

Coding Workflow

ROB 102: Introduction to AI & Programming

Lab Session 1

2021/09/03

Admin

Join Slack! (<https://um-fa21-rob102.slack.com/>)

Bookmark the course website (robotics102.org)

Office hours start next week

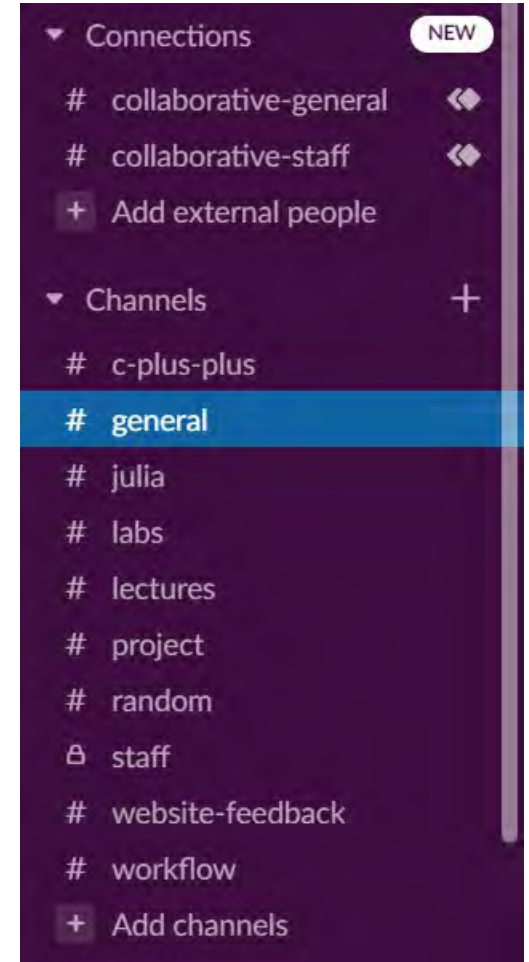
Robotics Building



Prof Jenkins: MW 1-3PM @ FRB 2236

Jana: Tu 10AM-12PM, W 3-5 PM @ FRB 2000

IAs: TBD

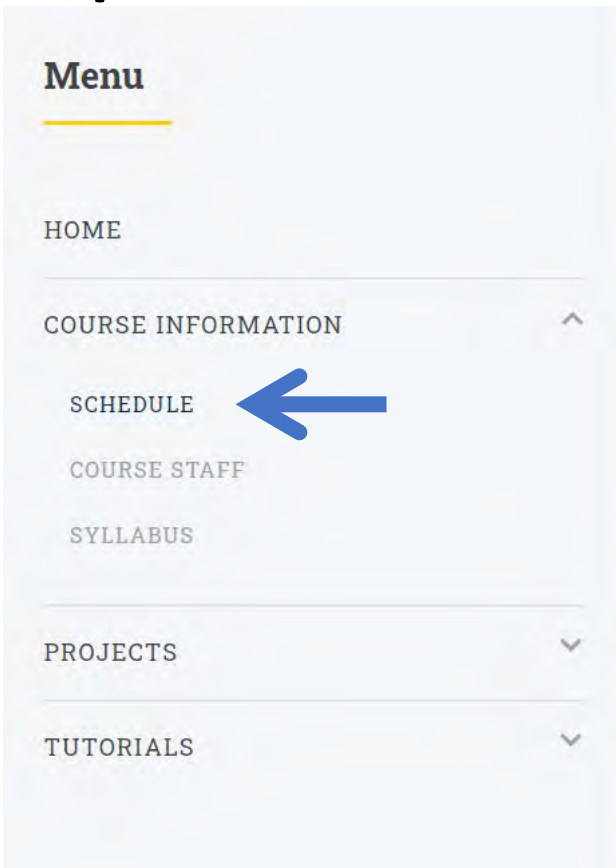


The lab space with the maze where we did the Pair Navigation demo.

Admin

Project 0 is out. We will go through it in class.

Due Sept. 20 at 11:59 PM.



ROB 102: Introduction to AI and Programming

Need ENGIN 101? Register for ROB 102 this Fall!

Learn how to code by making robots navigate autonomously!



Admin

Schedule

All lecture slides are available [here](#). They will be linked in the schedule once they are available.

Course Schedule (UMich)

Date	Topic	Readings	Project
Week 1			
Aug 30	Course Initialization Overview [Slides] Lecture Video: Hello World! [Slides]		Out: Project 0
Sept 1	In-Class Activity: Pair Navigation Lecture: Variables and Operators		
Sept 3	Lab: Coding Workflow		
Week 2			
Sept 6	Labor Day - No class		

← Watch the recorded lectures before next class.

← Link to project 0

← Labs are live (not recorded).

Admin

Schedule

All lecture slides are available [here](#). They will be linked in the schedule once they are available.

Course Schedule (UMich)

Date	Topic	Readings	Project
Sept 15	Lecture: Vectors		
Sept 17	Lab: Laser Rangefinding		
Week 4			
Sept 20	Lecture: Wall Following		Due: <u>Project 0</u>
Sept 22	Lecture: Structs Quiz 1		
Sept 24	Lab: Wall Following		
Week 5			



Project 0 due **Sept 20**
@ 11:59 PM

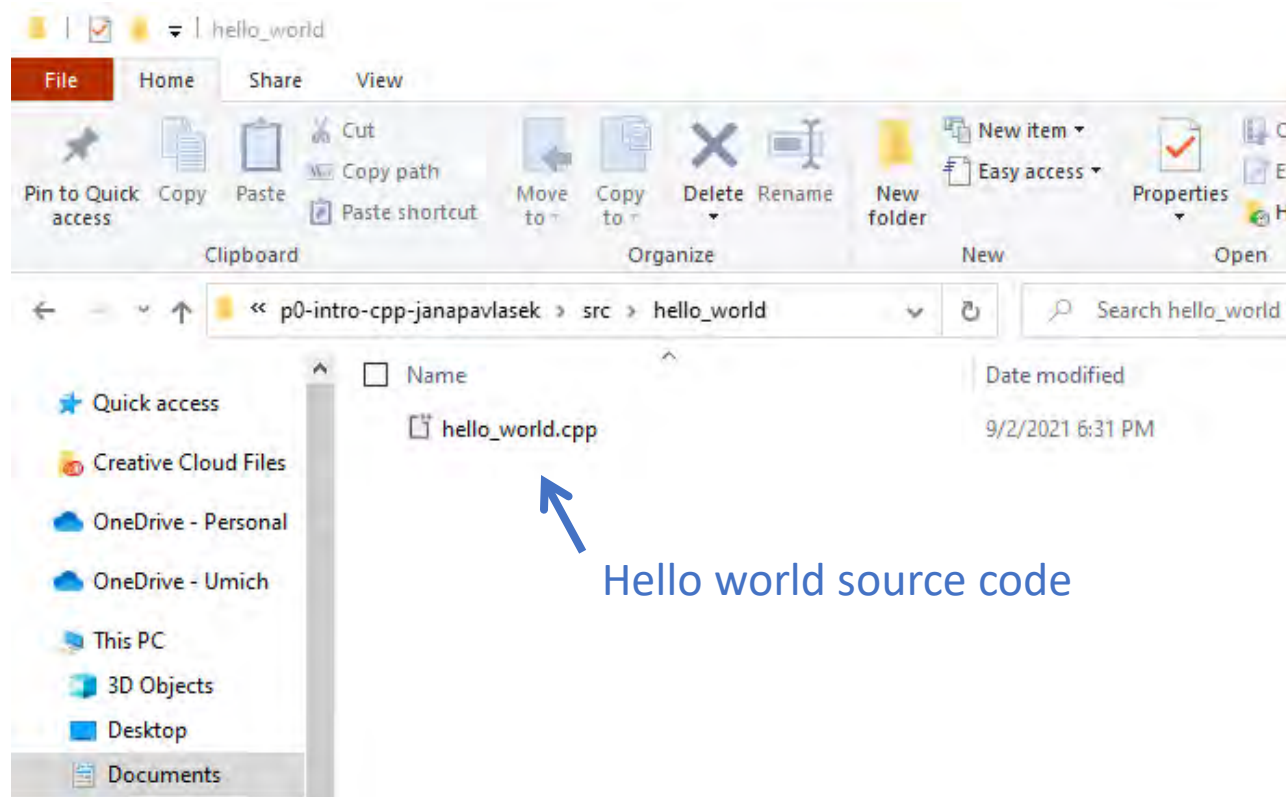
Today...

- Coding Workflow
- Git & version control
- VSCode & programming environment
- Docker & running your code
- *Computer setup help*

Tasks to complete:

- ✓ Install all software
- ✓ Push license to GitHub repo
- ✓ Run Hello World on your computer
- ✓ Push changes to your repository

Coding workflow



We need to get the code on our computers first!

Open code

Coding workflow

The screenshot shows a GitHub repository page for 'robotics102 / p0-intro-cpp-janapavlassek'. The repository is private and was generated from 'robotics102/intro-cpp'. The main branch is 'main'. The repository contains a 'src' directory and files like '.gitignore', 'README.md', and 'docker_run.sh'. The README.md file is open, showing the title 'ROB 102: Intro to C++' and a description: 'Template code for Project 0: Introduction to C++. See [project instructions](#) for...'. The right sidebar shows repository details, including 'About', 'Releases', and 'Packages'.

File	Commit	Time
src	Complete Hello World activity	yesterday
.gitignore	Initial commit	4 days ago
README.md	Initial commit	4 days ago
docker_run.sh	Initial commit	4 days ago

Pull code from GitHub



Open code

Coding workflow

```
hello_world.cpp X
src > hello_world > hello_world.cpp > ...
16 #include <iostream> // IO code for printing.
17
18 // The main function tells the compiler which code to execute.
19 int main(int argc, char** argv)
20 {
21     // TODO: Print out the message "Hello World! My name is __".
22     std::cout << "Hello World! My name is Jana\n";
23     return 0;
24 }
25
```

Pull code from GitHub



Open code



Update code

Coding workflow

```
hello_world.cpp X
src > hello_world > hello_world.cpp > ...
16 #include <iostream> // IO code for printing.
17
18 // The main function tells the compiler which code to execute.
19 int main(int argc, char** argv)
20 {
21     // TODO: Print out the message "Hello World! My name is __".
22     std::cout << "Hello World! My name is Jana\n";
23     return 0;
24 }
25
```

