Introduction to Git

Asad Haris

Department of Biostatistics
University of Washington

May 4, 2016

Step 0: Installation

- **Install Git**: `https://git-scm.com/downloads`
- **Easy Install for Linux**
  - `apt-get install git` (Ubuntu/Debian)
  - `yum install git` (Fedora)
  - Others can be found at `https://git-scm.com/download/linux`
- This is all we need to get started. You can work with Git completely off-line.
- Make an account at `https://github.com/` or `https://bitbucket.org/` We will discuss differences between the two.

Step 0: Installation

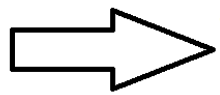
- **Install Git**: `https://git-scm.com/downloads`
- **Easy Install for Linux**
  - `apt-get install git` (Ubuntu/Debian)
  - `yum install git` (Fedora)
  - Others can be found at `https://git-scm.com/download/linux`
- This is all we need to get started. You can work with Git completely off-line.
- Make an account at `https://github.com/` or `https://bitbucket.org/` We will discuss differences between the two.

While everything is downloading/installing...

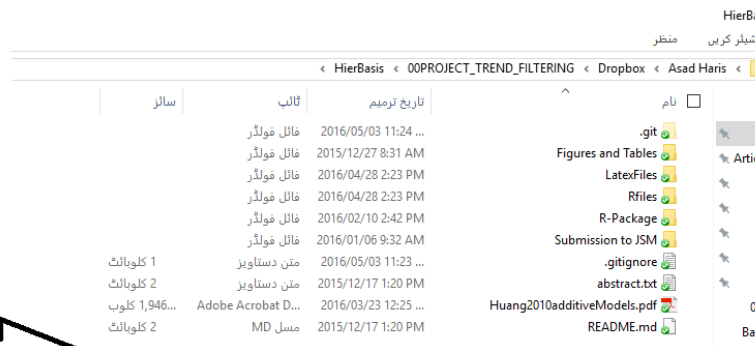
What is Git?

- Git is a version control system
- As the name suggests, a system to manage different versions of a project
- A project in Git is called a repository/repo
- Git allows us to take it a step further with many other features
- Collaborating on a project is much simpler with Github/Bitbucket

