

A Simple Introduction to Git: a distributed version-control system

CS 5010 Program Design Paradigms
“Bootcamp”
Lesson 0.5



© Mitchell Wand, 2012-2014

This work is licensed under a [Creative Commons Attribution-NonCommercial 4.0 International License](https://creativecommons.org/licenses/by-nc/4.0/).

Learning Objectives

- At the end of this lesson you should be able to explain:
 - how git creates a mini-filesystem in your directory
 - what commit, push, pull, and sync do
 - the elements of the basic git workflow
 - how git allows you to work across multiple computers
 - how git allows you and a partner to work together

Git is a **distributed** version-control system

- You keep your files in a *repository* on your local machine.
- You synchronize your repository with a repository on a server.
- If you move from one machine to another, you can pick up the changes by synchronizing with the server.
- If your partner uploads some changes to your files, you can pick those up by synchronizing with the server.

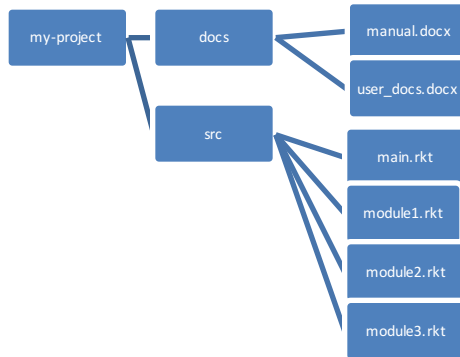
Git is a distributed **version-control** system

- Terminology: In git-speak, a “version” is called a “commit.”
- Git keeps track of the history of your commits, so you can go back and look at earlier versions, or just give up on the current version and go back some earlier version.

A simple model of git

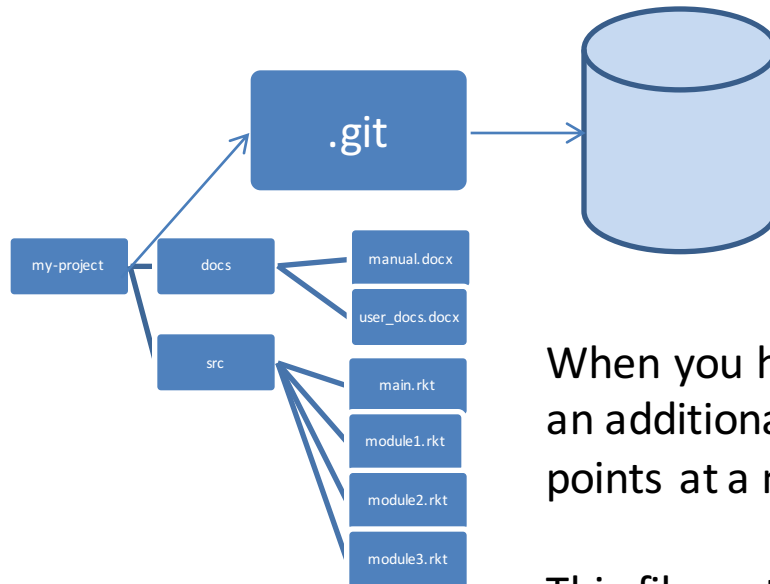
- Most git documentation gets into details very quickly.
- Here's a very simple model of what's going on in git.