Pull code from GitHub



Open code in VSCode



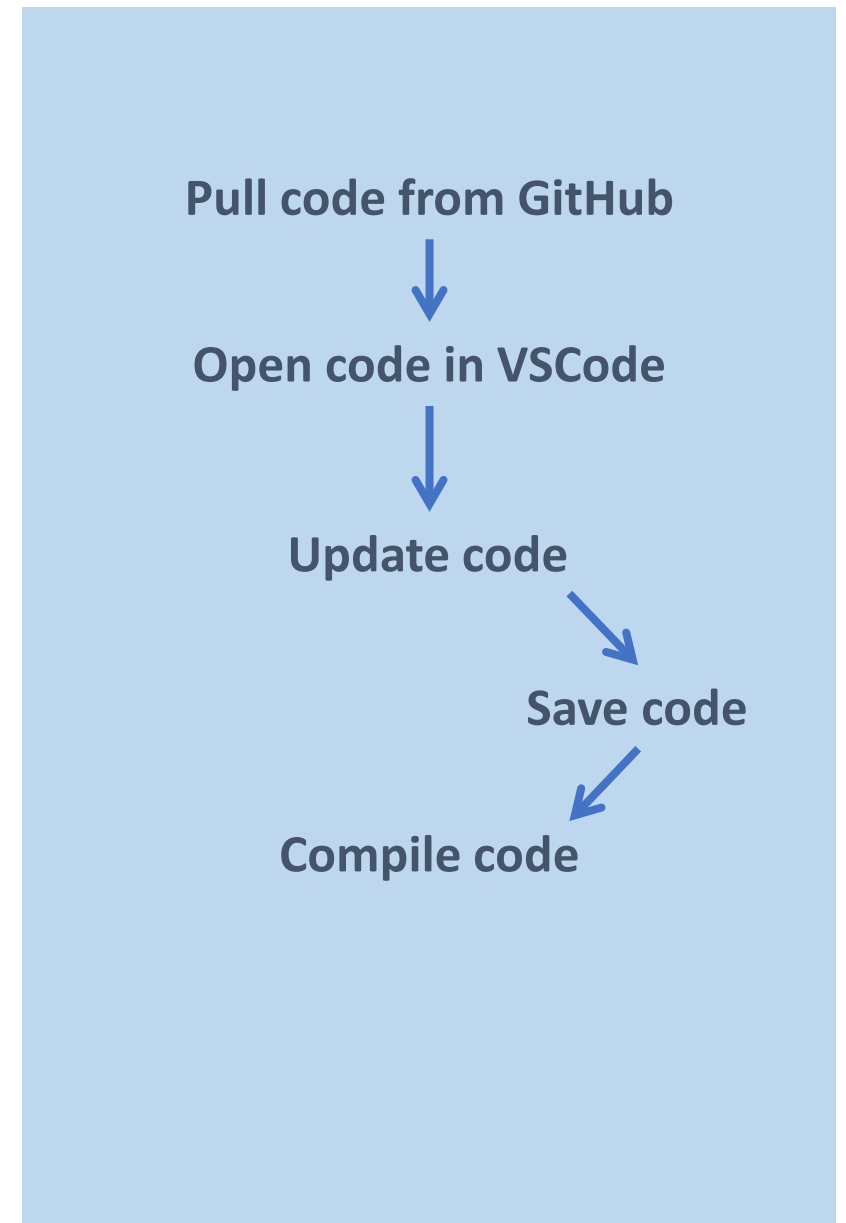
Update code



Save code

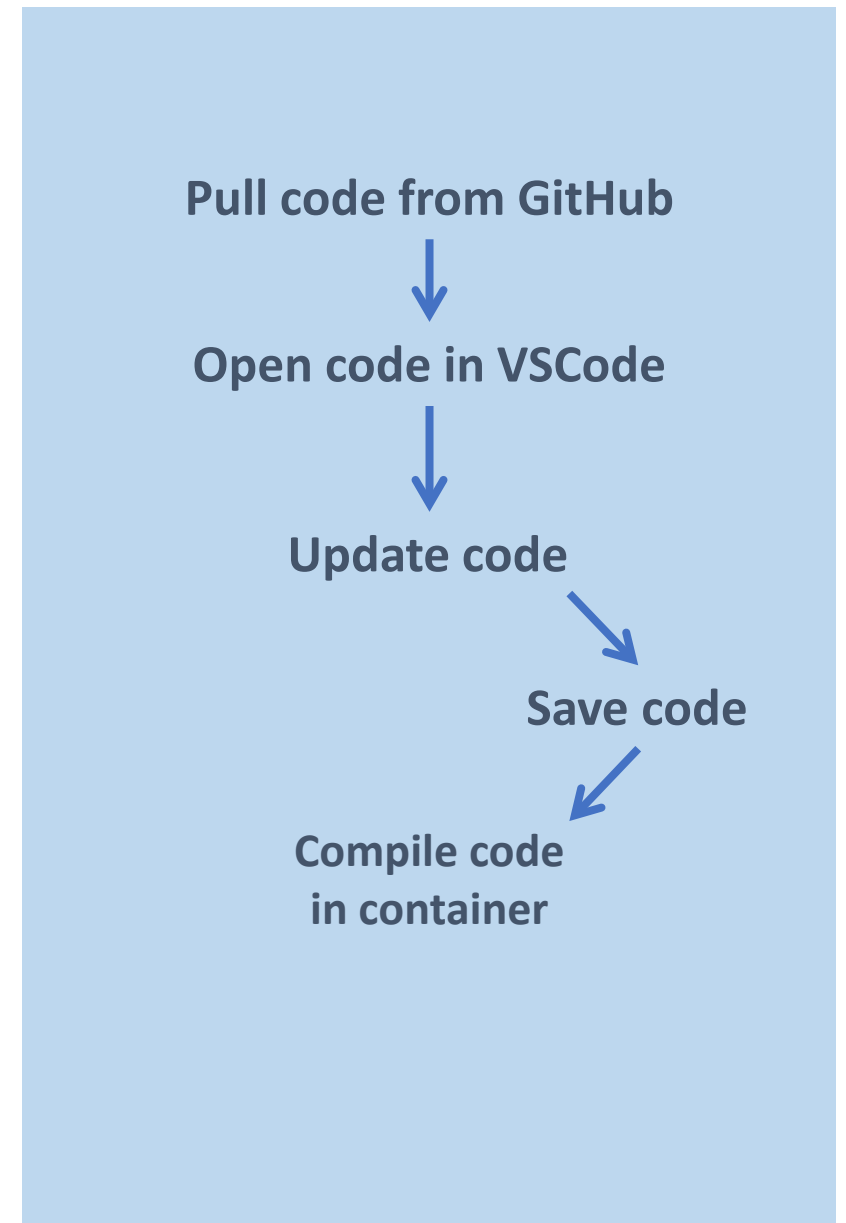
Coding workflow

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE
root@f44831118d31:/code/src/hello_world# g++ hello_world.cpp -o hello_world
root@f44831118d31:/code/src/hello_world#
```



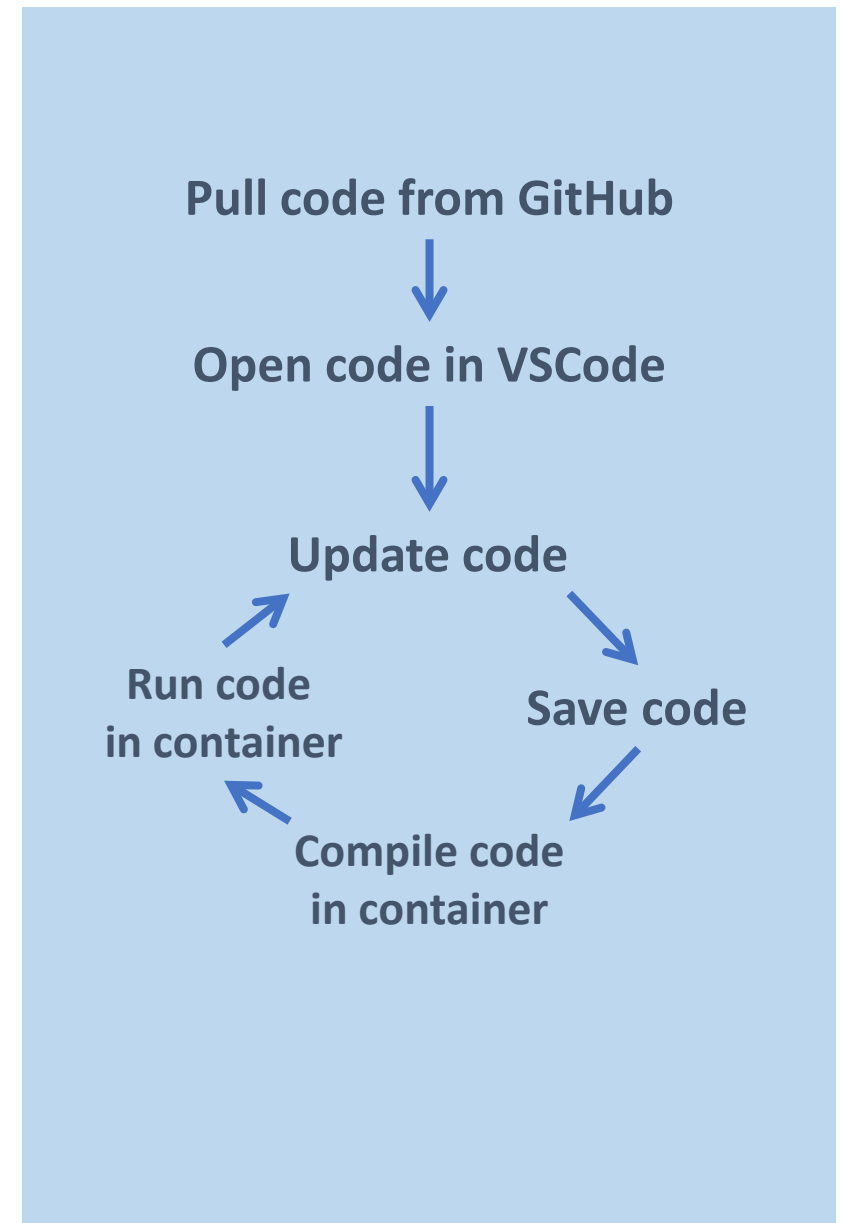
Coding workflow

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE
root@f44831118d31:/code/src/hello_world# g++ hello_world.cpp -o hello_world
root@f44831118d31:/code/src/hello_world#
```