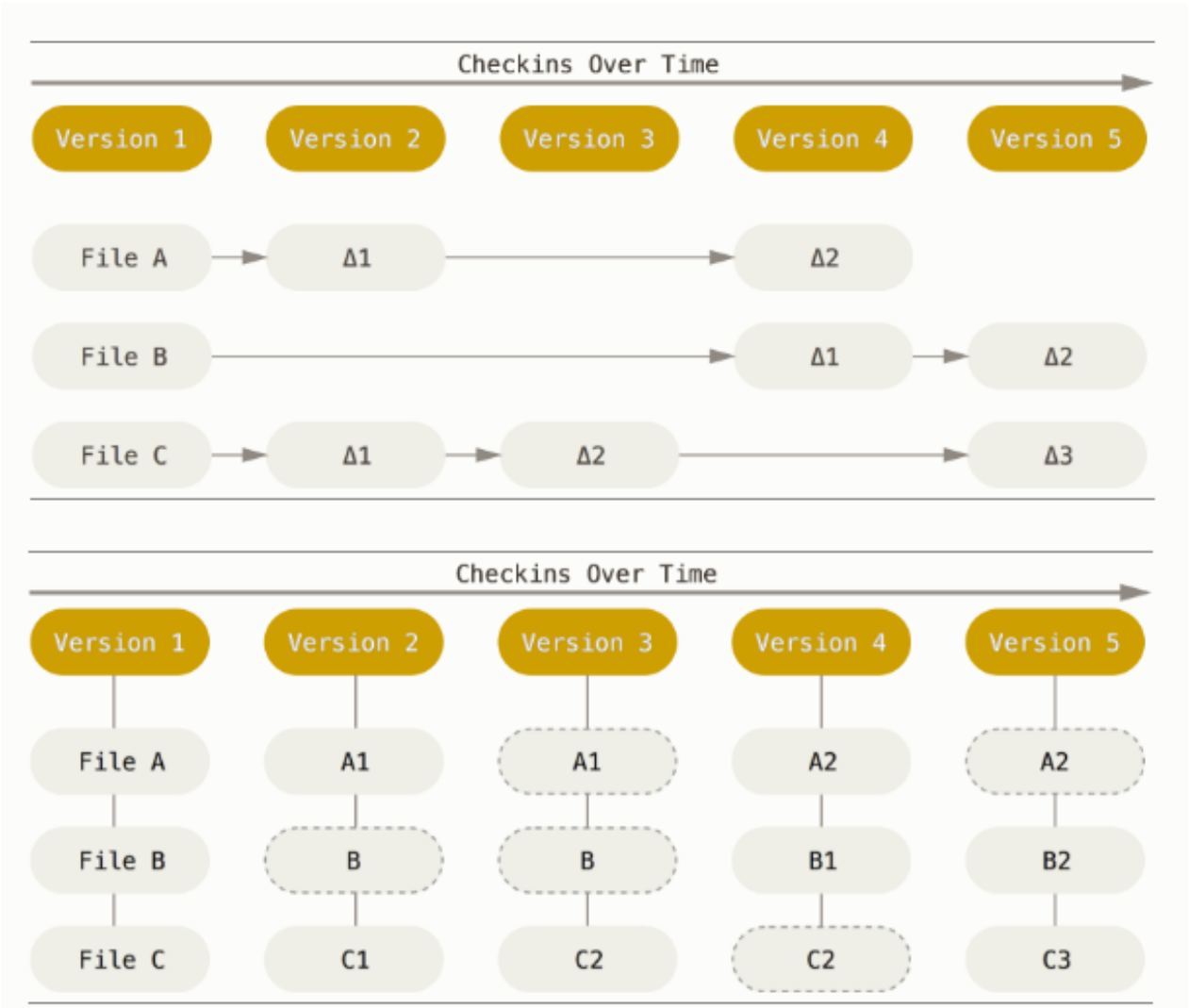
My First project



My Current Project



How does git work?



So why Git?

Advantages

- Allows you manage different versions of your project
- We can go back in time to previous versions
- Isn't restricted to specific type of projects (not just for computer scientists)
- Makes collaboration on a project really easy
- We have nice tools like GitHub and Bitbucket for collaboration and online sharing

Disadvantages

- Initial learning curve, which we will overcome today

As promised Github vs. Bitbucket

Essentially, Github/Bitbucket is a remote location to store/share your repositories

	Github	Bitbucket
Cost	Free	Free
Public Repositories	Unlimited	Unlimited
Private Repositories	5[1]	Unlimited
Collaborators	Unlimited	5[2]

Table: 1. After student discount. 2. For the free account, can have upto unlimited collaborators with paid account.

Working with Git

First Steps: Configure Git

Once Git is installed, we begin with configuring Git.

```
git config --global user.name "asadharis"
git config --global user.email aharis@uw.edu
git config --global color.ui true
```

This only needs to be done once!

Outline of Project

We will consider a simple project: A mini version of an assignment

1. We will first *initialize* a git repo
2. Begin writing up the assignment
3. Make an Rfile for all our work in R
4. Upload our Repo to github and collaborating

Starting a repo is easy

Starting a git Repo is very simple

```
# Make a dir where we want our project to reside
mkdir hw1_stat101
# Move to the working directory
cd hw1_stat101

# Initialize a Git repo
git init   # Really this is it!

# Check to see what the command did
ls -l -a   # Should see the .git folder
```

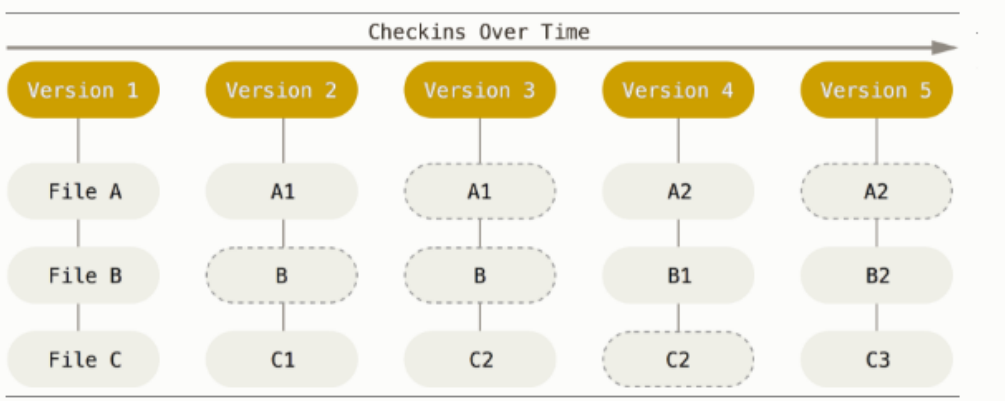
Begin work on the project

```
mkdir latex_files r_files
cd latex_files

vim hw1.tex
... Latex text goes here ...

pdflatex hw1.tex
```

