# Weekly Assignment – Using Git with GitHub

# Objectives:

· Understand basic git commands

· Clone a GitHub repository

· Update files within a repository

· Push changes to a remote GitHub repository

# [Git Command Cheat Sheet](https://education.github.com/git-cheat-sheet-education.pdf)

# Step 1: Setup GitHub Account

You will need a [GitHub](https://github.com/) account for use in this course. We will be using Git and GitHub to track and grade assignments for this course. Sign up for a new GitHub account with the following GitHub username:

Your Madison College username (everything before the ‘@’ in your Madison College email address) plus ‘-matc’. For example, if your Madison college email address is jsmith@madisoncollege.edu then your GitHub username would be ***jsmith-matc***

# Step 2: Log into NetLab

We will be using NetLab for this semester’s activities, labs, and projects. You can access NetLab by going to [http://netlab2.madisoncollege.edu](http://netlab2.madisoncollege.edu/) from any web browser. You will sign in with your Madison College username (***without***the @madisoncollege.edu).

If you ***have never*** used NetLab, your initial password will be ***Python123***. You will be required to change your password on first login. Please be sure to note this password.

If you ***have***used NetLab in another Madison College course, your credentials will remain the same. If you don't remember your password, I can reset it for you.

We will be logging into the Kali Linux machine for the majority of our course exercises. The account to log into this system is…

Username: kali

Password: kali

# Step 3: Setup your WeeklyAssignment Repository

This is the repository you will use for the remainder of the semester to turn in assignments.

1. From the Kali Linux VM in NetLab, open a terminal.

2. Navigate to a directory where you want to store the repository files for this course. I recommend doing this in the student ‘Documents’ or ‘Desktop’ folder. ***cd Documents***

3. Using a web browser on the Kali VM in NetLab, follow this invitation link to create your course repository:

Cathie Malin’s Class: <https://classroom.github.com/a/JQ3FeELD>

Mike Masino’s Class: <https://classroom.github.com/a/dqwkLXbr>

Mike Belton’s Class: <https://classroom.github.com/a/kjDt67UW>

4. Once the repository is created, navigate to the repository in GitHub.

5. Select the URL and copy to your clipboard…

6. From the terminal window in your Kali VM, run the git clone command to link the GitHub repository to your local workspace…

***git clone <URL copied in the previous step>***

Note you will get a warning that you cloned an empty repository. This is expected!

7. Change into the newly created repository directory. It should look something like w***eeklyassignments-<YourGitHubUserName>***

# Step 4: Setup Your Git Configuration

Git configuration is needed to log who is making changes to a given repository. In your local environment this may feel unnecessary since you are likely the only person making changes. But when you start collaborating and contributing changes to a centralized repository like GitHub, it is crucial to know who made changes and how to contact them.

1. To set up your git your config run the following commands:

***git config --global user.email “youremail@madisoncollege.edu”***

***git config --global user.name “Your Name”***

# Step 5: Working locally with GIT

1. Start by using ***git status*** to see the current state of your local repository. This command would show you all of the files that are not staged. At this point there are no files in staging.

2. Then check ***git branch –a*** to see which branches exist and which branch is your current working directory. Currently there are no branches.

3. Add a new text document to this folder and save the changes.

***nano githubexercise.txt***

4. User ***git status*** to see the changes

5. You can see that the new file is ‘Untracked’ meaning Git commands don’t yet know about it. To add your new file to be tracked by git, run ***git add githubexercise.txt***

*6.* Run ***git status*** to see what happened after a *git add command*

7. Your file is now staged meaning when you do the next commit, it will be included in the check point. Commit them to your local repository by running ***git commit –m “Initial commit of GitHub exercise”***

*8.* Now check ***git status*** to see the status of your repository. You will see there is nothing to commit because all changes have already been committed.

# Step 6: Pushing changes to your GitHub Repository

1. First, check if any remote repositories are defined for you to use by running ***git remote****.* You should see that the default remote called ‘origin’ is setup to point to the GitHub repository you cloned from.

*2.* \*\*OPTIONAL\*\* You can define a new a remote repository link to any remote repository.

*a.* Run ***git remote add <name\_of\_remote> <url\_of\_remote>***. I try to name my remote GitHub or something to tell me where the remote is pointing

*b.* Once the remote is defined, run ***git remote*** to see the remote you just defined

*3.* Push code from your local main branch to the remote main branch by running ***git push origin master***

*a.* Enter your GitHub credentials (if prompted)

**\*\*Important\*\*** GitHub is now changing the main branch from ‘master’ to ‘main’. If you do not have a master branch, try the command using ‘main’. You can list your branches using ***git branch -a***

4. Refresh the repository in GitHub to verify the new file is now there.

# Step 7: Make some changes and revert back

1. Run ***git log*** to see the changes you made

2. Add a new file to your repository called ‘filetorevert.txt’

3. Run ***git status*** to see the files that have changed

4. Run ***git add <filename>*** and ***git commit –m “Added a new file***” to commit this new file to your repository.

5. Run ***git log | tee gitlogbefore.txt*** to see the additional commit is logged in git.

6. Run ***git reset --hard <hashkey of earlier change> | tee gitreset.txt*** to revert your change. (i.e.: deletes the newly create file)

7. Run the following commands:

***git status***

***git log | tee gitlogafter.txt***

***ls > ls.txt***

8. Add the newly created files to your repository using the following commands

***git add .*** #Adds the files to staging

***git commit -m “READY FOR GRADING - Adding log files for assignment submission”***

***git push origin master***

**\*\*Important\*\*** GitHub is now changing the main branch from ‘master’ to ‘main’. If you do not have a master branch, try the command using ‘main’.

9. Refresh your repository in GitHub.com and ensure the following files were uploaded correctly:

a. ***githubexercise.txt***

b. ***gitlogbefore.txt***

c. ***gitreset.txt***

d. ***gitlogafter.txt***

# Submission:

Your instructor will grade this assignment using your GitHub repository so simply confirm the files listed in the previous step are in your GitHub repository. The URL should look something like…

<https://github.com/madisoncollege-it-programs/weeklyassignments-username>)

Please ensure your commit message includes ‘READY FOR GRADING’ so that your instructor knows you have completed your submission.