Using VS Code in 5 Pages

(specific to CSE1325, NOT a general VS Code tutorial!)

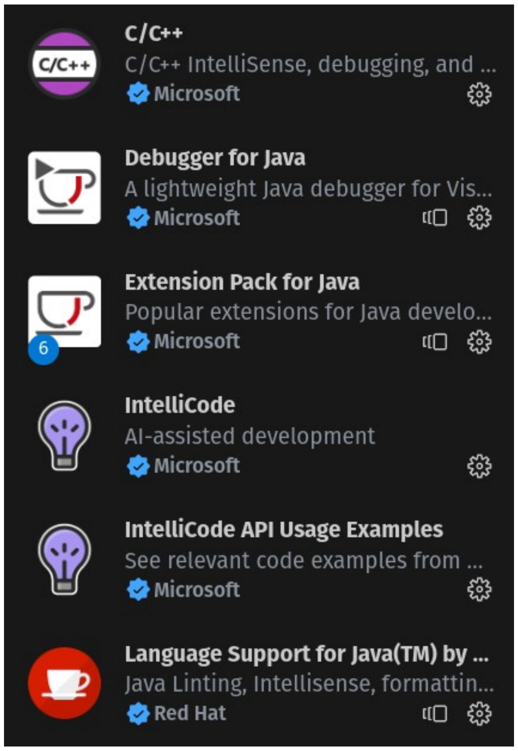
1 Getting Started

VS Code is a very nice Microsoft-supplied tool for editing code. It is also a sophisticated Integrated Development Environment (IDE), which means it will try to automate a lot of your work in ways you are not yet prepared to understand. ***Do not let it*!**

For CSE1325, you *may* but are NOT required to use VS Code – but ONLY as just a smart editor, nothing more. You are also welcome to use a simpler editor such as gedit, Notepad++, or Sublime.

Installing and Configuring VS Code Locally

1. If using GitHub Codespaces, skip to section 2. If using the CSE-VM VirtualBox appliance, skip to step 2 of this section. To install VS Code on your Linux, Mac, or Windows machine, follow instructions at https://code.visualstudio.com/download.
2. Select View > Extensions. In the "Search Extensions in Marketplace" text box, find and install and enable ONLY the following extensions to support this class. **Disable all other extensions!**



C/C++
C/C++ IntelliSense, debugging, and ...
Microsoft

Debugger for Java
A lightweight Java debugger for Vis...
Microsoft

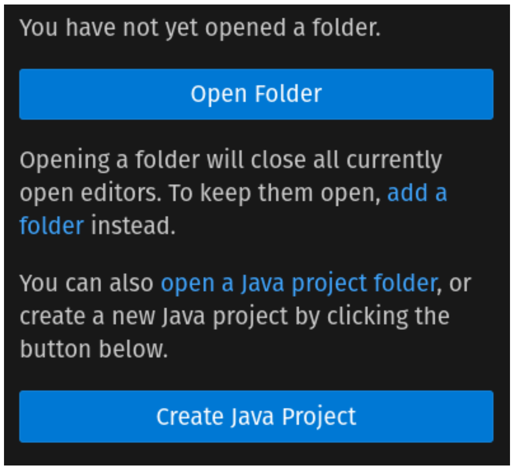
Extension Pack for Java
Popular extensions for Java develo...
Microsoft

IntelliCode
AI-assisted development
Microsoft

IntelliCode API Usage Examples
See relevant code examples from ...
Microsoft

Language Support for Java(TM) by ...
Java Linting, Intellisense, formattin...
Red Hat

3. VS Code will prompt you to Create Java Project. **Do NOT create a project in VS Code!** WARNING: If you create a project in the cse1325 directory, VS Code will write INCORRECT code into your homework and reduce your grade. Instead, **select Open Folder** where the program or package actually begins (usually the full_credit, bonus, or extreme_bonus directory) so that you can use VS Code as a very nice, simple text editor.

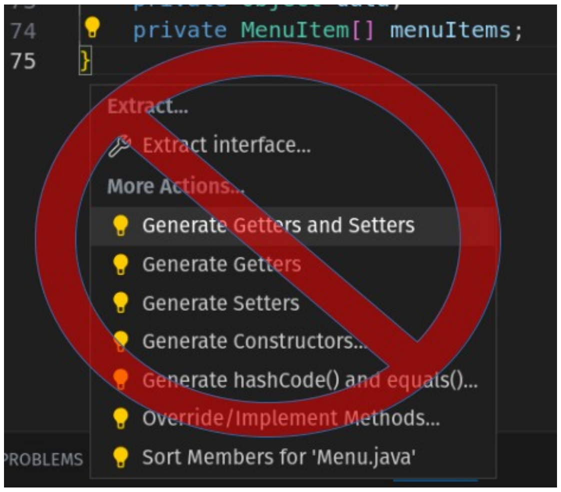


You have not yet opened a folder.