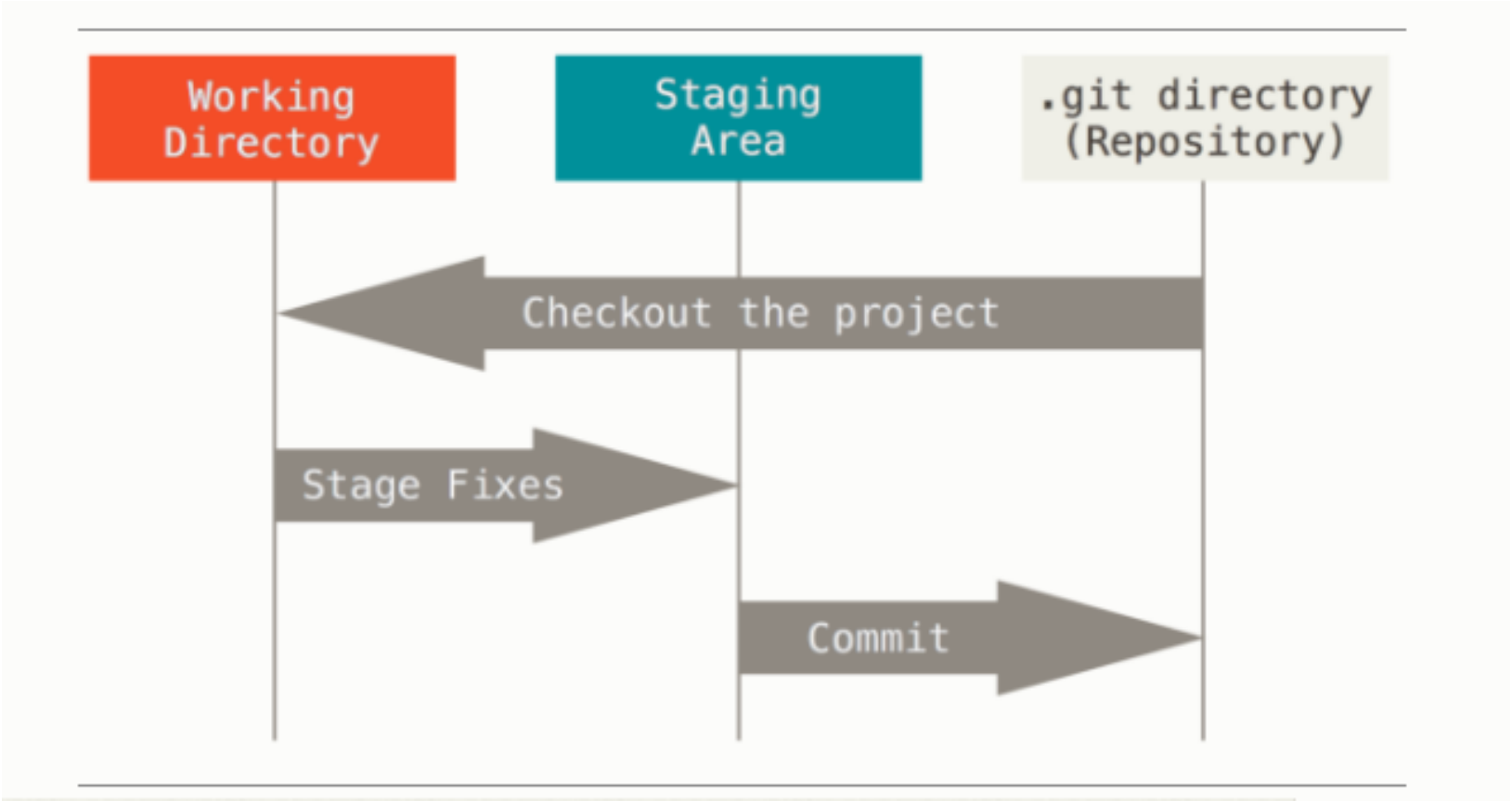
The first commit

Making a new commit is as easy as

```
git status
git add --all
git commit -m "My first commit"
```



