# Git basics

Stats 21

Miles Chen

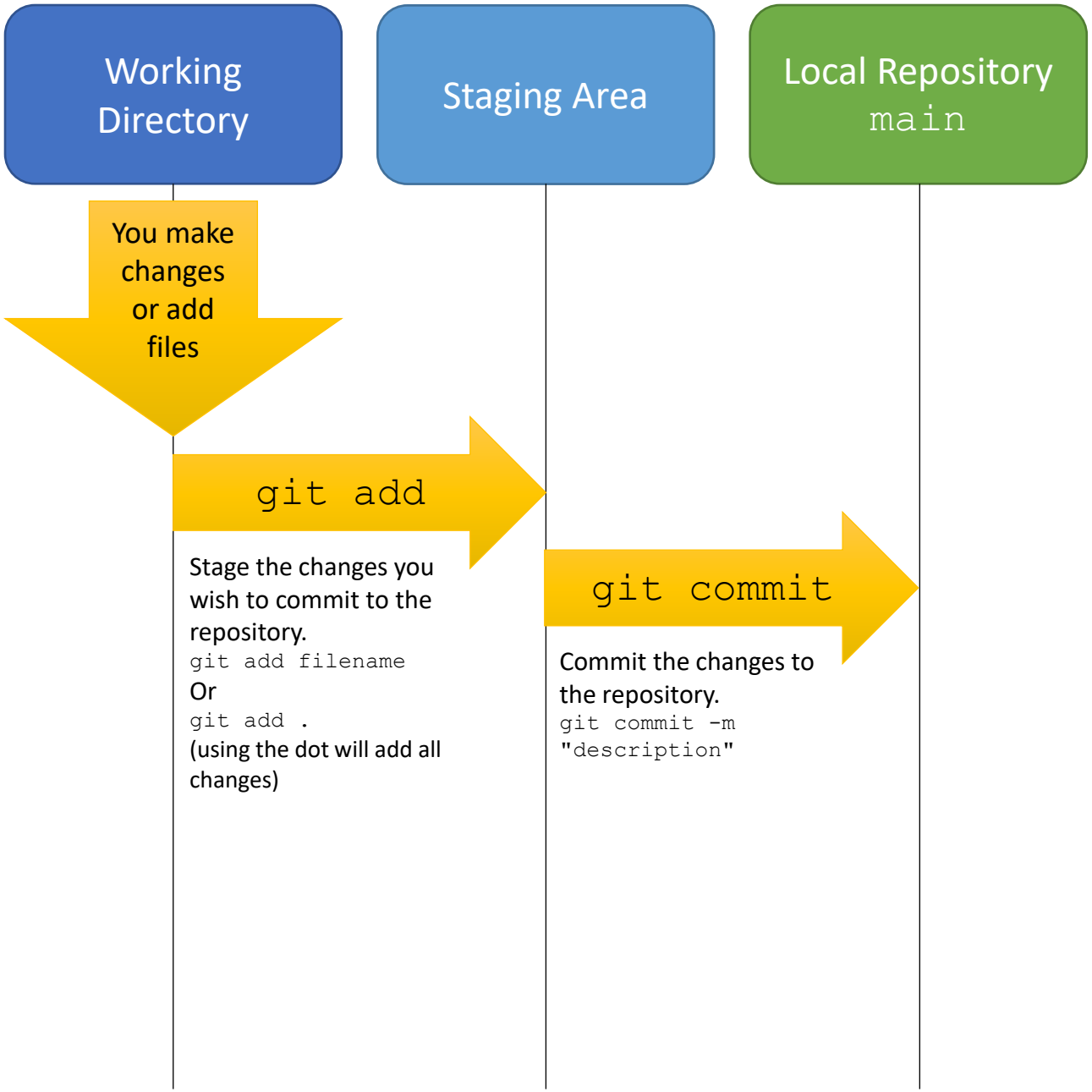
# What is Git

- Git is a version control system.
- All changes committed to the repository are tracked. You can always roll back a file to an older version.
- It allows for collaboration and will synchronize work done by multiple people.