Opening a folder will close all currently open editors. To keep them open, add a folder instead.

You can also open a Java project folder, or create a new Java project by clicking the button below.

4. A**lways open from the root of each of your Java packages** (full_credit, bonus, or extreme_bonus) to ensure VS Code doesn't modify your or add a new package statement. **Avoid any menu item that says "Generate"!** First, that's unethical – the TAs are grading what YOU wrote, not what VS Code wrote. Yes, we can usually tell the difference, and generating code is cheating and *will* be reported. Second, because we aren't using a VS Code *project*, the code generated will likely not even be correct.

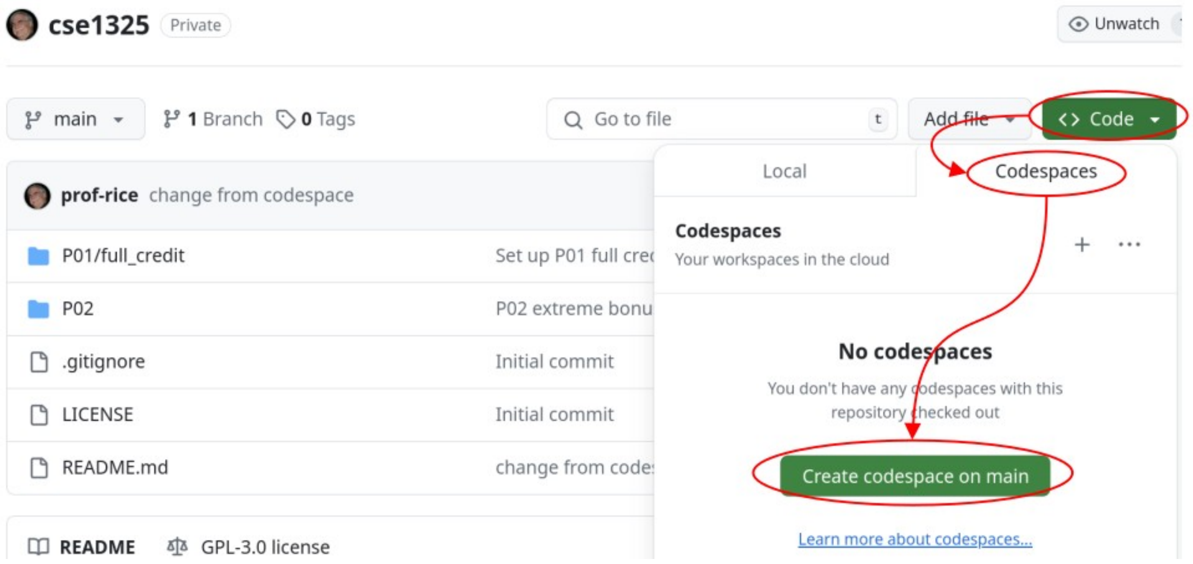


```
    private MenuItem[] menuItems;
}
```