Some details about commiting

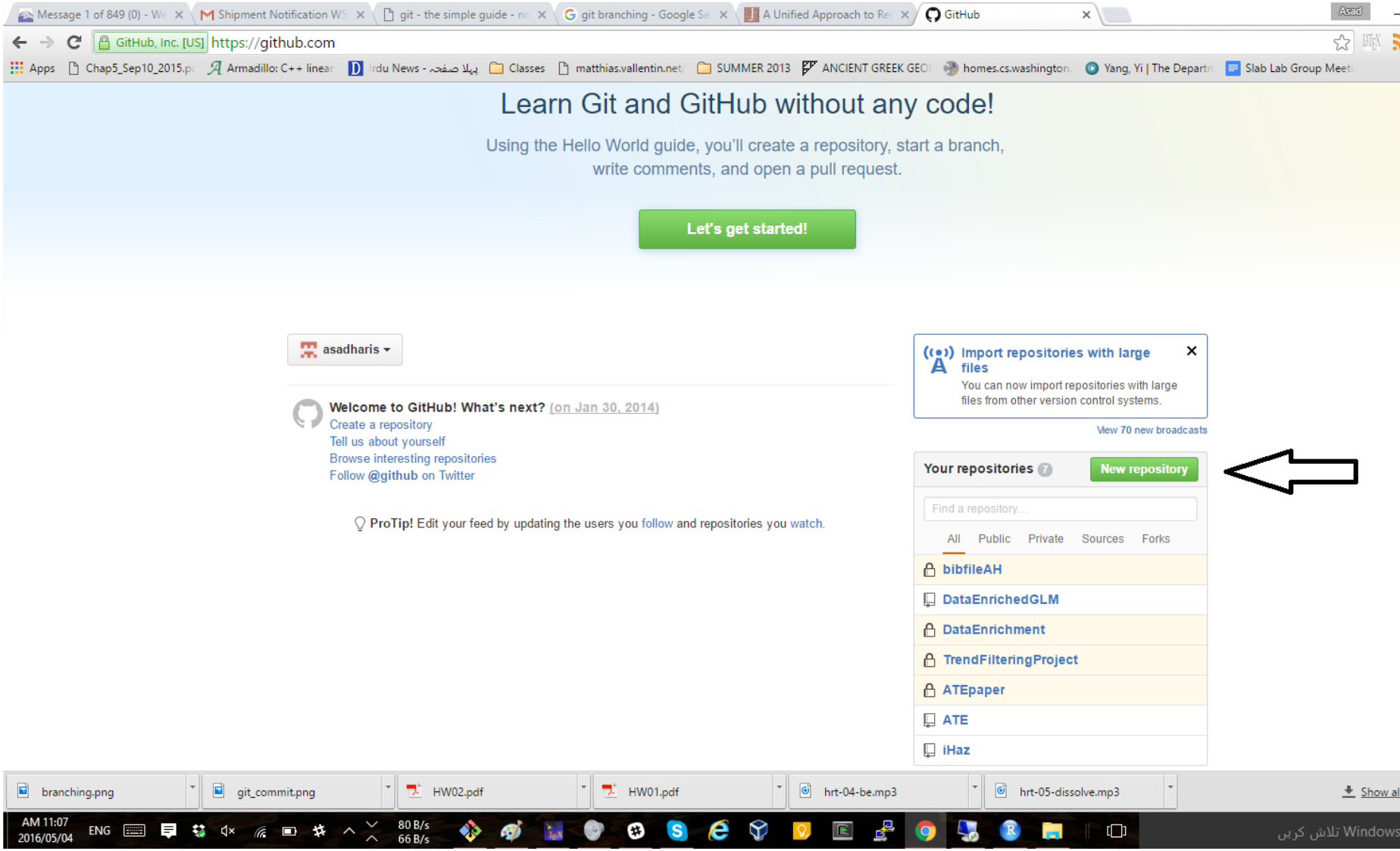


Adding an R file and Git history

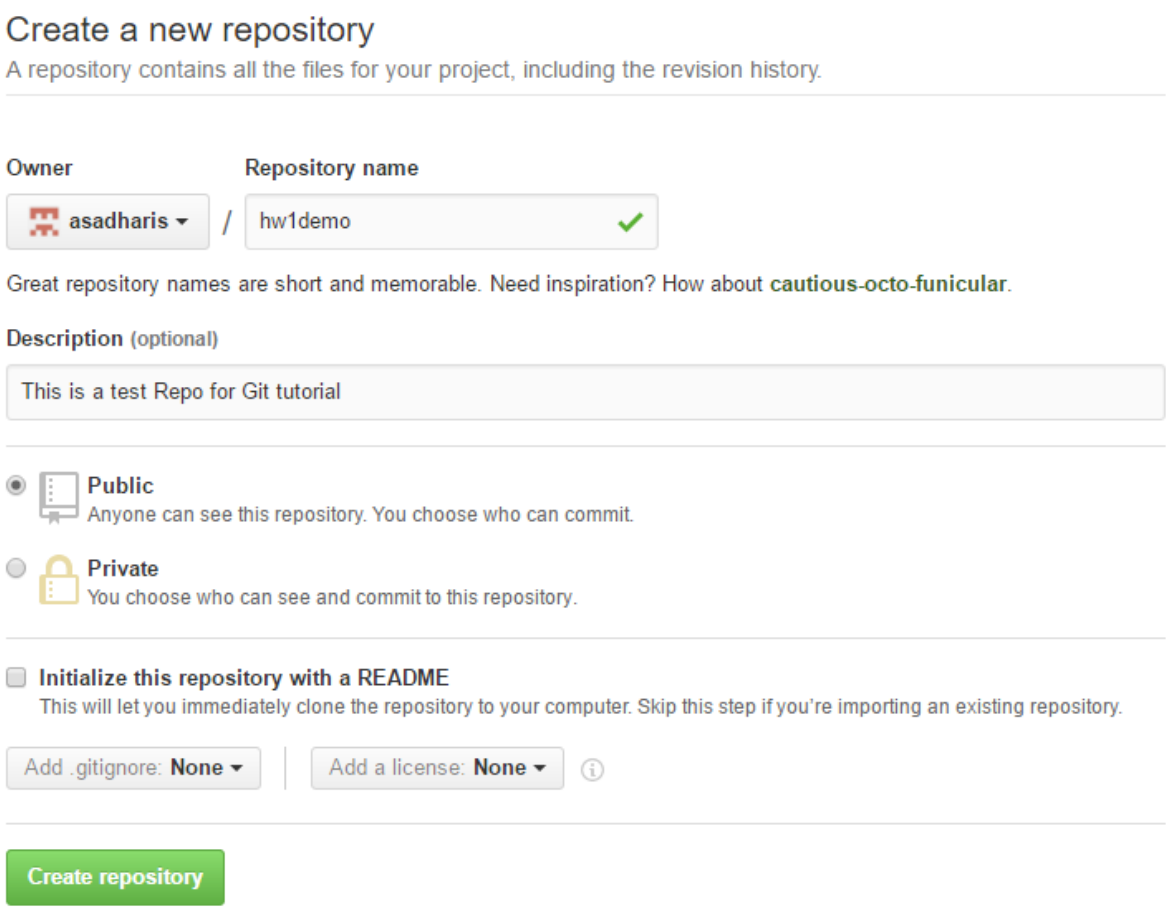
We can now add an Rfile, run our simulations and make another commit.
After this we now see the history of our project using the `log` command

```
git log  # Basic view

git log --help  # View other options

git log --graph   # The one I like
```