Your files



Here are your files, sitting
in a directory called my-
project

Your files in your git repository

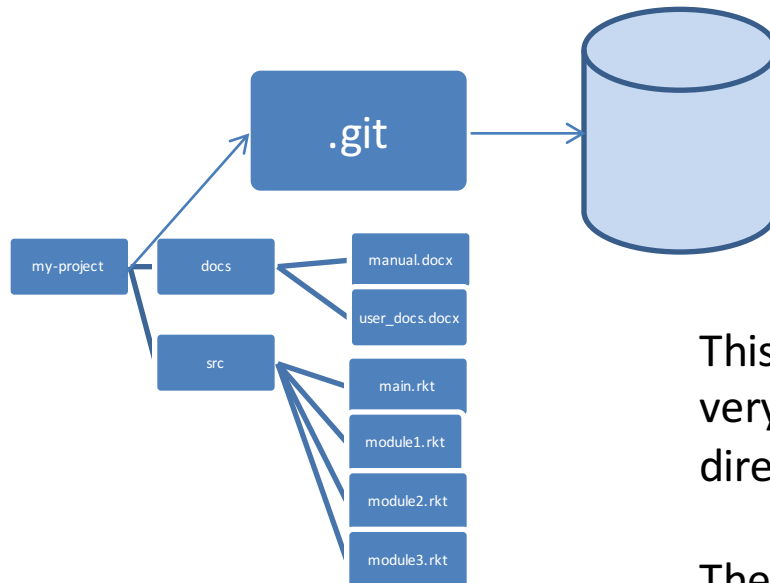


When you have a git repository, you have an additional directory called `.git`, which points at a mini-filesystem.

This file system keeps all your data, plus the bells and whistles that git needs to do its job.

All this sits on your local machine.

The git client



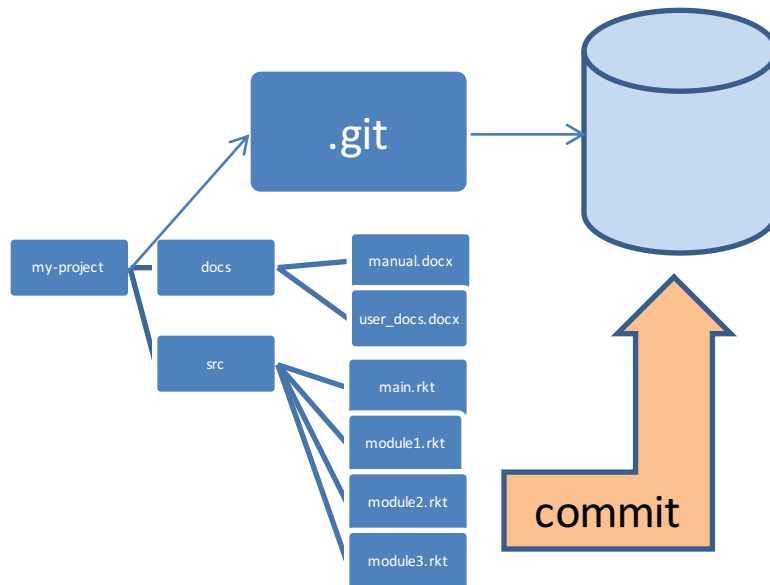
This mini-filesystem is highly optimized and very complicated. Don't try to read it directly.

The job of the git client (either Github for Windows, Github for Mac, or a suite of command-line utilities) is to manage this for you.

Your workflow (part 1)

- You edit your local files directly.
 - You can edit, add files, delete files, etc., using whatever tools you like.
 - This doesn't change the mini-filesystem, so now your mini-fs is behind.

A Commit



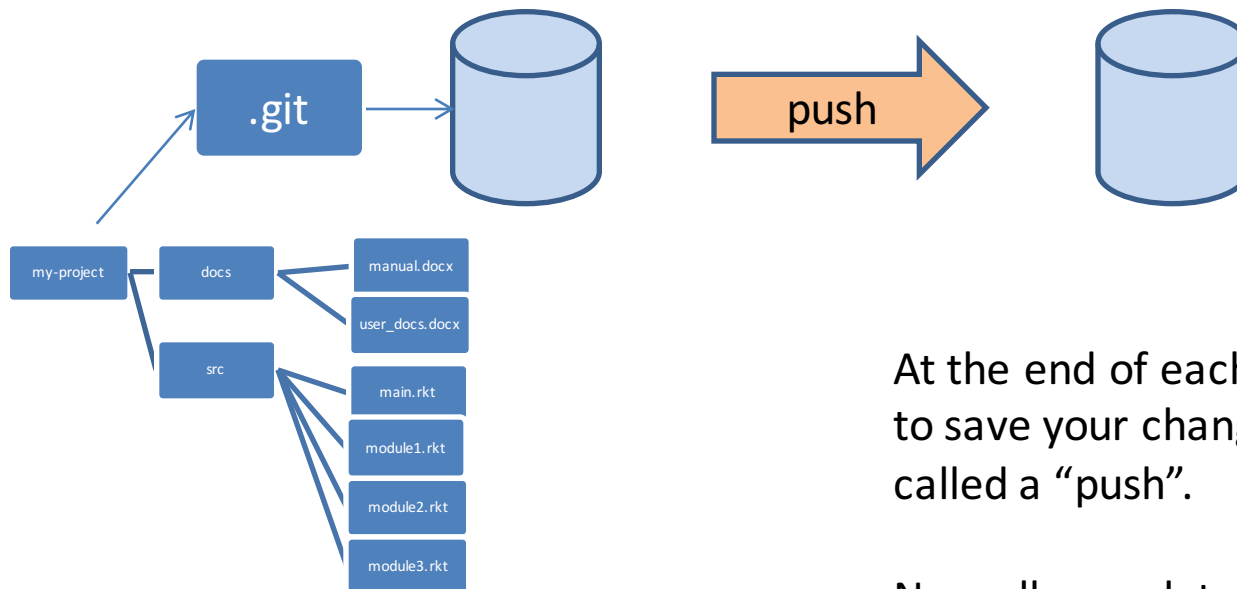
When you do a “commit”, you record all your local changes into the mini-fs.

