

First exercise with Git

Before proceeding, use the following instructions to set up your local assignment repository. This will be the place you work on your class assignments and it will be linked to two remote repositories on GitHub making a V shaped unidirectional work flow:

Download from this link: [1.16 The Course Workflow.pptx](#) for a diagram of this process.

1. **assignments upstream spring20** - You should be able to find this repository in our class organization on 2U's GitHub: https://github.prod.oc.2u.com/UCB-INFO-PYTHON/assignments_upstream_spring20. This is where we will post all class assignments. You have read access to this repository, and each week you will use a pull command to download the latest assignments to your own machine then push your submissions to your student repository (**recall the "V" shaped work flow**).
2. **Your student repository** - In this exercise you will make your own student remote repository. You should have write access to your student repository, but it will be only readable by you and your instructors. When you complete your homework each week, you will use a push command to upload your work to this repository.

Initial Setup

There are several ways that you can set up your local repository. We recommend the following procedure.

First create an empty repository in Github for your homework, you can do this through the github user interface.

Create a new repository

- This is done through add menu "+" on the top or the green NEW button on the upper right and select "New repository"

- Select Owner and the UCB-INFO-PYTHON organization
- Name it your Firstname_LastnameREPO so mine should be "**Gunnar_KleemannREPO**"

Create a new repository

A repository contains all the files for your project, including the revision history.

- Make the repository **private** with the radio button

Important: Do not add a readme file you need an empty repository

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner



UCB-INFO-PYTHON

Repository name

Great repository names are short and memorable. Need inspiration? How about **potential-potato**.

Description (optional)

☐ **Public**
Anyone can see this repository. You choose who can commit.

☒ **Private**
You choose who can see and commit to this repository.

☐ **Initialize this repository with a README**
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: **None**

Add a license: **None**



Create repository

When you look at your repository it should look **LIKE THIS (this is an empty repo)**

UCB-INFO-PYTHON / Gunnar_KleemannREPO Private

Unwatch 4 Star 0 Fork 0

<> Code

Issues 0

Pull requests 0

Projects 0

Wiki

Insights

Settings

Give access to the people you work with

You should give access to the collaborators and teams you need to work with.

Add teams and collaborators

Quick setup — if you've done this kind of thing before

Set up in Desktop

 or

HTTPS

SSH

https://github.com/UCB-INFO-PYTHON/Gunnar_KleemannREPO.git

We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# Gunnar_KleemannREPO" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/UCB-INFO-PYTHON/Gunnar_KleemannREPO.git
git push -u origin master
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/UCB-INFO-PYTHON/Gunnar_KleemannREPO.git
git push -u origin master
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Import code

ProTip! Use the URL for this page when adding GitHub as a remote.

Give the instructors read access

- In your new **private** repository, go to the settings tab, on the right and then select collaborators and teams on the left
- Give (only) instructors read access:

The screenshot shows the GitHub repository settings for 'UCB-INFO-PYTHON / Gunnar_KleemannREPO'. The repository is private. The 'Settings' tab is selected in the top navigation bar (marked with a red '1'). In the left sidebar, the 'Collaborators & teams' section is highlighted (marked with a red '2'). The 'Teams' section is active, showing a list of teams. A dropdown menu 'Choose a team' is open, displaying a list of teams. The 'Instructors' team is highlighted in blue (marked with a red '3').

- You should see this:

The screenshot shows the 'Teams' section in the GitHub repository settings. The 'Instructors' team is listed with 8 members and 'Read' access. A red '3' highlights the 'Instructors' team in the dropdown menu.

Clone the assignments directory on your system

You need to tell git that you will be pulling content (homeworks) onto your machine from assignments_upstream_spring20 repository and pushing modified content (completed homeworks) to YourNameREPO repository on github.

Open a command prompt and use it to navigate to your desktop or course working directory. Then execute the following commands:

Note: lines preceded by "#" are comments to explain each step and should not be executed.

```
# clone the assignment repository onto your computer

git clone https://github.prod.oc.2u.com/UCB-INFO-
PYTHON/assignments_upstream_spring20.git

# Note: This may be an empty repository at the beginning of the course.

cd assignments_upstream_spring20

# Add the assignments repo as the upstream (Think of **upstream as the
"source"/where stuff comes from**)

git remote add upstream https://github.prod.oc.2u.com/UCB-INFO-
PYTHON/assignments_upstream_spring20.git
```

You can find the URL for YourNameREPO by navigating to the appropriate repository in your web browser, then clicking on the "Clone or download" button in the upper right corner.

```
# set the origin (think of **origin this as the "sink"/where stuff goes**) to
your personal repository

git remote remove origin
git remote add origin <ENTER YOUR REPOSITORY HTTPS URL HERE>

# i.e. git remote add origin https://github.prod.oc.2u.com/UCB-INFO-
PYTHON/Gunnar_KleemannREPO.git
```

To check if you did everything right, execute the following command:

```
git remote -v
```

- The output should show "fetch" and "push" for two remotes, one named origin and one named upstream (repo name may vary slightly from the image).

```
(pygame2) Gunnars-MacBook-Pro-2:assignments_upstream_fall18 GunnarK$ git remote -v
origin https://github.com/UCB-INFO-PYTHON/Gunnar_KleemannREPO.git (fetch)
origin https://github.com/UCB-INFO-PYTHON/Gunnar_KleemannREPO.git (push)
upstream https://github.com/UCB-INFO-PYTHON/assignments_upstream_fall18.git (fetch)
upstream https://github.com/UCB-INFO-PYTHON/assignments_upstream_fall18.git (push)
```

- You should also use the **ls** (list) command to confirm that the assignment files have been copied to your machine.