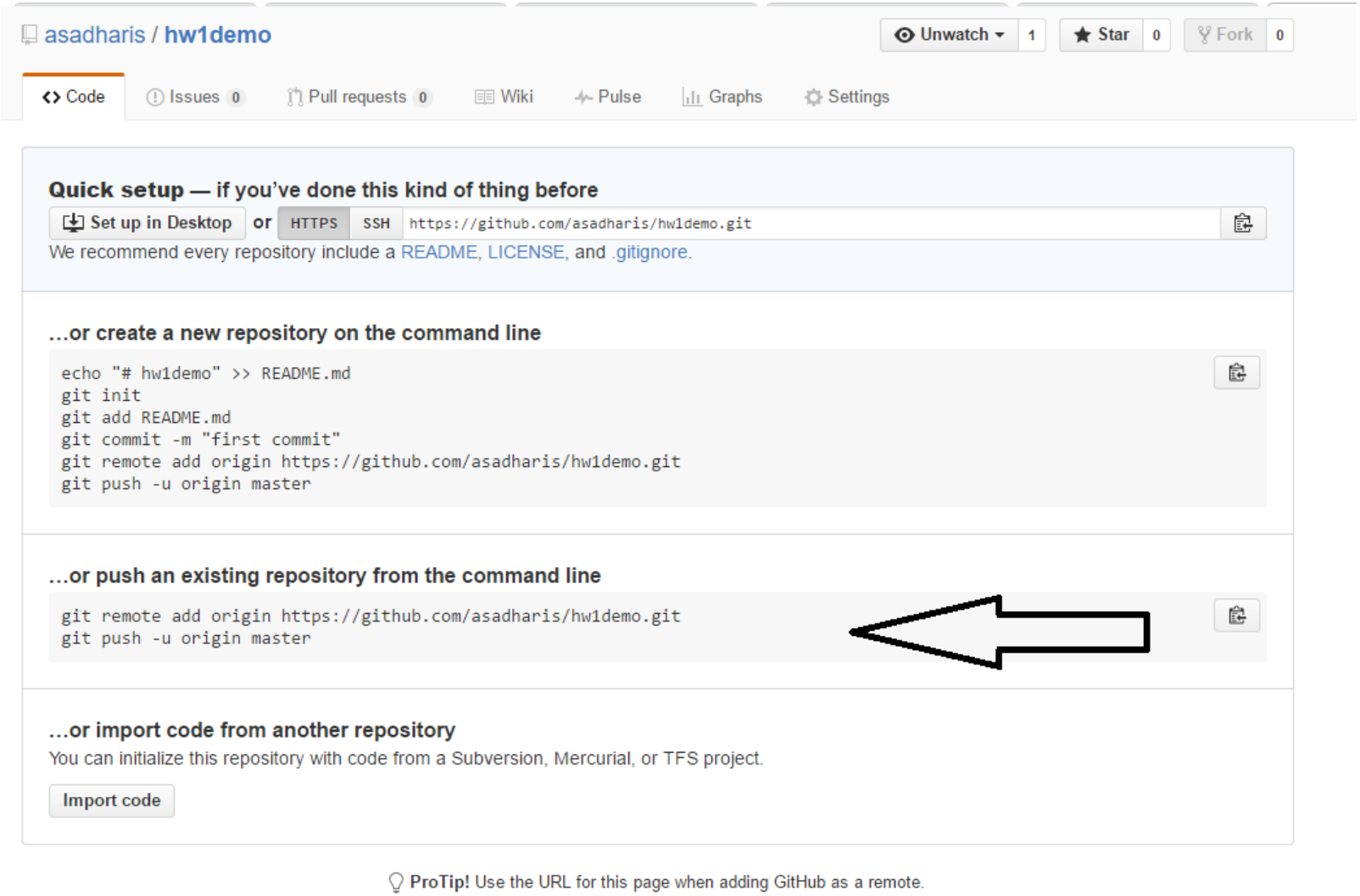
Working online: Sign-in to Github



Working online: Make the Repo



Working online: we will use option 2



Possible issue

The instructions on github may not work for linux/mac. So try the following

```
git remote set-url origin \\
 https://github.com/asadharis/hw1demo.git
```

Collaborating

Say your collaborator made changes to project and you begin work the next day.
First you need to bring in all the changes they made. For this we have the pull command

```
git pull
```

Working in Git is really just these 5 commands

`pull`	Pulls down current version of repo
`status`	check status of repo
`add`	add files to staging area
`commit`	commit staged files
`push`	Pushes files to Remote

Collaborating

Say your collaborator made changes to project and you begin work the next day.
First you need to bring in all the changes they made. For this we have the pull command

```
git pull
```

Working in Git is really just these 5 commands

`pull`	Pulls down current version of repo
`status`	check status of repo
`add`	add files to staging area
`commit`	commit staged files
`push`	Pushes files to Remote