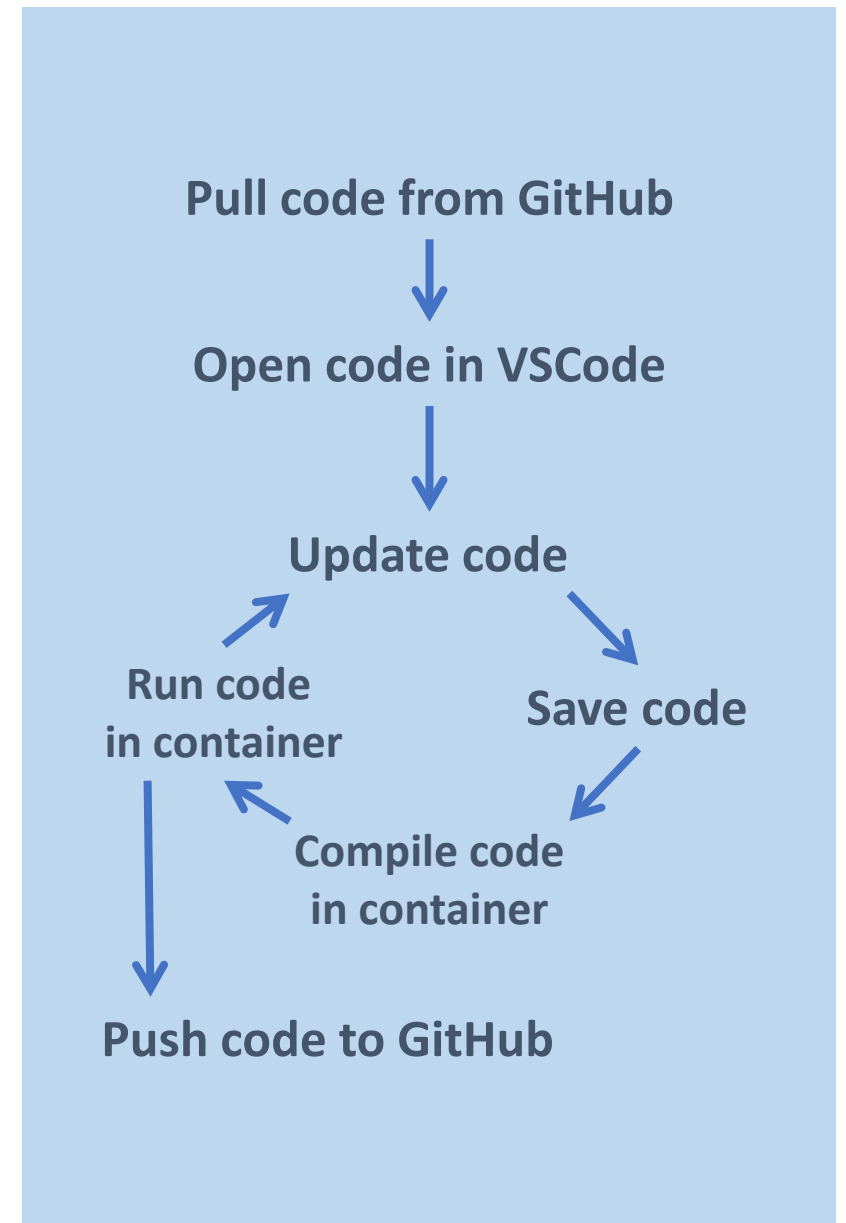
Coding workflow

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE
root@f44831118d31:/code/src/hello_world# ./hello_world
Hello World! My name is Jana
root@f44831118d31:/code/src/hello_world#
```



Coding workflow

The screenshot shows a GitHub repository page for 'robotics102 / p0-intro-cpp-janapavlassek'. The repository is private and has 3 unwatched items, 0 stars, and 0 forks. The main branch is 'main'. The repository contains a 'src' directory and files like '.gitignore', 'README.md', and 'docker_run.sh'. The README.md file is open, showing the title 'ROB 102: Intro to C++' and a description: 'Template code for Project 0: Introduction to C++. See [project instructions](#) for...'. The repository was created by GitHub Classroom and has 1 tag and no packages published.



What is Git?

Git is **version control** software, meaning that it tracks changes to your files as you work on them over time

Similar to “track changes” feature in document writing programs, except you must choose which versions to include in the tracking—it is not automatic

Widely used in academia and industry

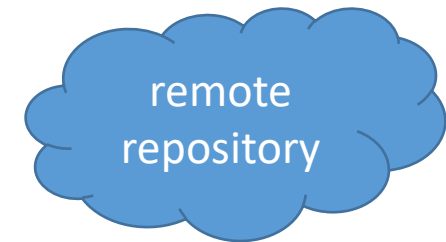
The only way for you to submit your homework for this course!

Git Workflow

The screenshot shows a GitHub repository page. At the top, there is a search bar and navigation links for Pulls, Issues, Marketplace, and Explore. The repository name is 'robotics102 / p0-intro-cpp-janapavlese' and it is marked as 'Private'. Below the repository name, there are statistics for Unwatch (3), Star (0), and Fork (0). The main navigation bar includes Code, Issues, Pull requests, Actions, Projects, Security, and Insights. The repository is currently on the 'main' branch. A file list shows a directory 'src' and files '.gitignore', 'README.md', and 'docker_run.sh'. The README content is visible, starting with 'ROB 102: Intro to C++' and 'Template code for Project 0: Introduction to C++'. On the right side, there are sections for 'About' (p0-intro-cpp-janapavlese created by GitHub Classroom), 'Releases' (1 tags), and 'Packages' (No packages published).

The cloud

Your Git repository in the cloud
https://github.com/gituname/my_repo

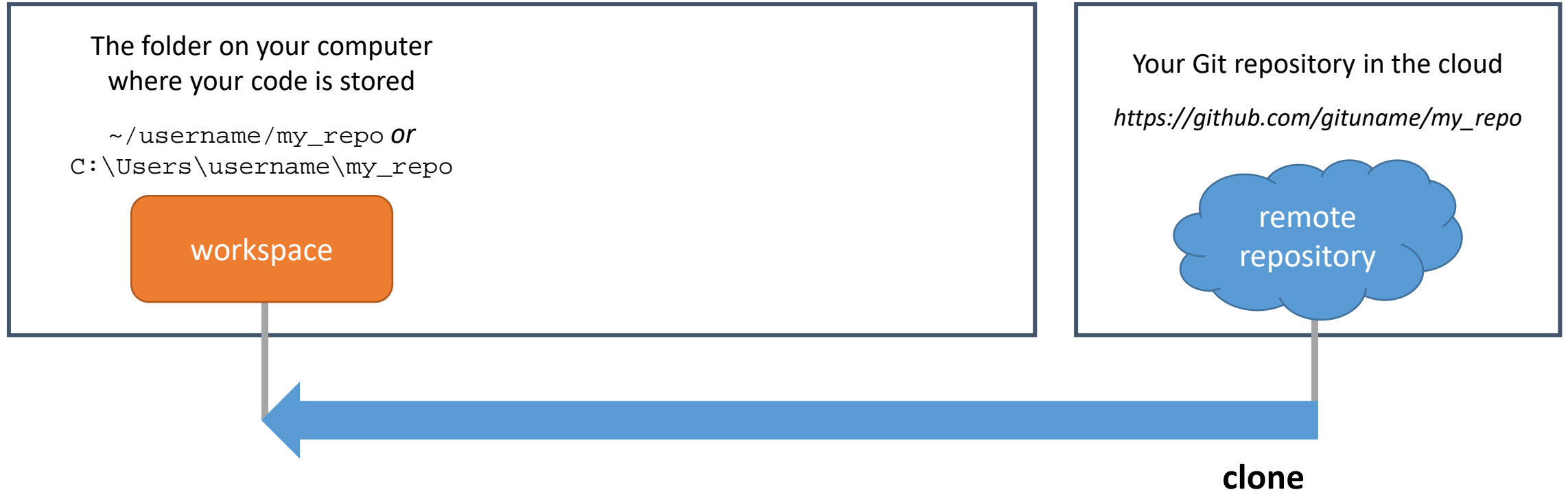


[https://github.com/robotics102/
p0-intro-cpp-janapavlese](https://github.com/robotics102/p0-intro-cpp-janapavlese)

Getting the code

Your computer

The cloud



Where's my code?

Your computer

The folder on your computer
where your code is stored

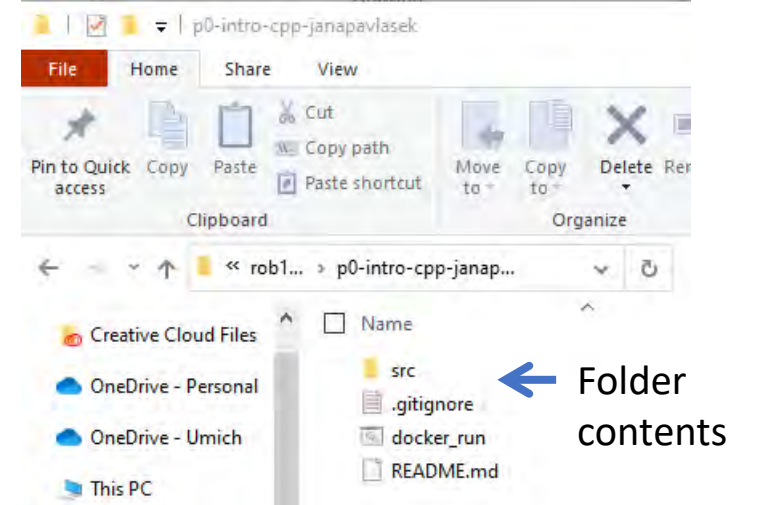
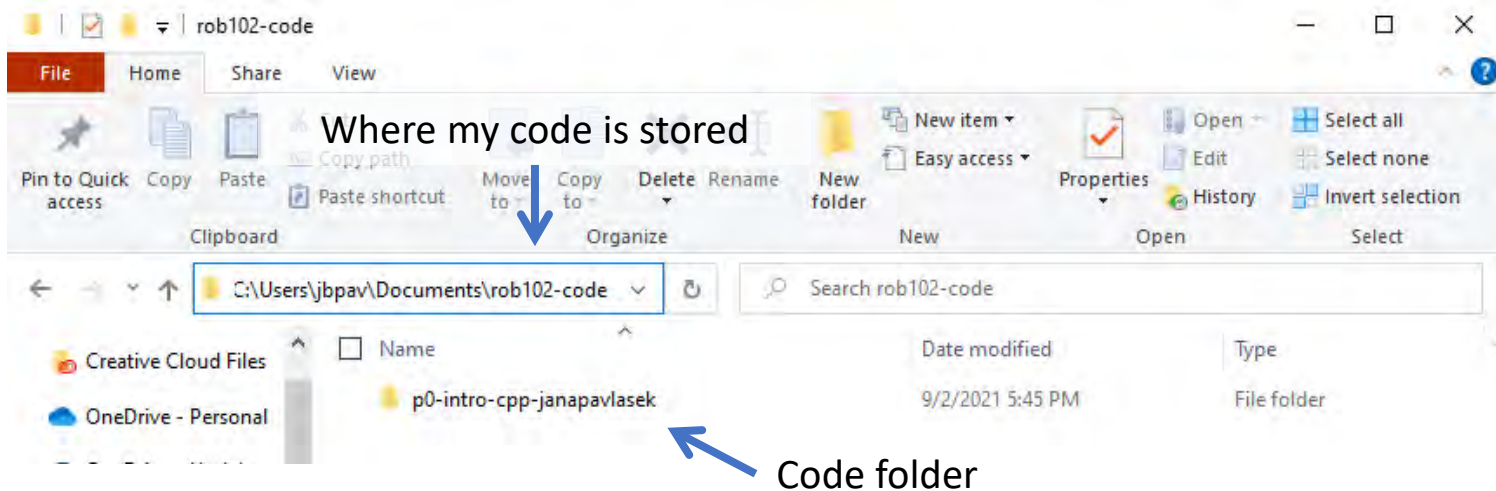
~/username/my_repo *OR*
C:\Users\username\my_repo

workspace

The cloud

Your Git repository in the cloud
https://github.com/robotics102/my_repo

remote
repository



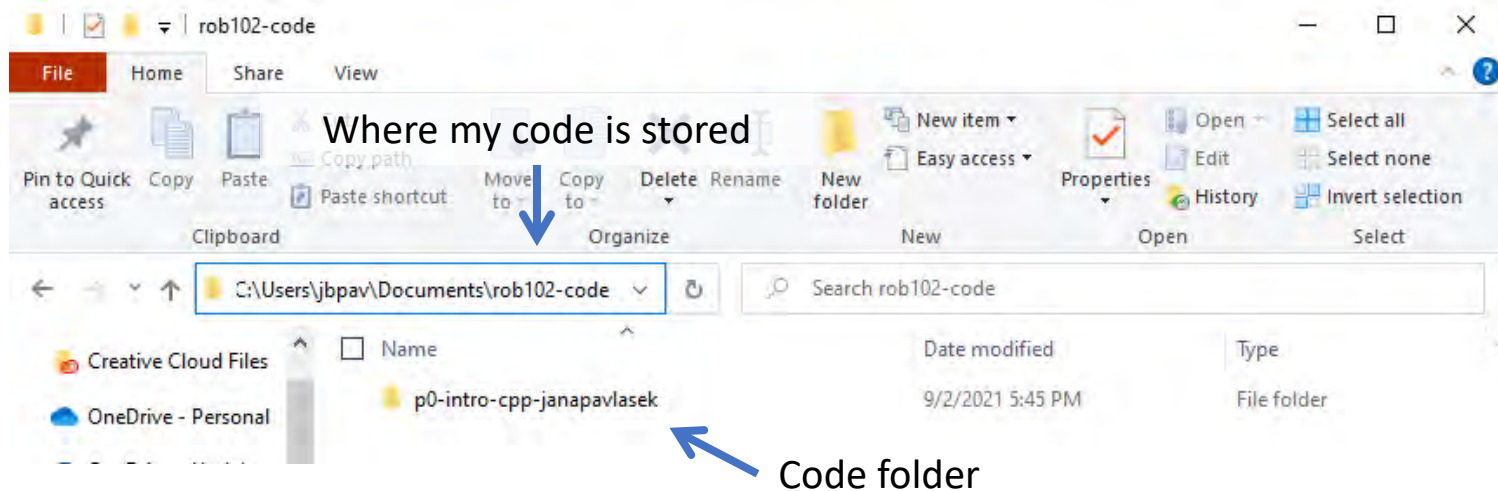
Where's my code?

Your computer

The folder on your computer
where your code is stored

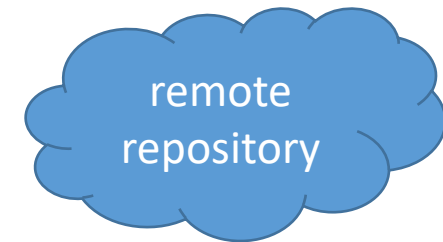
~/username/my_repo *OR*
C:\Users\username\my_repo

workspace



The cloud

Your Git repository in the cloud
https://github.com/robotics102/my_repo



Your computer organizes all the applications and files it contains in its **filesystem**, under different *folders* (like Documents, Downloads, etc...)

Make sure you know where in your computer's filesystem you are storing your code!!

Where's my code?

Your computer

The folder on your computer
where your code is stored

~/username/my_repo *OR*
C:\Users\username\my_repo

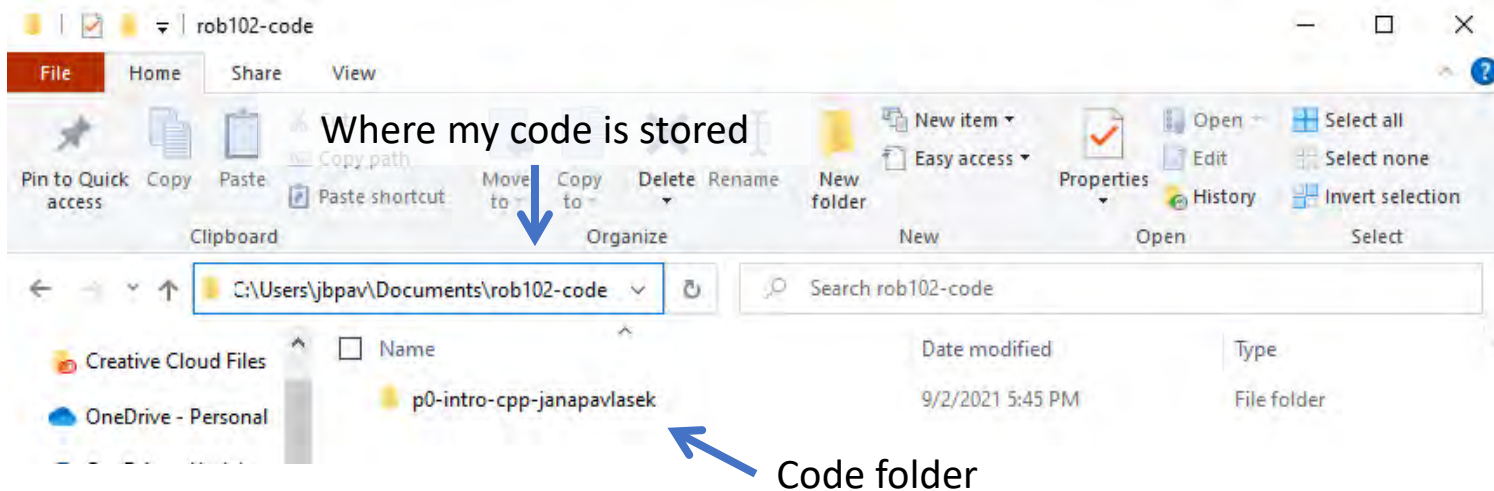
workspace

The cloud

Your Git repository in the cloud

https://github.com/robotics102/my_repo

remote
repository



Tip: Create a folder for all your ROB 102 code files (ex. in Documents folder).

Tip: Give the folder a name without spaces! We'll be using the command line, and spaces will be hard to deal with (ex. `rob102-code`, not `ROB 102 Code`).

Clone a repository: Your turn!

If you haven't already, make a GitHub account

- <https://github.com>

If you haven't already, install Git

- <https://robotics102.github.io/tutorials/setup>

If you haven't already, accept the assignment for P0

- <https://robotics102.github.io/projects/a0>

Getting the code

We will use GitHub Classroom to manage assignments. Use the following invite link to accept the assignment on the Github Classroom:



Accept the assignment:

<https://classroom.github.com/a/EXM5NKBh>

Clone a repository: Your turn!

1. Make a GitHub account (github.com)
2. Install Git
(robotics102.github.io/tutorials/setup)
3. Accept P0 assignment
(robotics102.github.io/projects/a0)

1. Make a folder where you will store your code for ROB 102.
2. Open a terminal.
3. Use the `cd` command go to the folder you created.
4. Use the `git clone` command to clone your code.

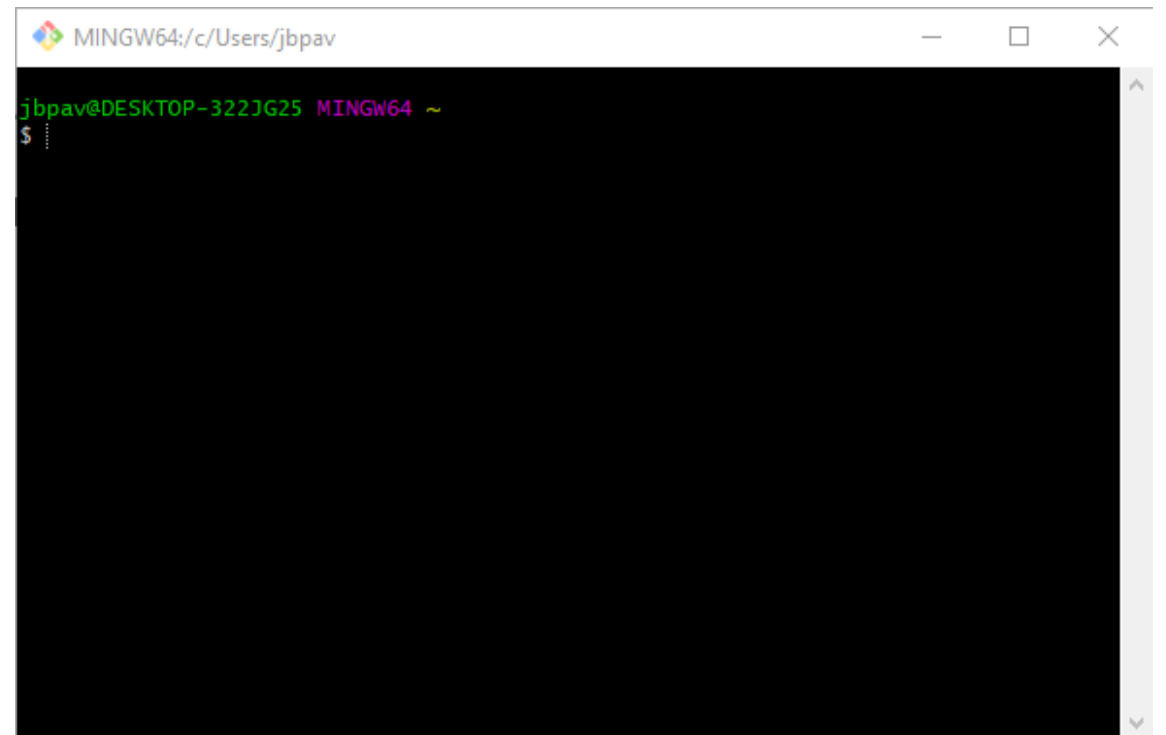
Let's go through the steps together.

Clone a repository: Your turn!

1. Make a folder where you will store your code for ROB 102
 - Remember: No spaces!
 - Good names: `rob102-code`, `ROB102Code`, etc.
 - Bad names: `ROB 102 Code`, etc.

Clone a repository: Your turn!

1. Make a folder where you will store your code for ROB 102.
2. Open a terminal
 - Windows: Search for “Git Bash”
 - Mac: Search for “Terminal”



Clone a repository: Your turn!

1. Make a folder where you will store your code for ROB 102
2. Open a terminal
3. Use the `cd` command go to the folder you created.



Our first terminal command!

`cd` stands for **C**hange **D**irectory.

Our terminal is always open in some directory (the “working” directory) on the filesystem.

The working directory is listed before the cursor. Or, use `pwd` to **P**rint **W**orking **D**irectory.

Terminal opens in the “home” directory (~)