The mini-fs is “append-only”. Nothing is ever over-written there, so everything you ever commit can be recovered.

Synchronizing with the server (1)

your local machine

a server, somewhere on the internet, eg. github.com

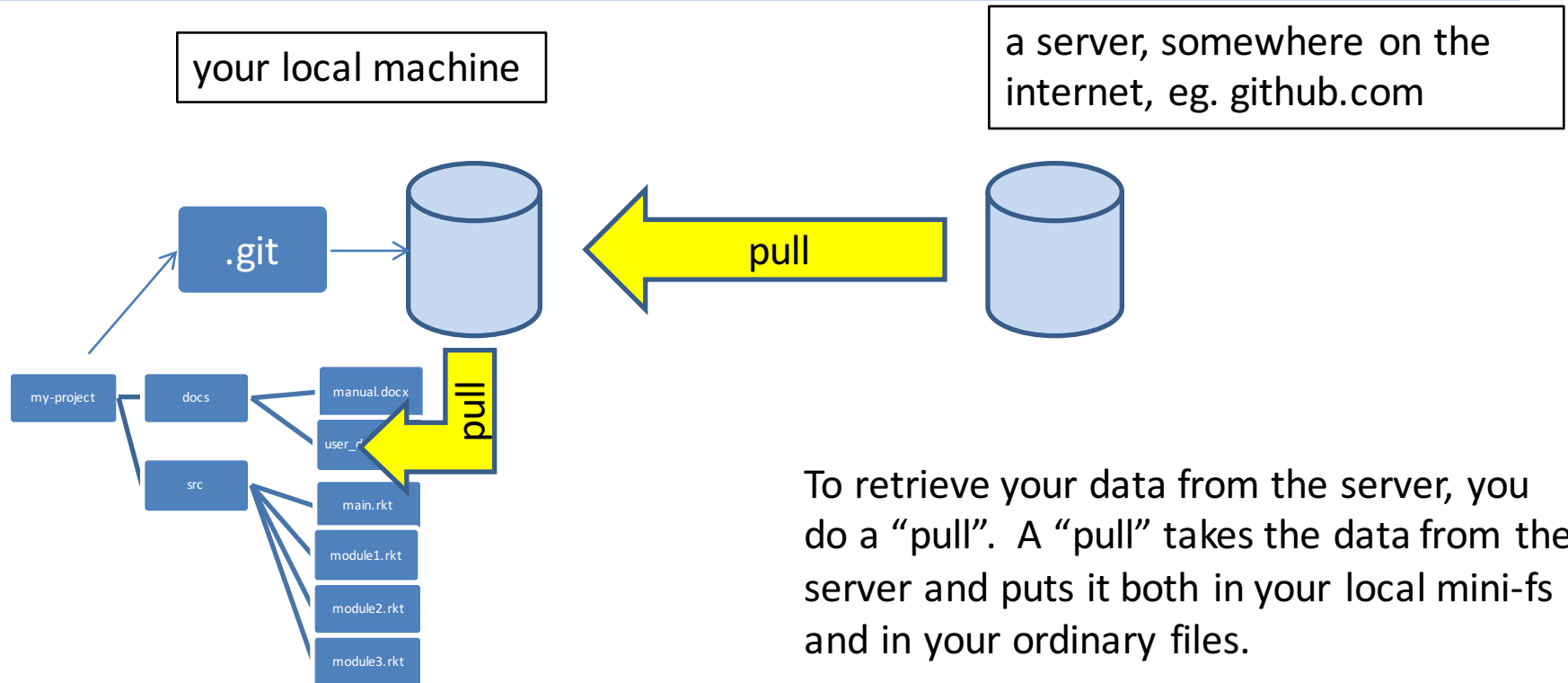


At the end of each work session, you need to save your changes on the server. This is called a “push”.