# The very basics

- Most important commands:
- `git status`
  - This tells you the status of the repository. I recommend running this command very frequently. Run it before and after every add, every commit, every fetch, every merge until you have a strong understanding of what git will do.
- `git add filename.txt`
  - This command add filename.txt to the staging area
- `git add .`
  - Will add all changes in the working directory to the staging area
- `git commit –m "description of the commit"`
  - This command commits the changes in the staging area to the repository. The current state of the repository is now forever tracked by git

# Making and committing changes to a local repository



# Why do I have to git add and git commit?

- Commits to a repository should be logical and small changes that can be easily understood. Our workflow, however, does not always reflect this.
- If you've been working on several files and making many different changes, you can break up the changes by using git add for just one (or a few files) at a time and committing those changes.
- Changes are tracked in the git log

# Restoring a previous version of a file

- `git log`
  - Lists all of the commits made to the repository. This list can be long. You can use space to go to the next page of the log.
  - To exit the log, type `q`
  - Identify the commit that has the version of the file you want. Again, the commits you make to the repository should have small, logical changes with descriptive messages so that it is easy to identify which commit has the version you like.
- `git checkout ab123ef filename.txt`
  - Replace `ab123ef` with the first few letters of the hash associated with the commit

# Reading changes made in a commit

- `git show 123abc`
  - This command will show the contents of the commit with SHA1 hash 123abc.
  - git will show information about the author, date, and comment of the commit.
  - It will then show the actual line changes:
    - `--- a/filename.txt` (file before the change)
      `+++ b/filename.txt` (file after the change)
    - `@@ -23,4 +23,5 @@` (the changes begin at line 23. The output shows 4 lines from the file before the change, after the change, you can now see five lines from the file)

```
  Lines from the file with no changes appear in white
- This Line is preceded by a minus sign. It has been removed from the file
+ This line is preceded by a plus sign. It was added to the file.
```

# Using GitHub

# GitHub

- GitHub is a company that hosts remote repositories using Git.
- Other companies offer similar services, such as GitLab and Bitbucket but GitHub remains the most popular. Many open-source projects are hosted on GitHub and uses GitHub as a central location for collaboration.
- It is not required to use GitHub to host remote repositories, but GitHub makes it easy.
- While it is possible to create a repository on your local machine first and then push it to GitHub, I recommend setting up the repository on GitHub first and then cloning it to your local machine.

# Getting Set Up

- First, create an account on GitHub.com
- Create an SSH key on your local machine.
- Add this SSH key to your GitHub account. This lets GitHub know that it can trust your machine to make changes to the repositories on your account.
- Create a repository on GitHub and then clone the repository to your machine.

# Creating an SSH Key

- https://docs.github.com/en/github/authenticating-to-github/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent
- Run the following command in your Terminal or Git Bash using the email for your GitHub account. This will create a public and private key pair.
- `$ ssh-keygen -t ed25519 -C "your_email@example.com"`
- If you are the only one who uses your computer, I recommend **not** creating a passphrase for your SSH key.
- Your SSH key will be saved in the default location `~/.ssh`
- Copy the contents of the public key and add it to your GitHub account.

# Adding an SSH Key to GitHub

- Locate the SSH public key on your computer. Probably in
  - C:\Users\yourname\.ssh on windows
  - /Users/yourname/.ssh on Mac
- Copy the contents of the key to the clipboard.
  - Open with a text editor and copy
  - terminal: clip < id_ed25519.pub
- Log into you GitHub account.
- Go to Settings > SSH and GPG keys: https://github.com/settings/keys
- Click “New SSH Key”
- For Title, use what you call your computer, e.g “Mac Laptop” or “Home PC”
- For Key, paste the contents of the public SSH key.

# Cloning your repository from GitHub to your machine

Step 0. Prerequisites:

- Create GitHub account
- SSH key created on your local computer and added to GitHub account.