```
MINGW64:/c/Users/jbpav/Documents/rob102-code
jbpav@DESKTOP-322JG25 MINGW64 ~
$ pwd
/c/Users/jbpav
jbpav@DESKTOP-322JG25 MINGW64 ~
$ cd Documents/rob102-code/
jbpav@DESKTOP-322JG25 MINGW64 ~/Documents/rob102-code
$
$
```

`cd` changes the directory

Clone a repository: Your turn!

1. Make a folder where you will store your code for ROB 102
2. Open a terminal
3. Use the `cd` command go to the folder you created.



Our first terminal command!

`cd` stands for **C**hange **D**irectory.

Our terminal is always open in some directory (the “working” directory) on the filesystem.

The working directory is listed before the cursor. Or, use `pwd` to **P**rint **W**orking **D**irectory.

```
jbpav@DESKTOP-322JG25 MINGW64 ~/Documents/rob102-code
$ cd a-fake-dir
bash: cd: a-fake-dir: No such file or directory

jbpav@DESKTOP-322JG25 MINGW64 ~/Documents/rob102-code
$ .....
```

The folder must exist, or you'll get an error!

Clone a repository: Your turn!

1. Make a folder where you will store your code for ROB 102
2. Open a terminal
3. Use the `cd` command go to the folder you created.
4. Use the `git clone` command to clone your code.

← Another terminal command!

All Git terminal commands look like:

Starts with `git`

→ `git cmd [arguments]`

← Some commands have arguments.

↑
A Git command
(`clone`, `commit`, `pull`...)

Remember how we didn't put spaces in our folder names?
In the terminal, spaces separate different commands and arguments!

Clone a repository: Your turn!

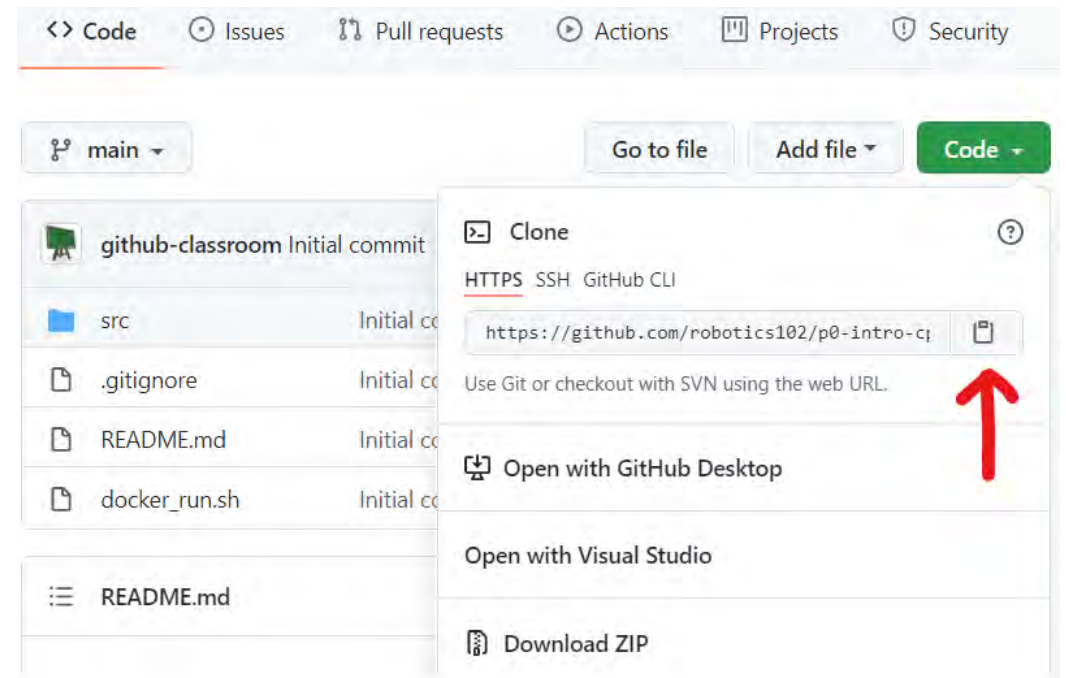
1. Make a folder where you will store your code for ROB 102
2. Open a terminal
3. Use the `cd` command go to the folder you created.
4. Use the `git clone` command to clone your code.

All Git terminal commands look like:

```
git cmd [arguments]
```

To clone your Git repository, type:

```
git clone ADDRESS-TO-YOUR-REPO
```



Clone a repository: Your turn!

1. Make a folder where you will store your code for ROB 102
2. Open a terminal
3. Use the `cd` command go to the folder you created.
4. Use the `git clone` command to clone your code.

All Git terminal commands look like:

```
git cmd [arguments]
```

To clone your Git repository, type:

```
git clone ADDRESS-TO-YOUR-REPO
```

```
jbpav@DESKTOP-322JG25 MINGW64 ~
$ cd Documents/rob102-code/

jbpav@DESKTOP-322JG25 MINGW64 ~/Documents/rob102-code
$ git clone https://github.com/robotics102/p0-intro-cpp-janapavlasek.git
Cloning into 'p0-intro-cpp-janapavlasek'...
remote: Enumerating objects: 15, done.
remote: Counting objects: 100% (15/15), done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 15 (delta 2), reused 12 (delta 2), pack-reused 0
Receiving objects: 100% (15/15), done.
Resolving deltas: 100% (2/2), done.

jbpav@DESKTOP-322JG25 MINGW64 ~/Documents/rob102-code
$ ls
p0-intro-cpp-janapavlasek/

jbpav@DESKTOP-322JG25 MINGW64 ~/Documents/rob102-code
$
```

Clone a repository: Your turn!

1. Make a folder where you will store your code for ROB 102
2. Open a terminal
3. Use the `cd` command go to the folder you created.
4. Use the `git clone` command to clone your code.

`ls` is another useful terminal command to list all the files in a directory

My directory now contains a single folder with the same name as the repository

Adding the name of a folder lists the contents of that folder

```
jbpav@DESKTOP-322JG25 MINGW64 ~
$ cd Documents/rob102-code/

jbpav@DESKTOP-322JG25 MINGW64 ~/Documents/rob102-code
$ git clone https://github.com/robotics102/p0-intro-cpp-janapavlasek.git
Cloning into 'p0-intro-cpp-janapavlasek'...
remote: Enumerating objects: 15, done.
remote: Counting objects: 100% (15/15), done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 15 (delta 2), reused 12 (delta 2), pack-reused 0
Receiving objects: 100% (15/15), done.
Resolving deltas: 100% (2/2), done.

jbpav@DESKTOP-322JG25 MINGW64 ~/Documents/rob102-code
$ ls
p0-intro-cpp-janapavlasek/

jbpav@DESKTOP-322JG25 MINGW64 ~/Documents/rob102-code
$ ls p0-intro-cpp-janapavlasek/
README.md  docker_run.sh*  src/
```

Clone a repository: Your turn!

1. Make a GitHub account (github.com)
2. Install Git
(robotics102.github.io/tutorials/setup)
3. Accept P0 assignment
(robotics102.github.io/projects/a0)

1. Make a folder where you will store your code for ROB 102
2. Open a terminal
3. Use the `cd` command go to the folder you created.
4. Use the `git clone` command to clone your code.

We learned:

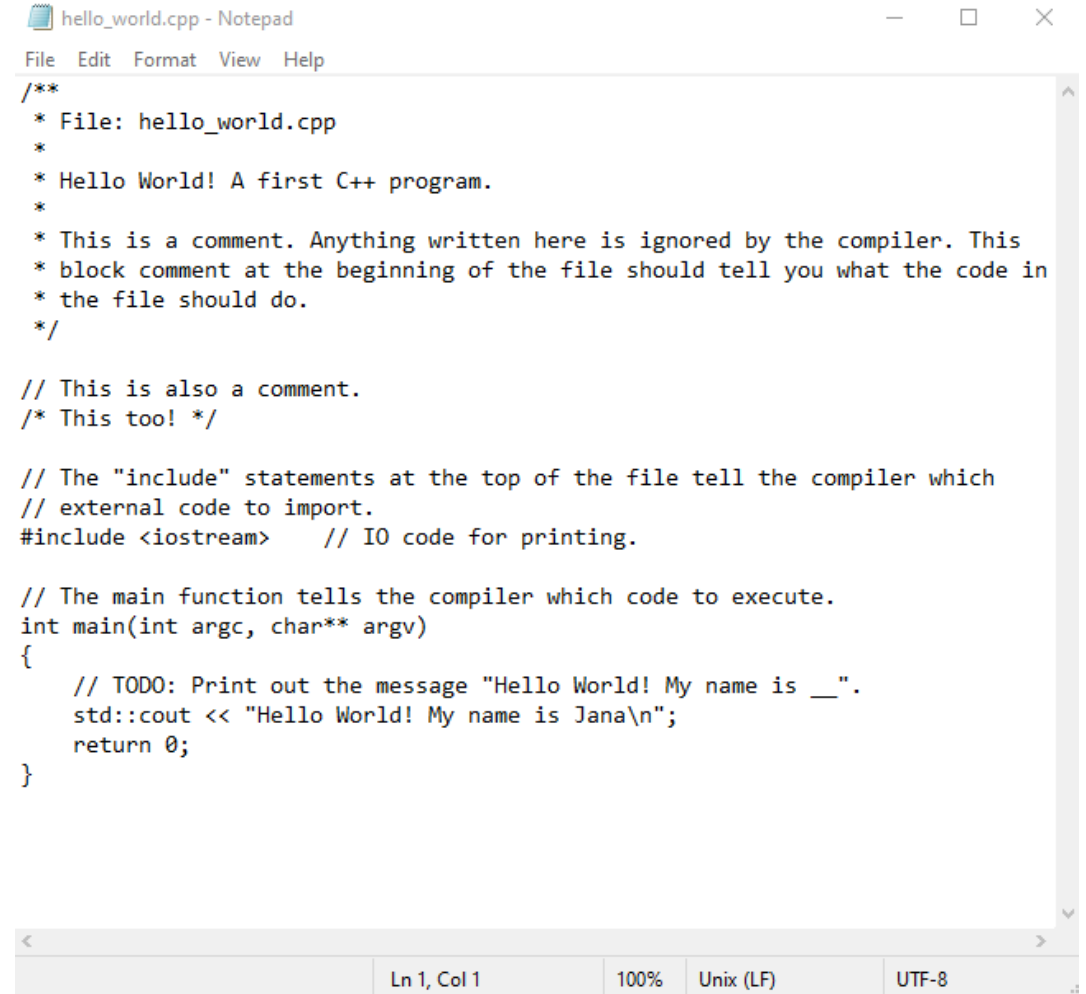
- ✓ How to change directories in a terminal with `cd`
- ✓ How to list files in a directory with `ls`
- ✓ How to clone a Git repository with `git clone`

Editing your code

We will need some way to view and edit the code files in the repository.

Idea #1: Notepad

This would work, but there's a much better way!



```
hello_world.cpp - Notepad
File Edit Format View Help
/**
 * File: hello_world.cpp
 *
 * Hello World! A first C++ program.
 *
 * This is a comment. Anything written here is ignored by the compiler. This
 * block comment at the beginning of the file should tell you what the code in
 * the file should do.
 */

// This is also a comment.
/* This too! */

// The "include" statements at the top of the file tell the compiler which
// external code to import.
#include <iostream>    // IO code for printing.

// The main function tells the compiler which code to execute.
int main(int argc, char** argv)
{
    // TODO: Print out the message "Hello World! My name is __".
    std::cout << "Hello World! My name is Jana\n";
    return 0;
}