Now all your data is backed up.

- You can retrieve it, on your machine or some other machine.
- We can retrieve it (that’s how we collect homework)

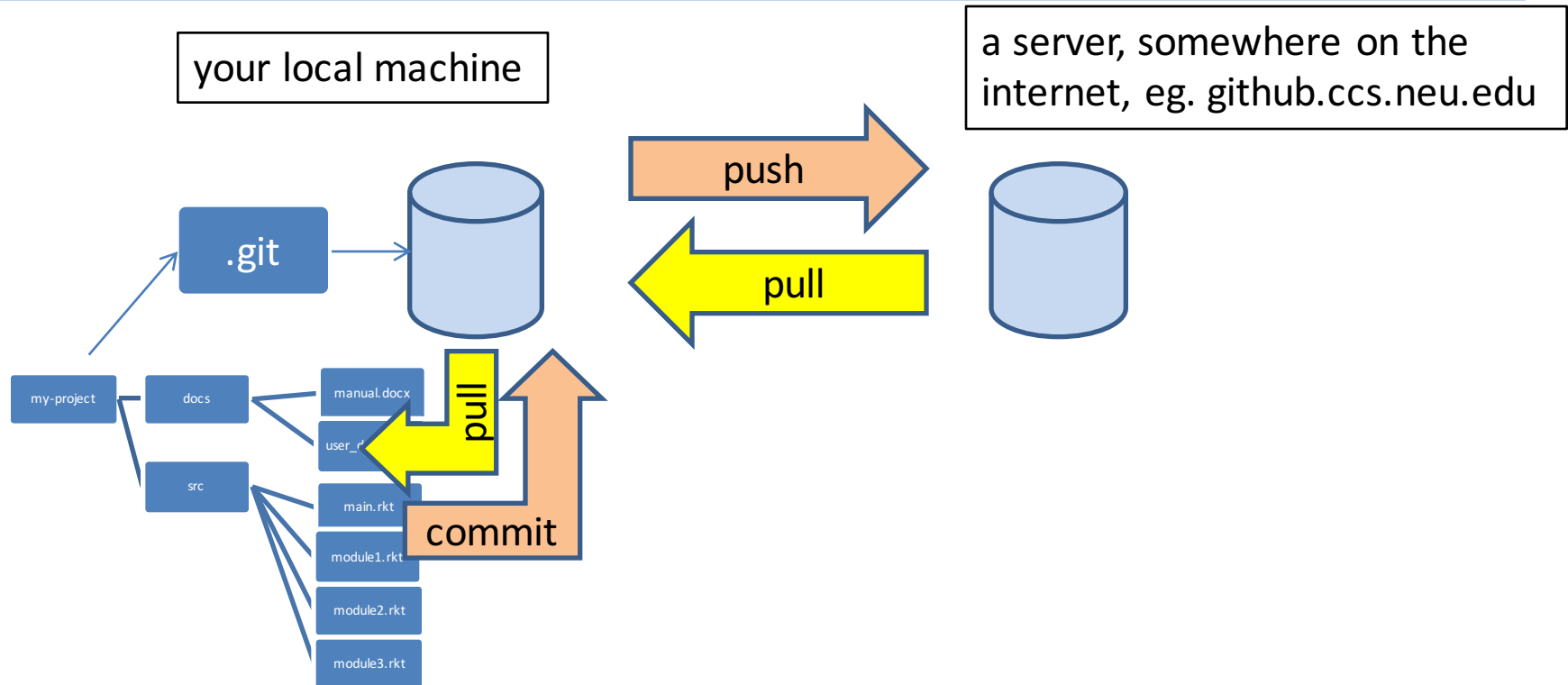
Synchronizing with the server (2)



To retrieve your data from the server, you do a “pull”. A “pull” takes the data from the server and puts it both in your local mini-fs and in your ordinary files.