Step 1. Create a new repository `'my_repo'` on GitHub

Your Remote Repository on GitHub `origin/main`

`git clone`

Step 2. Clone the repository to your local machine. This will create a folder `'my_repo'` on your computer.

```
git clone git@github.com:your_username/my_repo.git
```

Your Local Computer. A new folder is created:

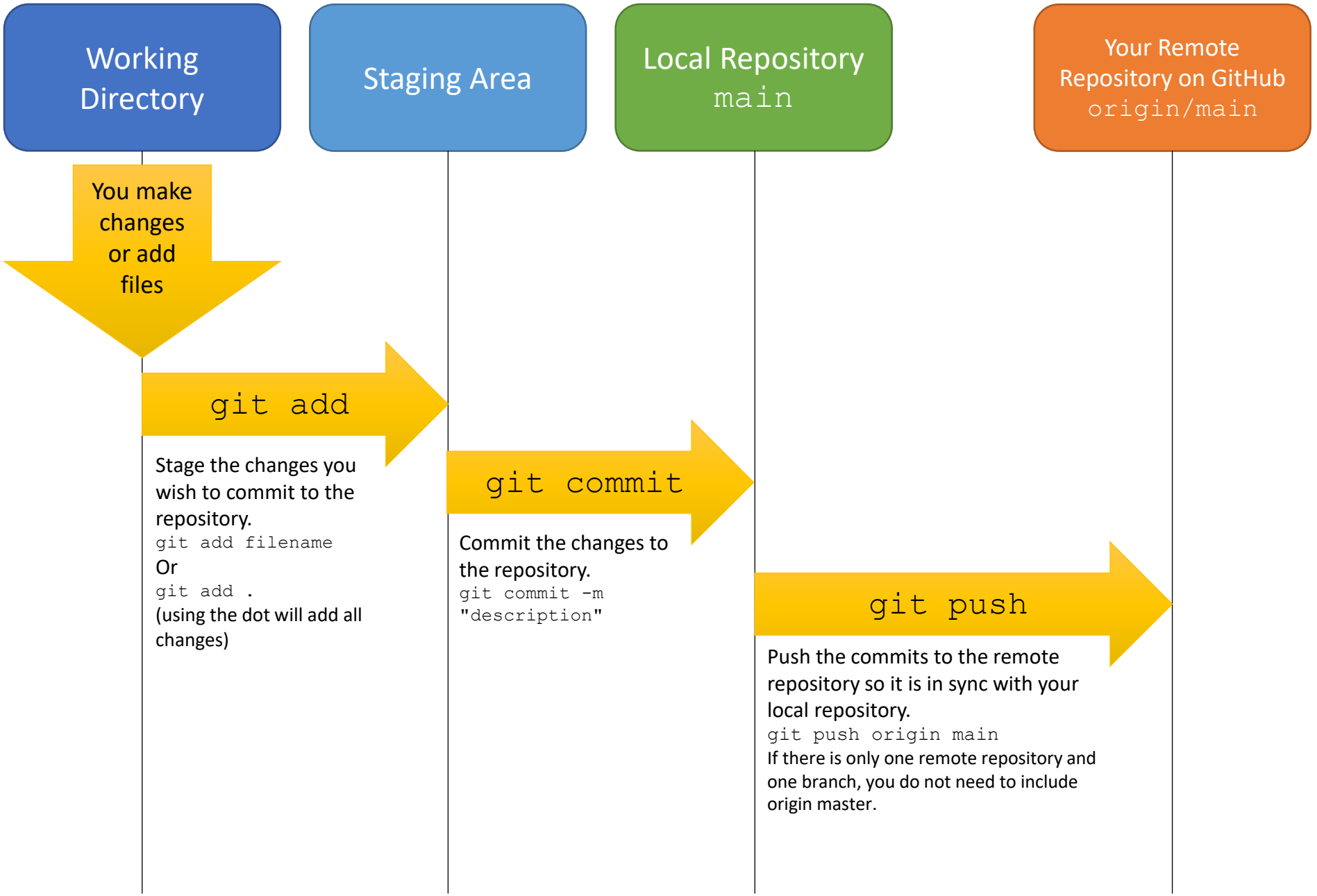
`C:\users\yourname\my_repo` or `/Volumes/Mac/Users/yourname/my_repo`