Ln 1, Col 1    100%    Unix (LF)    UTF-8
```

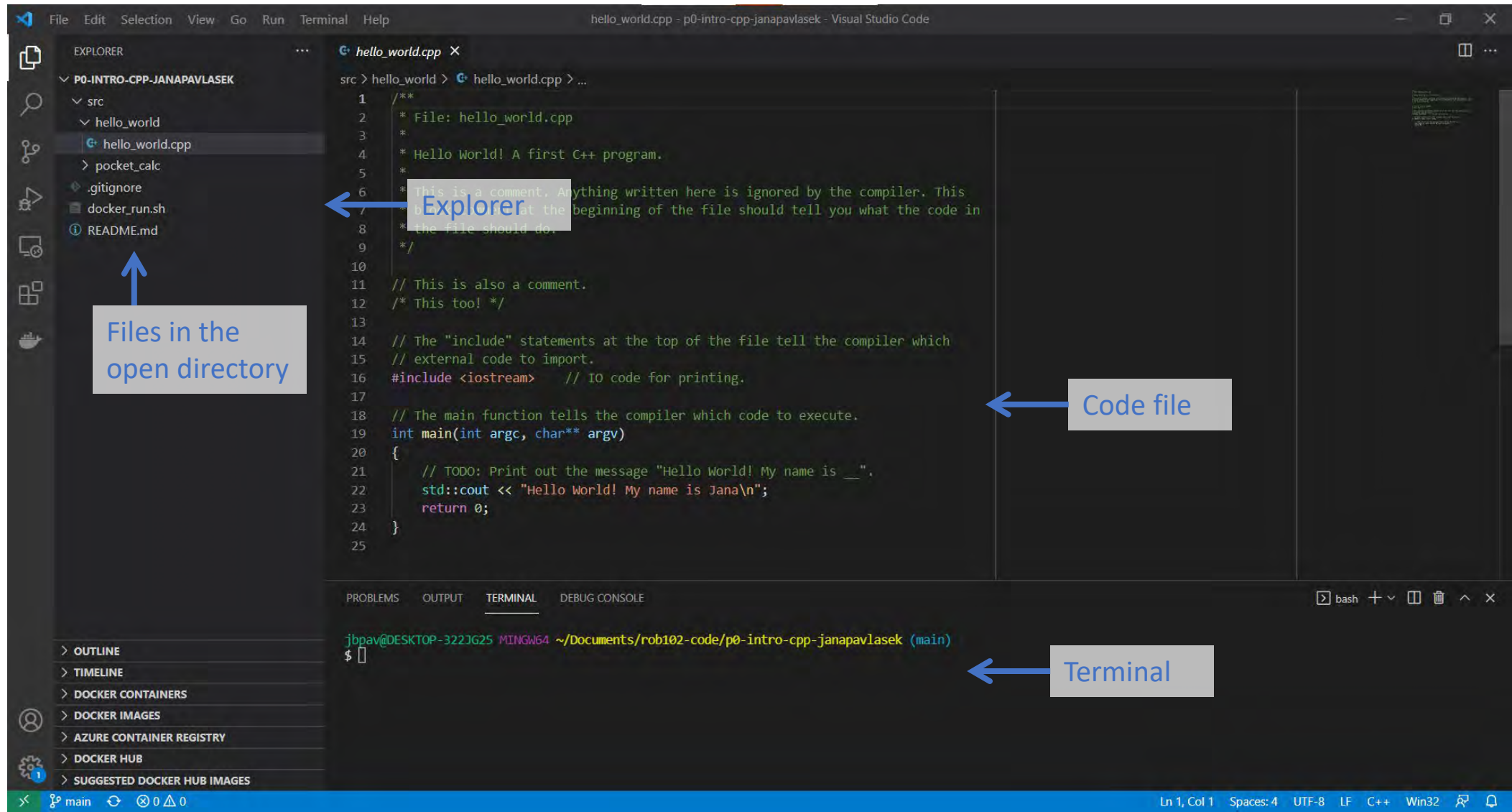

Visual Studio Code

VSCode (short for Visual Studio Code) is an Integrated Development Environment (IDE).

It has many features, like a built-in terminal, syntax highlighting, Git diff highlighting (shows you what code changed since you last committed), and much more!

Plus, you can install extensions to get even more features.

VSCode



VSCode: Your turn!

1. If you haven't already, install VSCode
 - <https://robotics102.github.io/tutorials/setup>
2. Install the C++ extension
3. Open your repository in VSCode

Careful: There is another IDE called “Visual Studio.” We are not using that one. If you're searching for information on VSCode, make sure you aren't looking at Visual Studio.

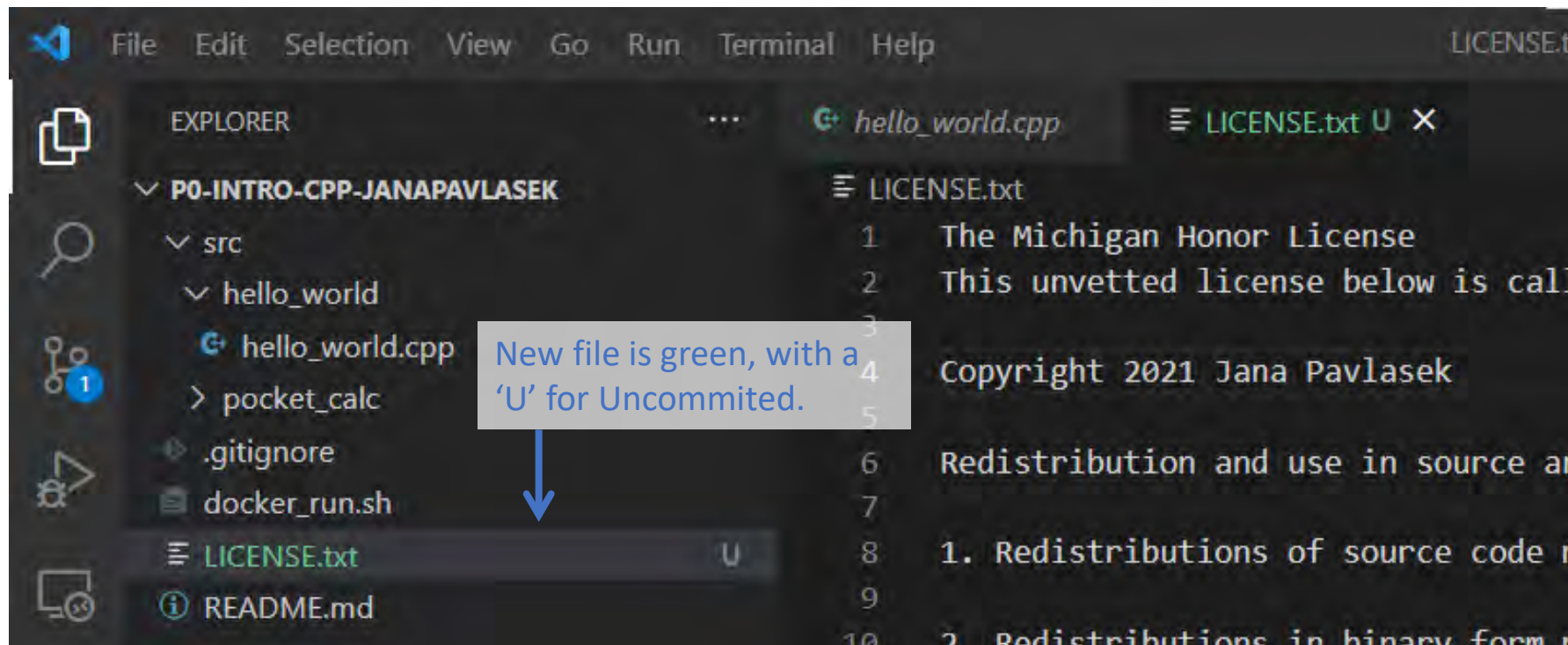
Adding the License

Now that we can edit code, let's make a change to the repository.

1. In VSCode, make a new file (File > New File)
2. Go to autorob.org/MichiganHonorLicense.txt
3. Copy the contents into the new file
4. Replace <YEAR> with 2021 and <COPYRIGHT HOLDER> with your name
5. Save the file with the name "LICENSE.txt"

Adding the License

Now that we can edit code, let's make a change to the repository.



Git Workflow: Pushing Changes

Your computer

The cloud

The folder on your computer
where your code is stored

`~/username/my_repo` *OR*
`C:\Users\username\my_repo`

workspace

Your Git repository in the cloud
`https://github.com/gituname/my_repo`

remote
repository

Our workspace has changes which are
not synced to the remote repository!

No LICENSE.txt →

robotics102 / p0-intro-cpp-janapavleseck Private Unwatch 3 Star 0 Fork 0
generated from robotics102/intro-cpp

<> Code Issues Pull requests Actions Projects Security Insights

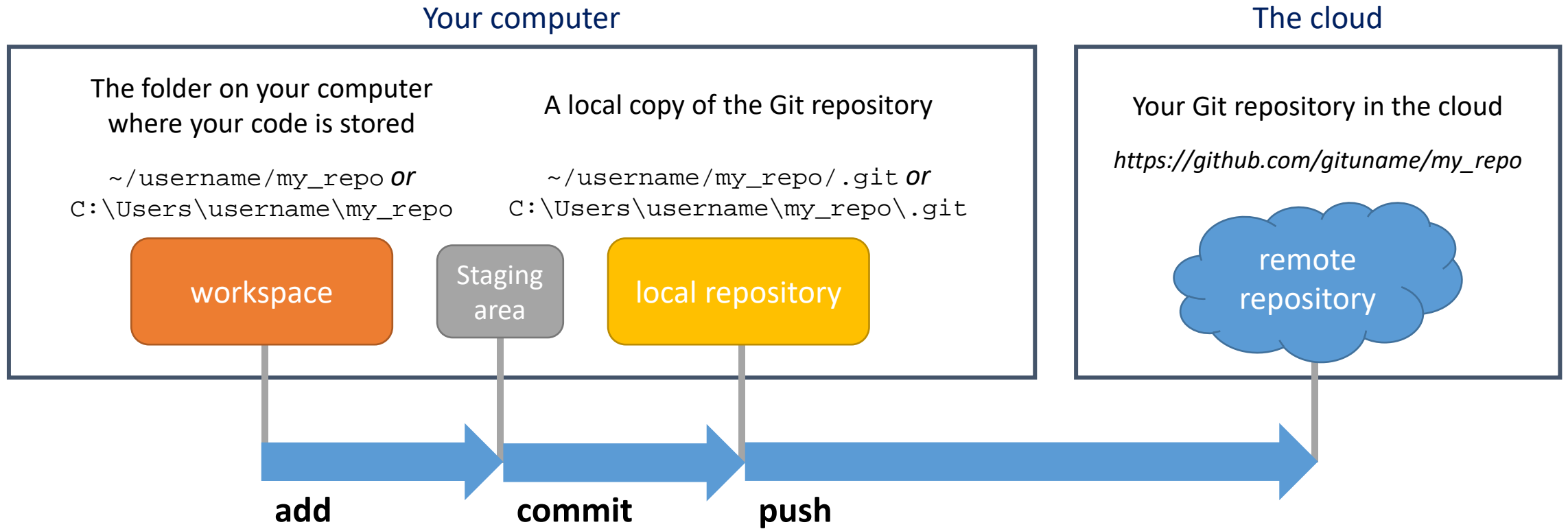
main

Go to file Add file Code About

janapavleseck	Complete Hello World activity	yesterday	2
src	Complete Hello World activity	yesterday	
.gitignore	Initial commit	4 days ago	
README.md	Initial commit	4 days ago	
docker_run.sh	Initial commit	4 days ago	

Releases 1 tags

Git Workflow: Pushing Changes



We need to **commit** our changes to our local repository and then **push** them to the remote repository.

Git Workflow: Pushing Changes

Open a terminal in VSCode and type:

```
git status
```

Untracked means
this is a **new** file

Shows which files
have been added
or changed

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE

jbpav@DESKTOP-322JG25 MINGW64 ~/Documents/rob102-code/p0-intro-cpp-janapavlasek (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
(use "git add <file>..." to include in what will be committed)
  LICENSE.txt

nothing added to commit but untracked files present (use "git add" to track)

jbpav@DESKTOP-322JG25 MINGW64 ~/Documents/rob102-code/p0-intro-cpp-janapavlasek (main)
$
```

Notice: VSCode opens a
terminal in your open
code folder by default.

Git Workflow: Staging Changes

In your VSCode terminal, type:

Argument: name of file to add.



```
git add LICENSE.txt
```

If you see this warning, ignore it!

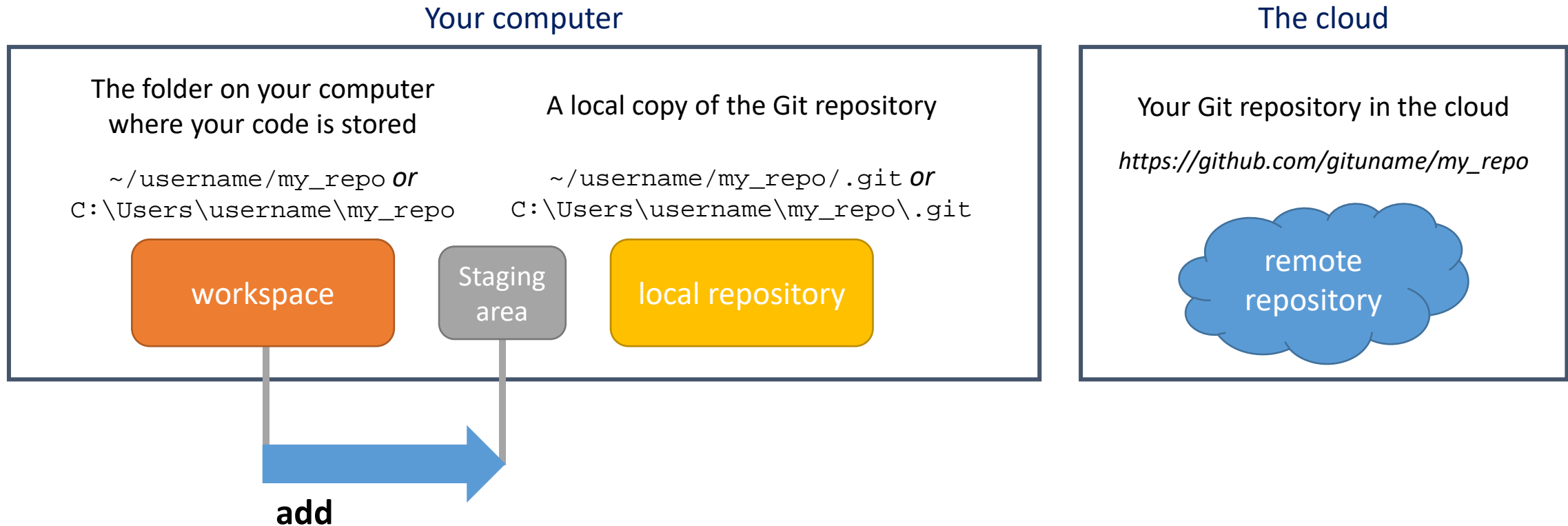
```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE
jbpav@DESKTOP-322JG25 MINGW64 ~/Documents/rob102-code/p0-intro-cpp-janapavlsek (main)
$ git add LICENSE.txt
warning: CRLF will be replaced by LF in LICENSE.txt.
The file will have its original line endings in your working directory

jbpav@DESKTOP-322JG25 MINGW64 ~/Documents/rob102-code/p0-intro-cpp-janapavlsek (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
   new file:   LICENSE.txt
```

Type git status again to see the file ready to be committed

Git Workflow: Staging Changes



Now our change is **staged** and ready to be committed!

Git Workflow: Committing Changes

In your VSCode terminal, type:

```
git commit -m "Add license"
```

Argument: commit message.
Make the message descriptive!