Review what we have done

A short summary of what we have done so far

```
git init  # Start a repo/begin work
git status  # Check status of changes made
git add  --all # Stage all the files
git commit -m "Some message"   # A version is complete

# Set-up remote and then
git push
git pull

# Check progress of project
git log
```

Going back in Time

We have two options

1. **Review history/old versions**

```
git checkout [commit-number]
```

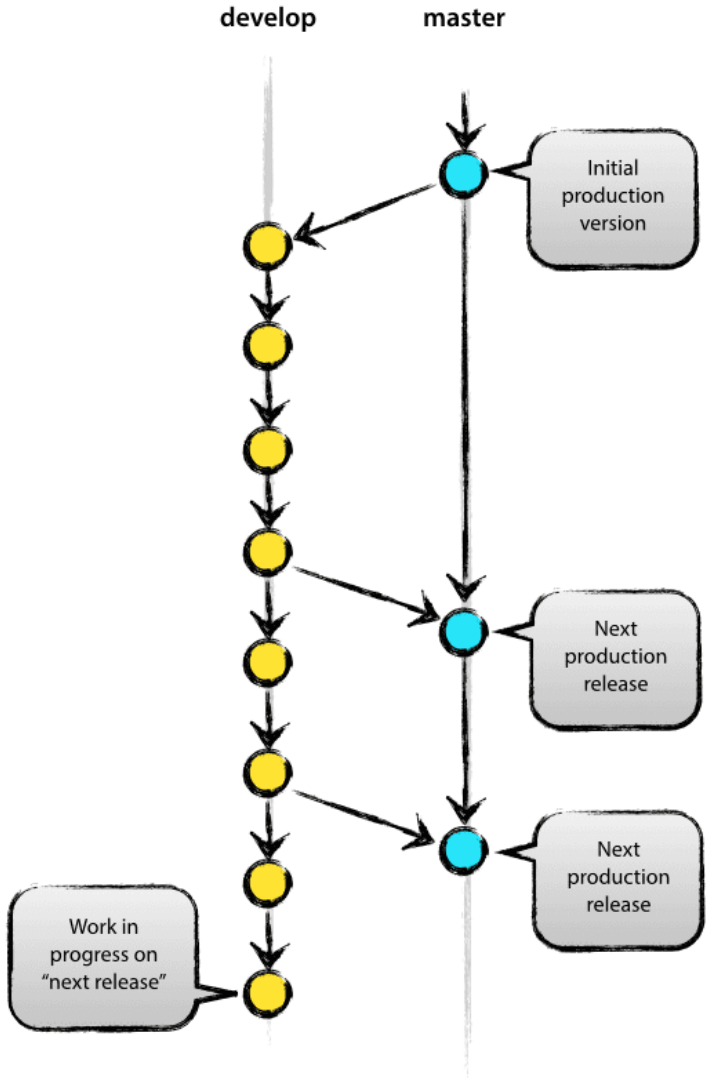
2. **Go back in time or start over**

```
git reset [commit-number]  # Keep local changes
git reset [commit-number] --hard  # Destroy local changes
```

Merge Conflict

If two people are working on the same file we may have some conflicts.
See terminal

Branching


develop
master
Initial production version
Next production release
Next production release
Work in progress on "next release"

Branching

For working with branches we have the following commands

```
git branch  # View all branches
git branch testV2  # Make a new branch
git checkout testV2  # Move to other branch
```

The .gitignore file

- There may be some files we don't want to track
- In some cases it is good practice to not track some files. e.g. the extra files latex generates will always lead to merge conflicts
- Add all untracked files in a text file called: .gitignore
- I usually borrow a gitignore file and add things to it. e.g. `https://gist.github.com/kogakure/149016`

Conclusion

- Git can be used for projects of all sizes
- Other useful things can be done with a bit of creativity
- Many tutorials out there:
  - (Official) `https://git-scm.com/docs/gittutorial`
  - (More detailed) `https://www.atlassian.com/git/tutorials/`
  - (My favorite) `https://www.youtube.com/watch?v=0fKg7e37bQE`
- A cheat-sheet I use `https://training.github.com/kit/downloads/github-git-cheat-sheet.pdf`
- As always google is your friend