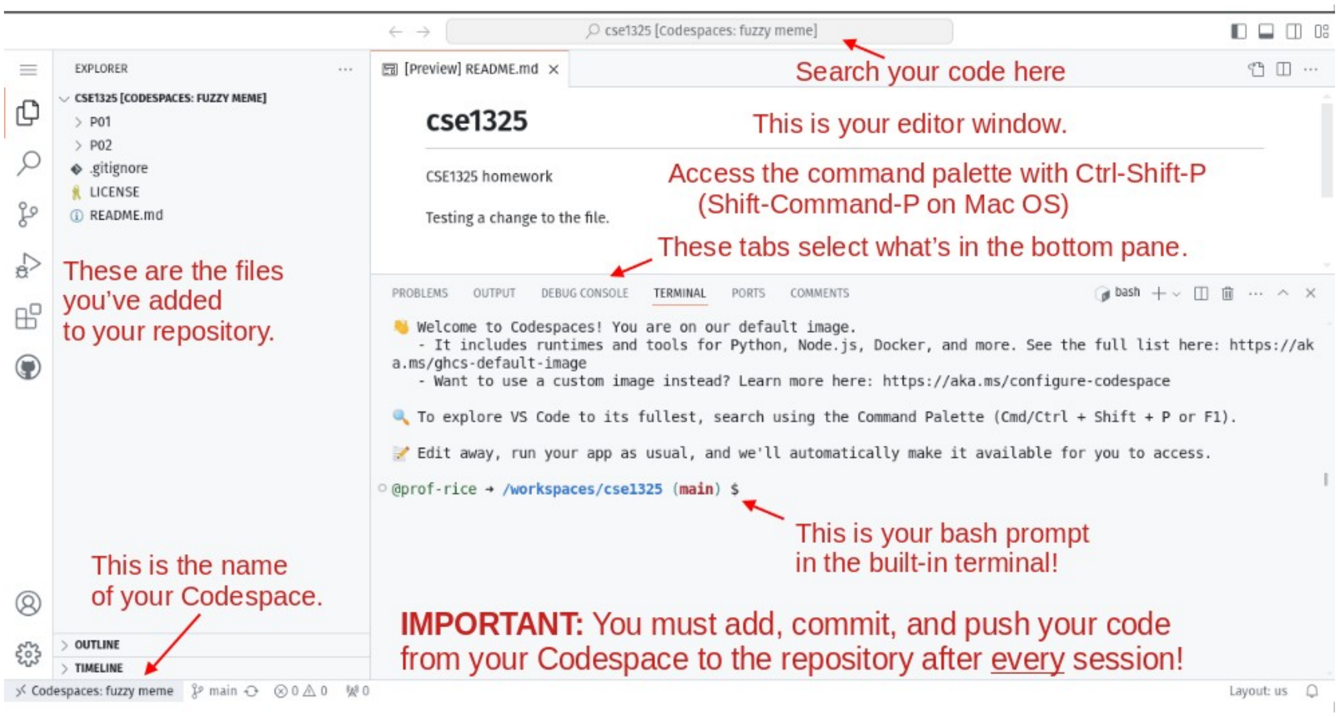
Extract...
Extract interface...
More Actions...
Generate Getters and Setters
Generate Getters
Generate Setters
Generate Constructors...
Generate hashCode() and equals()...
Override/Implement Methods...
Sort Members for 'Menu.java'
PROBLEMS

2 Using VS Code in GitHub Codespaces

GitHub provides a free online VS Code development environment. To use it, open your repository on the GitHub website, and select **Code** > **Codespaces** > **Create codespace on main**.



That's it – you have a Codespace running in the cloud! You may run as many Codespaces as you like.



```
👋 Welcome to Codespaces! You are on our default image.
   - It includes runtimes and tools for Python, Node.js, Docker, and more. See the full list here: https://ak
a.ms/ghcs-default-image
   - Want to use a custom image instead? Learn more here: https://aka.ms/configure-codespace

🔍 To explore VS Code to its fullest, search using the Command Palette (Cmd/Ctrl + Shift + P or F1).

📝 Edit away, run your app as usual, and we'll automatically make it available for you to access.

@prof-rice ➜ /workspaces/cse1325 (main) $
```

Search your code here

This is your editor window.

Access the command palette with Ctrl-Shift-P (Shift-Command-P on Mac OS)

These tabs select what's in the bottom pane.

These are the files you've added to your repository.

This is your bash prompt in the built-in terminal!

This is the name of your Codespace.

IMPORTANT: You must add, commit, and push your code from your Codespace to the repository after every session!

Important Warnings

If you don't use your running Codespace, it will stop itself after 30 minutes. If you don't relaunch a Codespace for 30 days, it will be deleted a*long with all of your uncommitted code*.

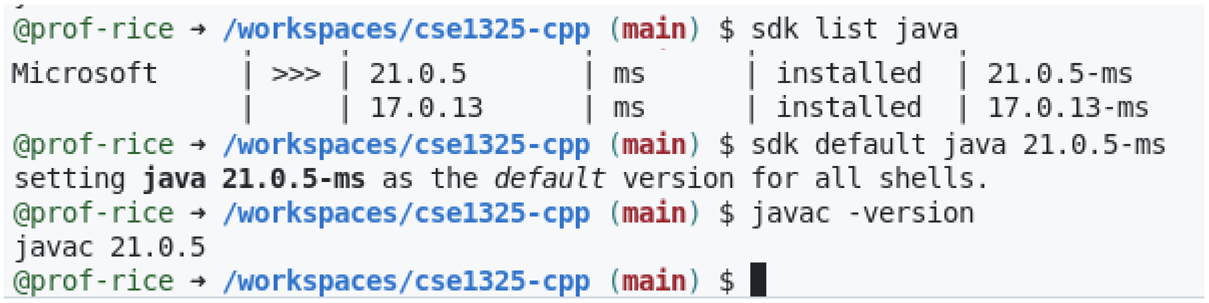
For this reason, **it's VERY important to always add, commit, and push code into GitHub proper!**