Commit added to
the "main" branch



```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE

jbpav@DESKTOP-322JG25 MINGW64 ~/Documents/rob102-code/p0-intro-cpp-janapavlasek (main)
$ git commit -m "Add license"
[main 7932e1a] Add license
1 file changed, 20 insertions(+)
create mode 100644 LICENSE.txt

jbpav@DESKTOP-322JG25 MINGW64 ~/Documents/rob102-code/p0-intro-cpp-janapavlasek (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
(use "git push" to publish your local commits)

nothing to commit, working tree clean
```

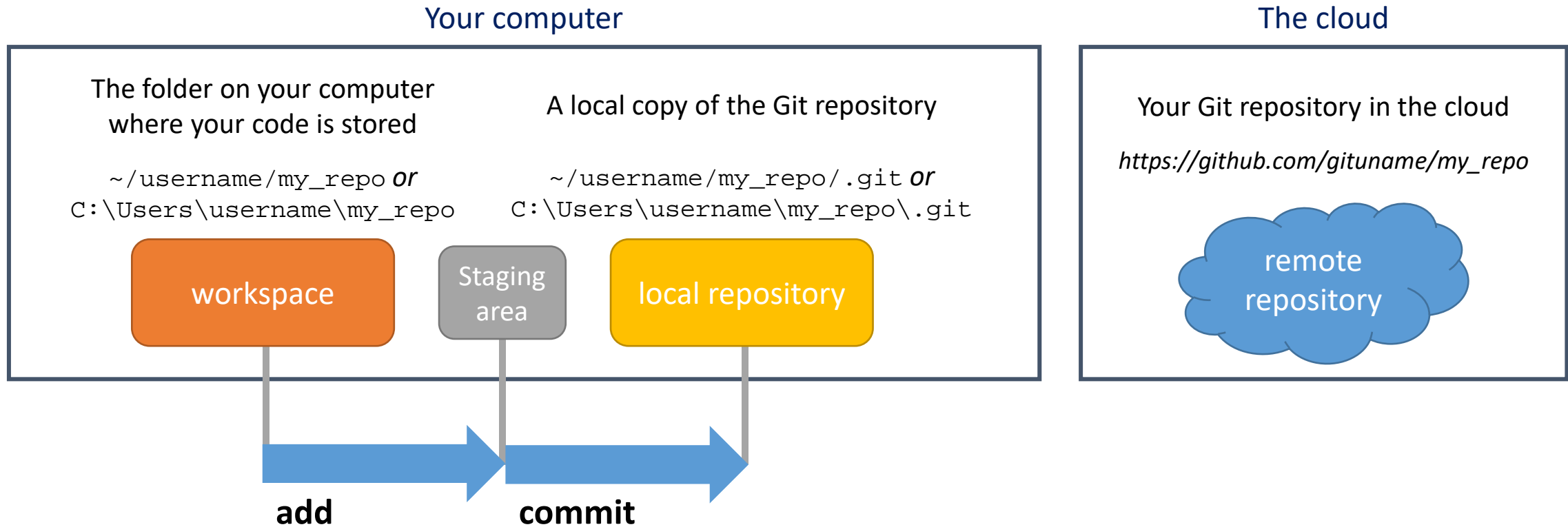
We have one unpushed
commit!



No more uncommitted
changes in our workspace



Git Workflow: Committing Changes



Now our change is **committed** and ready to be **pushed**!

Git Workflow: Pushing Changes

In your VSCode terminal, type:

```
git push
```

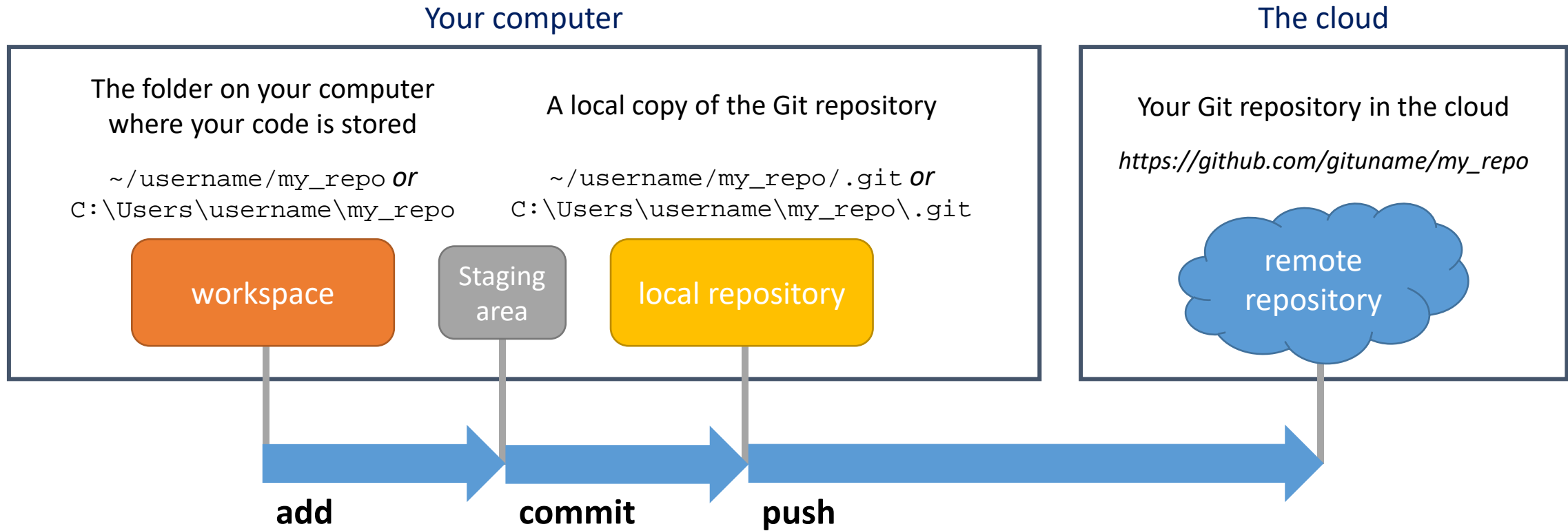
```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE

jbpav@DESKTOP-322JG25 MINGW64 ~/Documents/rob102-code/p0-intro-cpp-janapavlsek (main)
$ git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 1.30 KiB | 1.30 MiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/robotics102/p0-intro-cpp-janapavlsek.git
 cff208f..7932e1a  main -> main
```



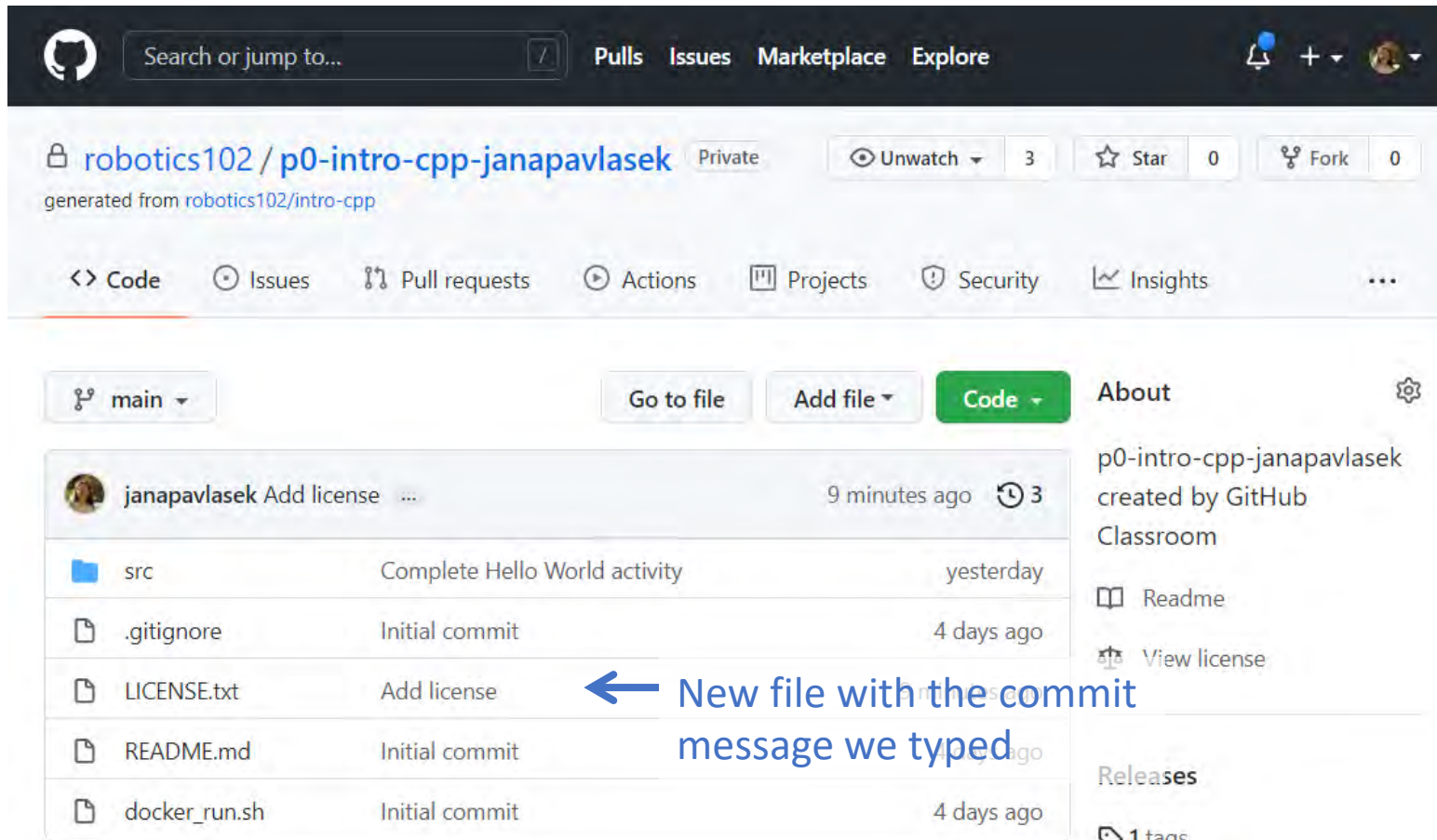
Commit pushed to “main” branch!

Git Workflow: Pushing Changes



Now our license file is synced to the remote repository in the cloud!

Git Workflow: Pushing Changes

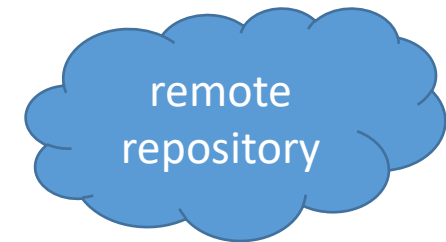


The screenshot shows a GitHub repository page for 'robotics102 / p0-intro-cpp-janapavleseck'. The repository is private and has 3 forks. The commit history shows a commit by 'janapavleseck' 9 minutes ago with 3 files changed. The files listed are 'src', '.gitignore', 'LICENSE.txt', 'README.md', and 'docker_run.sh'. The 'LICENSE.txt' file is highlighted with a blue arrow and the text 'New file with the commit message we typed'.