Working Directory

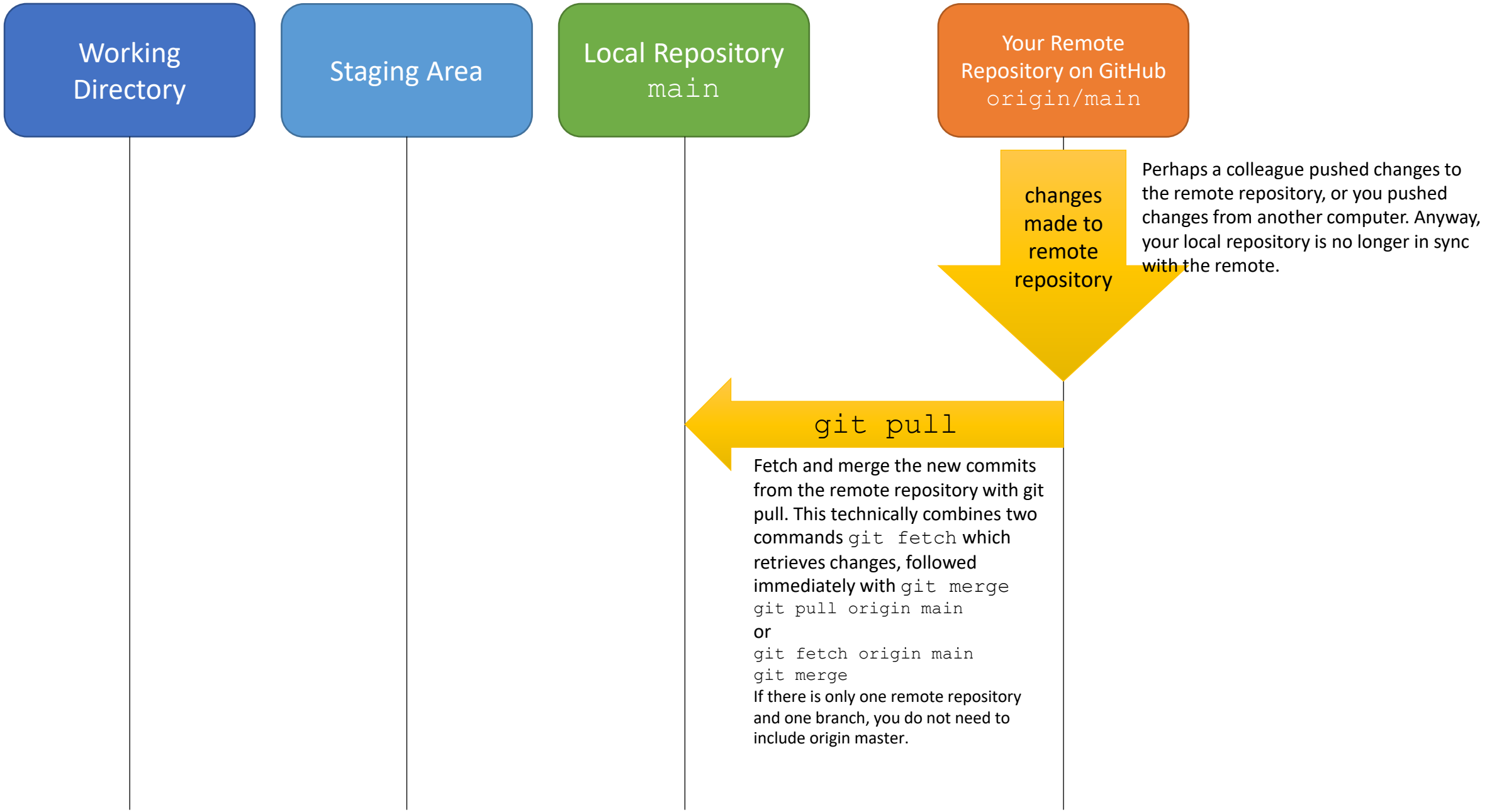
Staging Area

Local Repository `main`

# Pushing changes from your local computer to GitHub



# Pulling changes from your GitHub Repo to your local computer



# Repository for this class

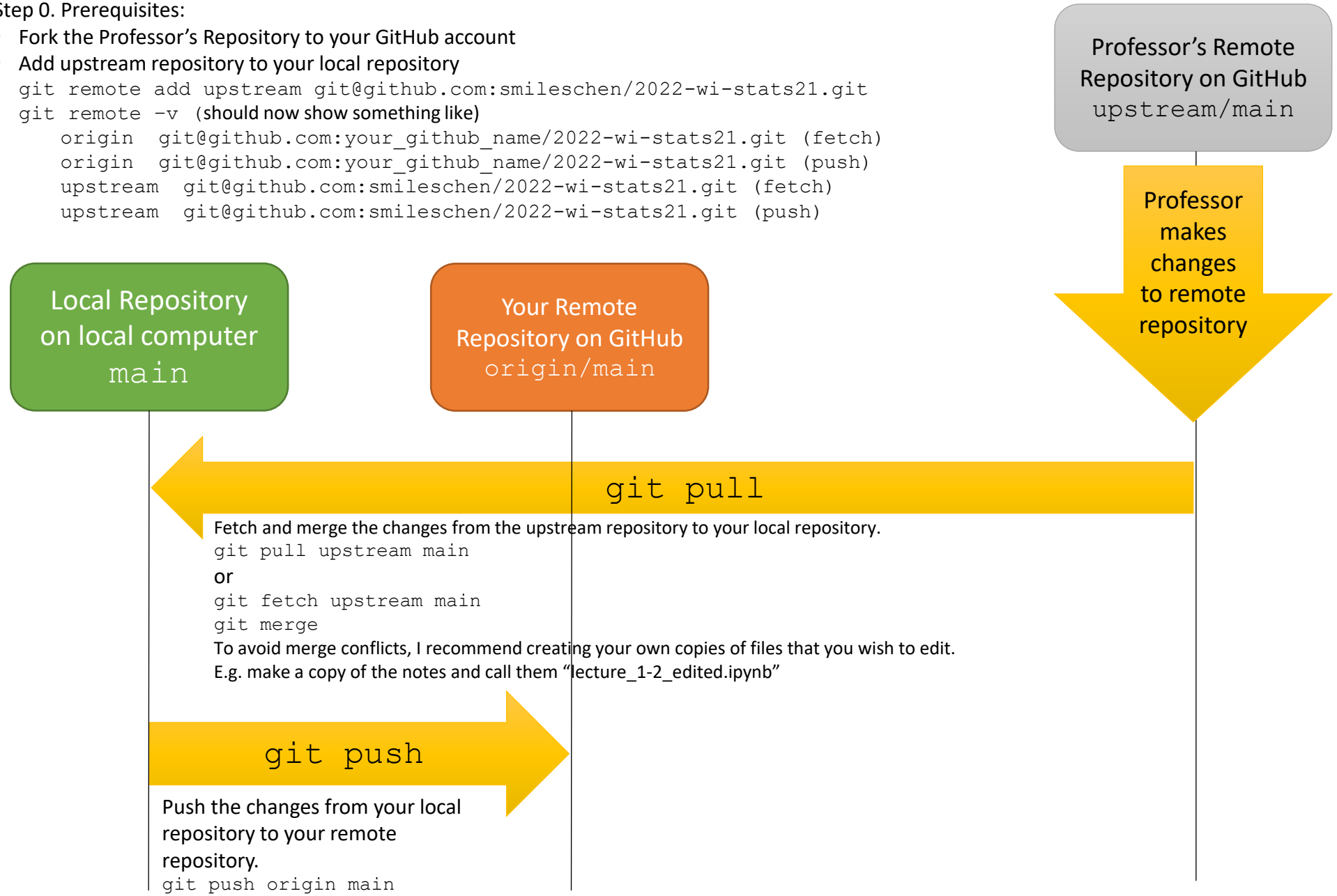
- https://github.com/smileschen/2022-wi-stats21
- Fork this repository to your GitHub account.
- Clone the repository to your local machine.
- Add the professor's repository as an upstream remote repository.
- `git pull` changes that I (the professor) make to the repository.

# Pulling changes from Professor's GitHub repo

Step 0. Prerequisites:

- Fork the Professor's Repository to your GitHub account
- Add upstream repository to your local repository
- `git remote add upstream git@github.com:smileschen/2022-wi-stats21.git`
- `git remote -v` (should now show something like)

```
origin  git@github.com:your_github_name/2022-wi-stats21.git (fetch)
origin  git@github.com:your_github_name/2022-wi-stats21.git (push)
upstream  git@github.com:smileschen/2022-wi-stats21.git (fetch)
upstream  git@github.com:smileschen/2022-wi-stats21.git (push)
```



# Reading changes made in a commit on GitHub

- GitHub has some very nice features.
- One is the log of commits made to a repository.
- You can click each commit and view the exact changes made.