- When you begin working on your laptop, desktop, or Codespace, **always start with `git pull` to retrieve changes from your repository.**
- When you finish working on your laptop, desktop, or Codespace, **always use `git add; git commit -m 'message' ; git push`** to push the changes you made back to your repository.
  - **WARNING**: A Codespace is *temporary* unless you pay Microsoft for a permanent one. **Your code is NEVER automatically pushed to the repository!** ALWAYS push it yourself!

Adding Java 21 and ant to Each Codespace

Codespace uses a unique package manager very different from Ubuntu's Advanced Package Tool (apt) called Software Development Kit Manager (SDKMAN), which excels at managing parallel versions of the same tools. The base command is **`sdk`**; you can get a summary of commands with **`sdk help`**. You'll find other useful commands at https://sdkman.io/usage .

The default Codespace provides Java 11, quite out of date. To see all available versions, type **`sdk list java`** (type **`q`** to exit the list). Among the long list of versions available, the status of 21.0.5-ms should be "installed". To make it your default version, type **`sdk default java 21.0.5-ms`**. You can verify the version with **`javac -version`** for the Java compiler and **`java -version`** for the Java runtime.

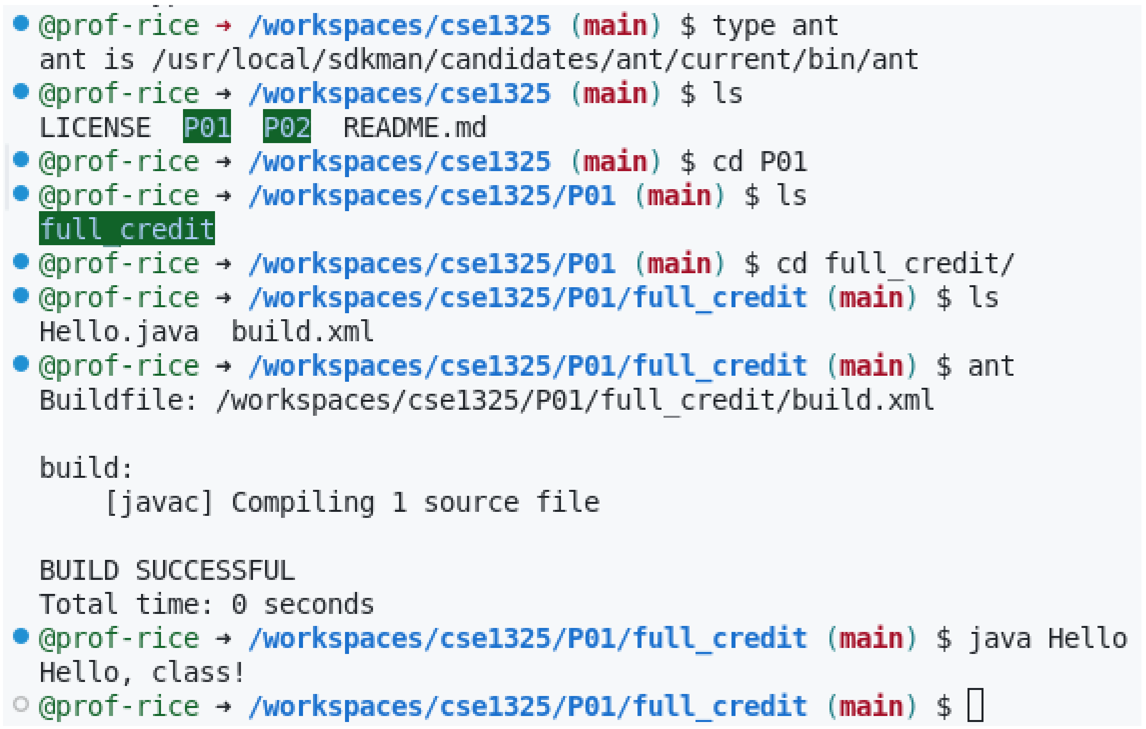


```
@prof-rice → /workspaces/cse1325-cpp (main) $ sdk list java
Microsoft      | >>> | 21.0.5        | ms     | installed  | 21.0.5-ms
               |     | 17.0.13       | ms     | installed  | 17.0.13-ms
@prof-rice → /workspaces/cse1325-cpp (main) $ sdk default java 21.0.5-ms
setting java 21.0.5-ms as the default version for all shells.
@prof-rice → /workspaces/cse1325-cpp (main) $ javac -version
javac 21.0.5
@prof-rice → /workspaces/cse1325-cpp (main) $
```

