted in a red box. Other options in the menu include 'Rename', 'Export changes to a branch', 'Change machine type', 'Auto-delete codespace', 'Open in Browser', 'Open in Visual Studio Code', 'Open in JetBrains Gateway', 'Open in JupyterLab', and 'Delete'.

main 1 Branch 0 Tags

Go to file

Code

About

No description, website, or topics provided.

Readme

Activity

Custom properties

0 stars

Local Codespaces

Codespaces

Your workspaces in the cloud

On current branch

organic guacamole

Active

main No changes

Codespace usage for this repository is paid for by philipcla

Rename

Export changes to a branch

Change machine type

Stop codespace

Auto-delete codespace

Open in Browser

Open in Visual Studio Code

Open in JetBrains Gateway Preview

Open in JupyterLab Preview

Delete

philipclaude added more info about the program

.devcontainer initial commit

Image.java initial commit

README.md added more info

smiley.ppm initial commit

README

CSCI 201 Lecture 22

Author: Philip Caplan

About

This program blurs the background of an image containing a circle in the center.

Additional notes:

- For more info: <https://code.visualstudio.com/docs/sourcecontrol/intro-to-git>
- **Homework 10** due Tuesday 12/10.
- Last Exit Ticket (12T): **install a few tools** before Thursday's class.
- Thursday we will talk about **testing**.
- Friday's lab (**Lab 10**) will have two components:
 - We will complete the **Course Response Form** in the lab period.
 - You will submit your reflection for your **Participation** grade.
- **Final Exam:** (released Tuesday 12/10 and will be due Friday 12/13)
 - **Two components:** Part 1 on Canvas and Part 2 (programming) submitted to Gradescope.
 - **Study Guide** will be posted by the end of the week.