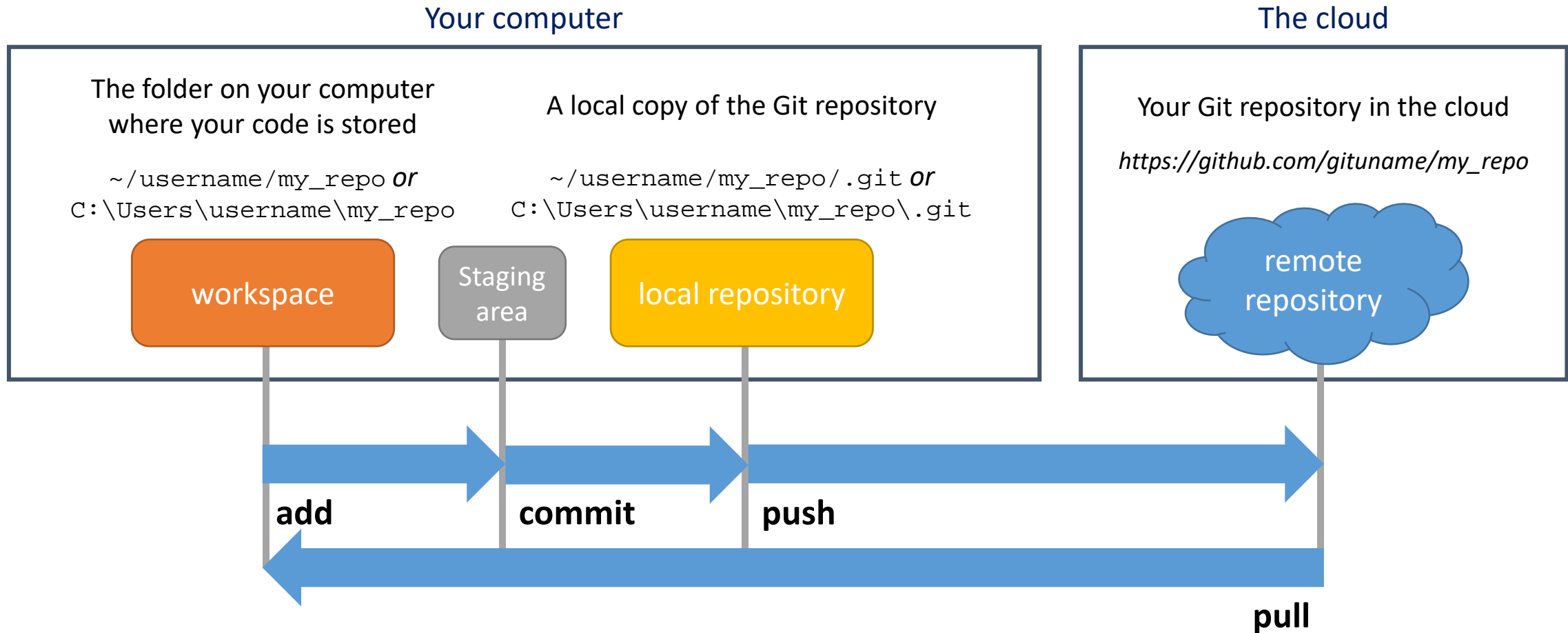
File	Commit Message	Time
src	Complete Hello World activity	yesterday
.gitignore	Initial commit	4 days ago
LICENSE.txt	Add license	9 minutes ago
README.md	Initial commit	4 days ago
docker_run.sh	Initial commit	4 days ago

The cloud

Your Git repository in the cloud
https://github.com/gituname/my_repo



Git Workflow: Pulling Changes



If someone else (the instructor, a teammate, you on a different computer...) has pushed changes to your repository, you can **pull** them using the command `git pull`

Running your code: Docker

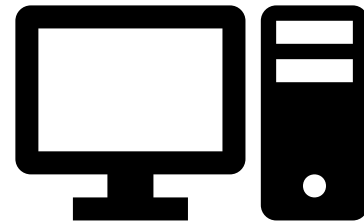
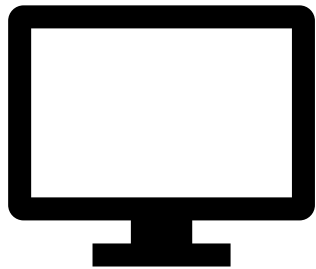
In ROB 102, we will use a program called **Docker** to run our code.

Docker is a program that lets us run **containers**. Containers are environments that imitate a computer.

Docker sets up a container based off an **image**, which is a snapshot of the state of a computer at a given point. Ours will use the **Linux** operating system.

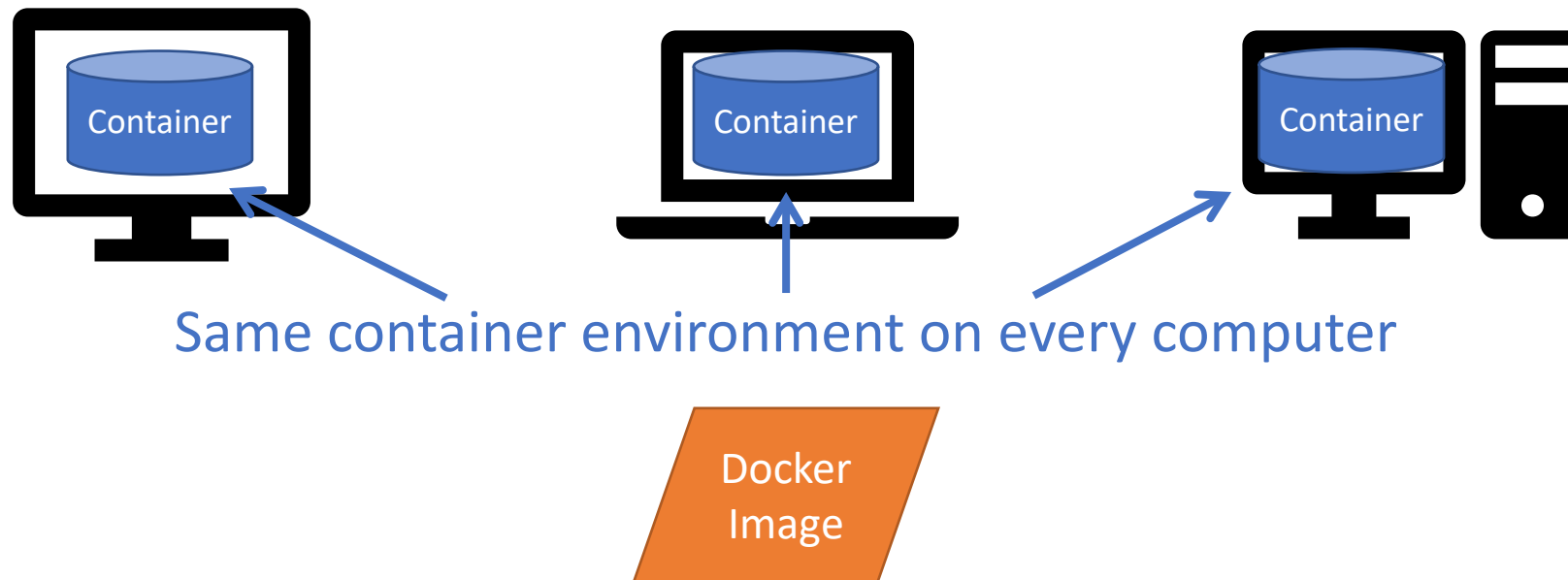
Running your code: Docker

We can use a Docker image to run containers that look **exactly the same on every computer!** It contains *all the dependencies* we need to get started running our code.



Running your code: Docker

We can use a Docker image to run containers that look **exactly the same on every computer!** It contains *all the dependencies* we need to get started running our code.



Docker: Your turn!

If you haven't already, install Docker

- <https://robotics102.github.io/tutorials/setup>

Also install the Docker and Docker Explorer VSCode extensions.

Docker: Your turn!

In your VSCode terminal, type:

```
./docker_run.sh
```

The screenshot shows the VSCode interface with the following elements:

- DOCKER CONTAINERS:** A list of containers, including 'rob102-cpp-intro (gcc:8.4)' which is highlighted with a green icon. A blue arrow points to this entry with the text 'List of docker containers (green = running)'.
- DOCKER IMAGES:** A list of images, including 'gcc:8.4' which is highlighted with a blue icon. A blue arrow points to this entry with the text 'List of docker images downloaded'.
- TERMINAL:** A terminal window showing the execution of the command `./docker_run.sh`. The output indicates that the image 'gcc:8.4' was not found locally and was pulled from the library. The terminal prompt changes from `root@f44831118d31:/#` to `root@f44831118d31:/#`, indicating the terminal is now inside the container. A blue arrow points to the terminal prompt with the text 'This terminal is inside the Docker container!'.
- PROBLEMS, OUTPUT, DEBUG CONSOLE:** These panels are visible at the top of the terminal area.

Additional annotations in the terminal area:

- A blue arrow points to the output line '8.4: Pulling from library/gcc' with the text 'Yours will look different (it will download the image)'.

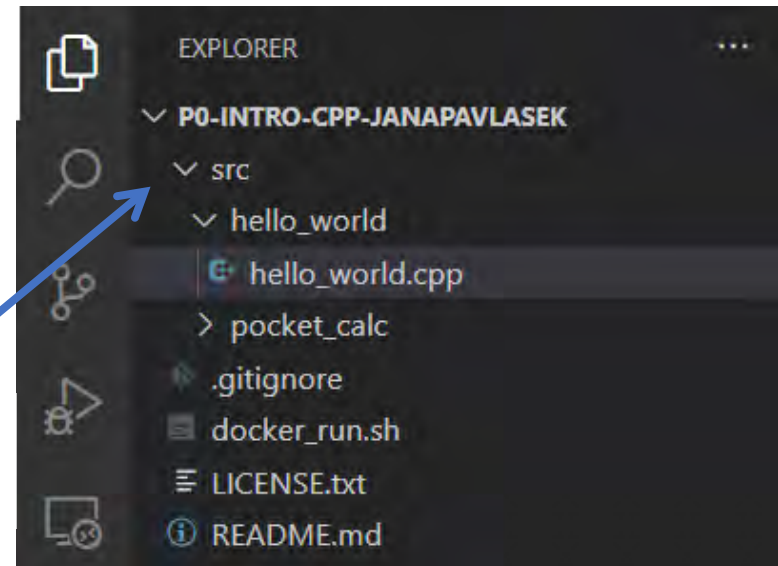
Docker: Exploring the container

The `src` folder in the repository is **mounted** inside the container.

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE
root@f44831118d31:/# cd code
root@f44831118d31:/code# ls
src
root@f44831118d31:/code# ls src/hello_world/
hello_world.cpp
root@f44831118d31:/code#
```

Change to code directory

`/code/src/ mirrors C:\Users\jana\p0-intro-cpp-janapavleseck/src`



You will write code in VSCode, and compile and run it in the container.

Docker: Running your code

Edit the file `src/hello_world/hello_world.cpp` to print the message “Hello World!”

```
cd /code/src/hello_world
```

In the Docker container, change directories to “`src/hello_world`”.
Type the command:

```
g++ hello_world.cpp -o hello_world
```

↑
The C++ compiler
we will be using

↑
The C++ source file
to compile

↑
The name we want to
give our executable

Docker: Running your code

In the Docker container (still in the `hello_world` directory), type the command:

```
./hello_world
```

Our executable is created by the compiler

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE

root@f44831118d31:/# cd /code/src/hello_world/
root@f44831118d31:/code/src/hello_world# g++ hello_world.cpp -o hello_world
root@f44831118d31:/code/src/hello_world# ls
hello_world  hello_world.cpp
root@f44831118d31:/code/src/hello_world# ./hello_world
Hello World! My name is Jana
root@f44831118d31:/code/src/hello_world#
```

Compile the code

The code runs!

Command line cheat sheet

<code>cd my_directory/</code>	<p>Changes the working directory to <code>my_directory</code>.</p> <ul style="list-style-type: none">• If we write <code>/my_directory/</code> with a slash at the beginning, it is interpreted as an absolute path. Without the slash, the path is relative to the current directory.• Typing just <code>cd</code> with no argument brings you to the home directory.
<code>ls</code>	<p>Lists all the files and folders in the current directory. Typing <code>ls my_dir/</code> lists the files in the directory <code>my_dir/</code>.</p>
<code>pwd</code>	<p>Prints the path of the current working directory.</p>
<code>./my_exec</code>	<p>Runs executable called “<code>my_exec</code>”.</p>
<code>g++ my_code.cpp -o my_exec</code>	<p>Compiles the code in the file <code>my_code.cpp</code> into an executable called <code>my_exec</code>.</p>
<code>exit</code>	<p>Closes the current terminal.</p>

Git cheat sheet

<code>git clone ADDRESS</code>	Clone the repository located at ADDRESS to the current folder.
<code>git status</code>	Show the status of the Git repository (untracked files, modified files, current branch).
<code>git add FILE</code>	Add FILE for staging (next step: commit).
<code>git commit -m "My commit msg"</code>	Commit the files currently added for staging with the given commit message (next step: push).
<code>git push</code>	Push the commits created to the remote repository.
<code>git pull</code>	Pull any new commits in the remote repository into the local repository.
<code>git tag p0-v1</code>	Tag all commits currently on this branch with tag p0-v1.
<code>git push origin p0-v1</code>	Push tag p0-v1 to the remote repository.

Tasks for Today

- ✓ Install all software
- ✓ Push license to GitHub repo
- ✓ Run Hello World on your computer
- ✓ Push the changes to your repository