Workflow for Each Week

Each week, you will begin by navigating to your local version of spring20, and downloading the latest changes from the remote **assignments_upstream_spring20** repository. You do this with a git pull:

```
git pull upstream master
```

IMPORTANT keep the original material intact, don't modify it in place or there can be nasty merge conflicts. You will make a separate folder **assignments_upstream_spring20/SUBMISSIONS** folder to keep your work separate from the originals.

- Make a copy of your assignment and move it to the SUBMISSIONS folder.
- Complete all the exercises in the **assignments_upstream_spring20/SUBMISSIONS** folder on your local machine and commit your changes to git.
- Finally, you'll push your changes up to your personal student repository on github. You can do this with the following command:

```
git push origin master
```

Completing the Exercise

For this exercise you will post your first work to your personal version of the assignments_upstream_spring20 repository. The Github repository **Installation** contains the exercise.

- Make a new folder called **"SUBMISSIONS"** in your local assignments_upstream_spring20 folder

IMPORTANT: the SUBMISSIONS folder is the place that you can safely put modified files. Don't add or change files outside of the submissions folder because that can cause UGLY merge conflicts.

 - Try using the **mkdir** command from within your local assignments_upstream_spring20
- Now that you have your assignments_upstream repository set up on your computer *move out of it* and Clone the installation directory to your local machine

```
cd ..  
git clone https://github.prod.oc.2u.com/UCB-INFO-PYTHON/Installation.git
```

- Copy the file "First_GitHub_Exercise.txt"
 - From your local Installation repository. To your local **assignments_upstream_spring20/SUBMISSIONS** folder.
 - To copy the file you can practice using the command line **cp** command or just drag and drop the file.
- Open the "First_GitHub_Exercise.txt", answer the questions, and save.
- Commit the changes to your local repository. Go back to your command terminal and type the following.

```
git status
```

- This should confirm that you have a modified file in your repository. Go ahead and add the file.

```
git add SUBMISSIONS/

#check status to see that your file has been added as a new file (in green)

git status

#Then commit your changes.

git commit -m "completed GitHub exercise".
```

Pushing Changes to GitHub

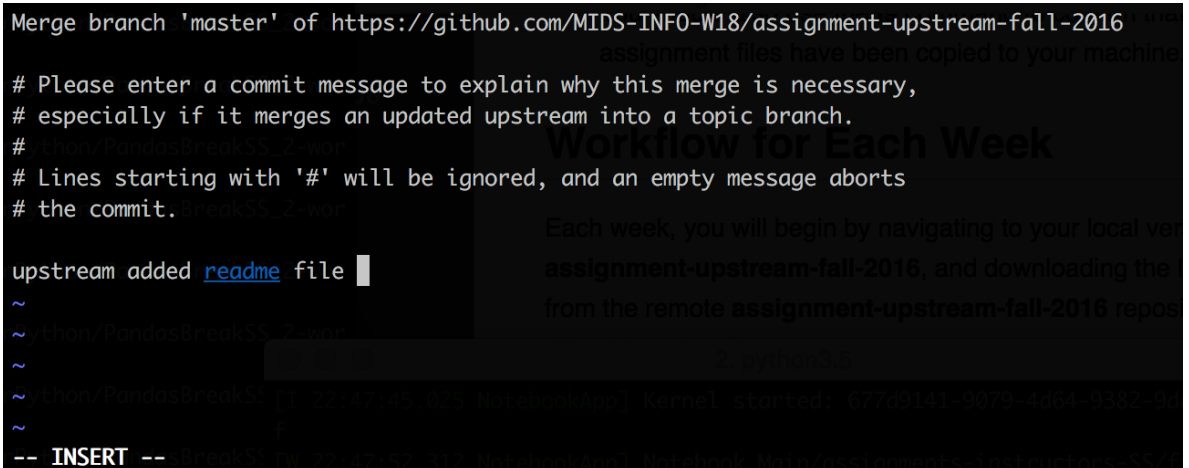
Now it is time to push your changes up to your GitHub repository. First, run `git status` to confirm that all your code is currently committed. Next, push your changes to the master branch of origin, representing your repository on GitHub.

```
git push origin master
```

Check the GitHub repository in your browser to confirm that your changes are there.

Merging

- From time to time you may have to merge the upstream and your local drive when you **pull** in a version of the repository that are not the same. You may see a screen like the one below (Note: We are using an old 2016 repository in this example)



```
Merge branch 'master' of https://github.com/MIDS-INFO-W18/assignment-upstream-fall-2016
# Please enter a commit message to explain why this merge is necessary,
# especially if it merges an updated upstream into a topic branch.
# Lines starting with '#' will be ignored, and an empty message aborts
# the commit.
upstream added readme file
~
~
~
~
-- INSERT --
```

This is VIM, a command line text editor. you need to enter a short message overwriting one of the blue tilde and then write the message to file to do this:

- 1) Type **i** to enter the "insert" mode (look to the bottom of the screen for the word "INSERT")
- 2) Use arrow keys to navigate to the line above the blue tilde and type in a message (any explanation about one line long)
- 3) Exit the insert mode by pushing **Esc** ("INSERT" will disappear) then
- 4) Type **:wq** this means **w**"write then quit" you should see some sort of message indicating that the merge was successful as shown below.

5) **Side Note:** If you have not made any changes VIM will not want to quit and you can force it to quit without recording changes using **:q!**. * Since you are entering a merge note, This should not happen, however it is good to note that VIM has a rich array of advanced commands*

```
Gunnars-MacBook-Pro:assignment-upstream-fall-2016 GunnarK$ git pull upstream master
warning: no common commits
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/MIDS-INFO-W18/assignment-upstream-fall-2016
Merge made by the 'recursive' strategy.
 README.md | 4 +++++ master -> upstream/master
1 file changed, 4 insertions(+)
create mode 100644 README.md
```