To install ant, follow a similar process. First, find the available versions with **`sdk list ant`**. The latest at the time of this writing was 1.10.13, so you would install using **`sdk install ant 1.10.13`**
A version is automatically set to default after installation. Use **`ant -version`** to check ant's version.

Not surprisingly, GitHub Codespaces pre-installs the git command line tool.

Now that you have a modern Java compiler with the ant and git tools, you're ready! Change to the P01/full_credit directory (your solution for assignment P01). Type **`ant`**, and your code should build. Then type **`java Hello`** to run your code in your Codespace!



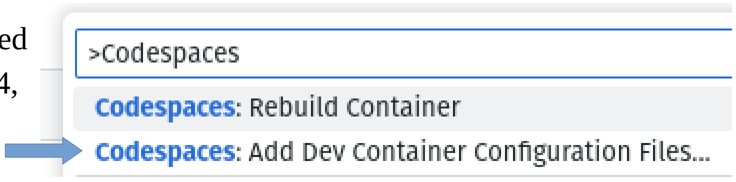
```
● @prof-rice → /workspaces/cse1325 (main) $ type ant
  ant is /usr/local/sdkman/candidates/ant/current/bin/ant
● @prof-rice → /workspaces/cse1325 (main) $ ls
  LICENSE  P01  P02  README.md
● @prof-rice → /workspaces/cse1325 (main) $ cd P01
● @prof-rice → /workspaces/cse1325/P01 (main) $ ls
  full_credit
● @prof-rice → /workspaces/cse1325/P01 (main) $ cd full_credit/
● @prof-rice → /workspaces/cse1325/P01/full_credit (main) $ ls
  Hello.java  build.xml
● @prof-rice → /workspaces/cse1325/P01/full_credit (main) $ ant
  Buildfile: /workspaces/cse1325/P01/full_credit/build.xml

  build:
      [javac] Compiling 1 source file

  BUILD SUCCESSFUL
  Total time: 0 seconds
● @prof-rice → /workspaces/cse1325/P01/full_credit (main) $ java Hello
  Hello, class!
○ @prof-rice → /workspaces/cse1325/P01/full_credit (main) $ 
```

Adding Gnu gcc to Each Codespace

Codespace may or may not include the gcc compiler suite. When it does, it seems to be gcc 9, which is too old for this class. For a modern version, you'll need to rebuild your workspace like this.

1. Select Ctrl-Shift-P or Cmd-Shift-P. Run **Codespaces: Add Dev Container Configuration Files…**

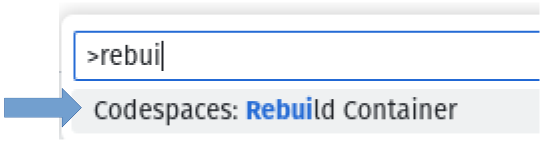
Select the C++ container configuration (you may need to ask for all configurations first), then Ubuntu 24.04, and then run. Now the file you need to edit is available in your Codespace.


>Codespaces
Codespaces: Rebuild Container
Codespaces: Add Dev Container Configuration Files...

2. Edit file .devcontainer/Dockerfile and append this text to the end starting on a new line *exactly*:

```
RUN apt-get update && export DEBIAN_FRONTEND=noninteractive \
    && apt-get -y install --no-install-recommends software-properties-common \
    && add-apt-repository ppa:ubuntu-toolchain-r/test \
    && apt-get update \
    && apt-get -y install --no-install-recommends gcc-12 g++-12 \
    && update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-12 100 \
    && update-alternatives --install /usr/bin/g++ g++ /usr/bin/g++-12 100
```

3. Select Ctrl-Shift-P or Cmd-Shift-P, type "rebuild", then select **Codespaces: Rebuild Container.** This will take awhile.



Once the CodeSpace has restarted, run **gcc -version** on the command line. It should be version 12!