If your local file has changed, git will merge the changes if possible. If it can't figure out how to the merge, you will get an error message. We'll learn how to deal with these in the next lesson.

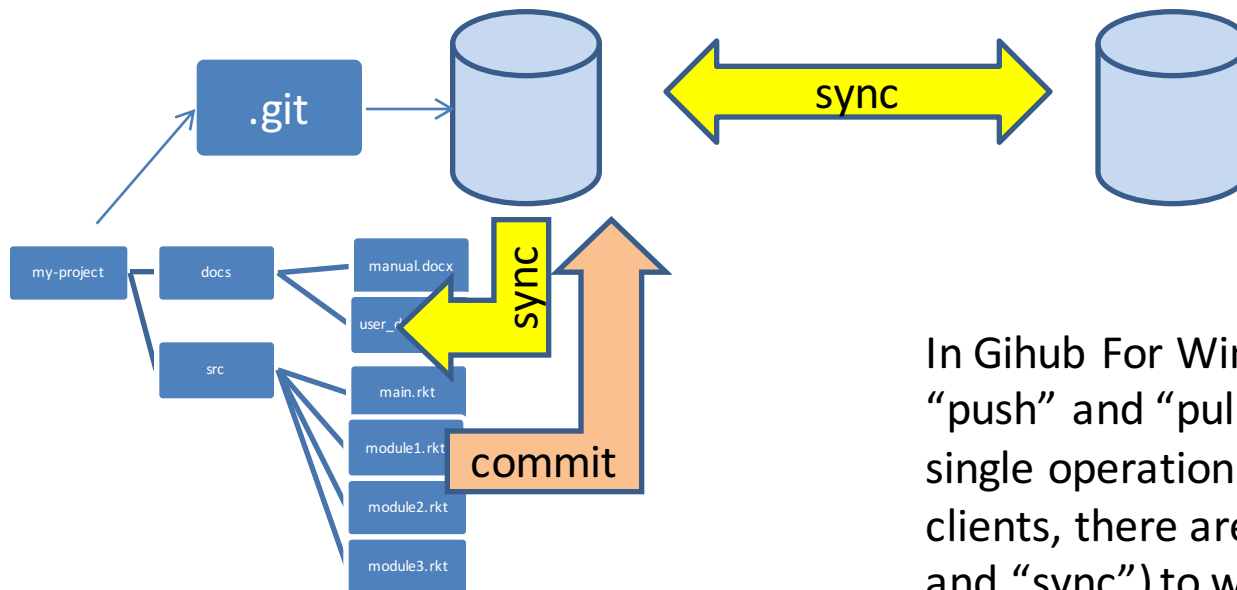
The whole picture



The whole picture using GHFW

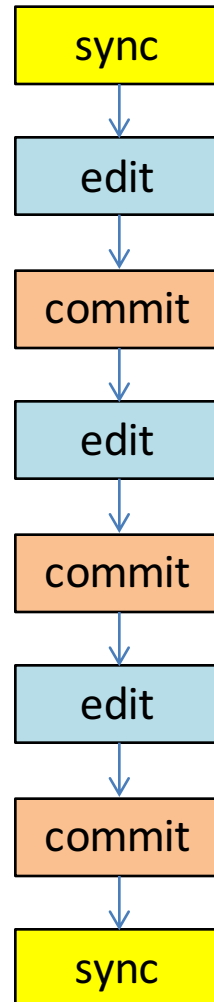
your local machine

a server, somewhere on the internet, eg. github.ccs.neu.edu



In Github For Windows or Github For Mac, “push” and “pull” are combined into a single operation called “sync”. So in these clients, there are only two steps (“commit” and “sync”) to worry about, not three.

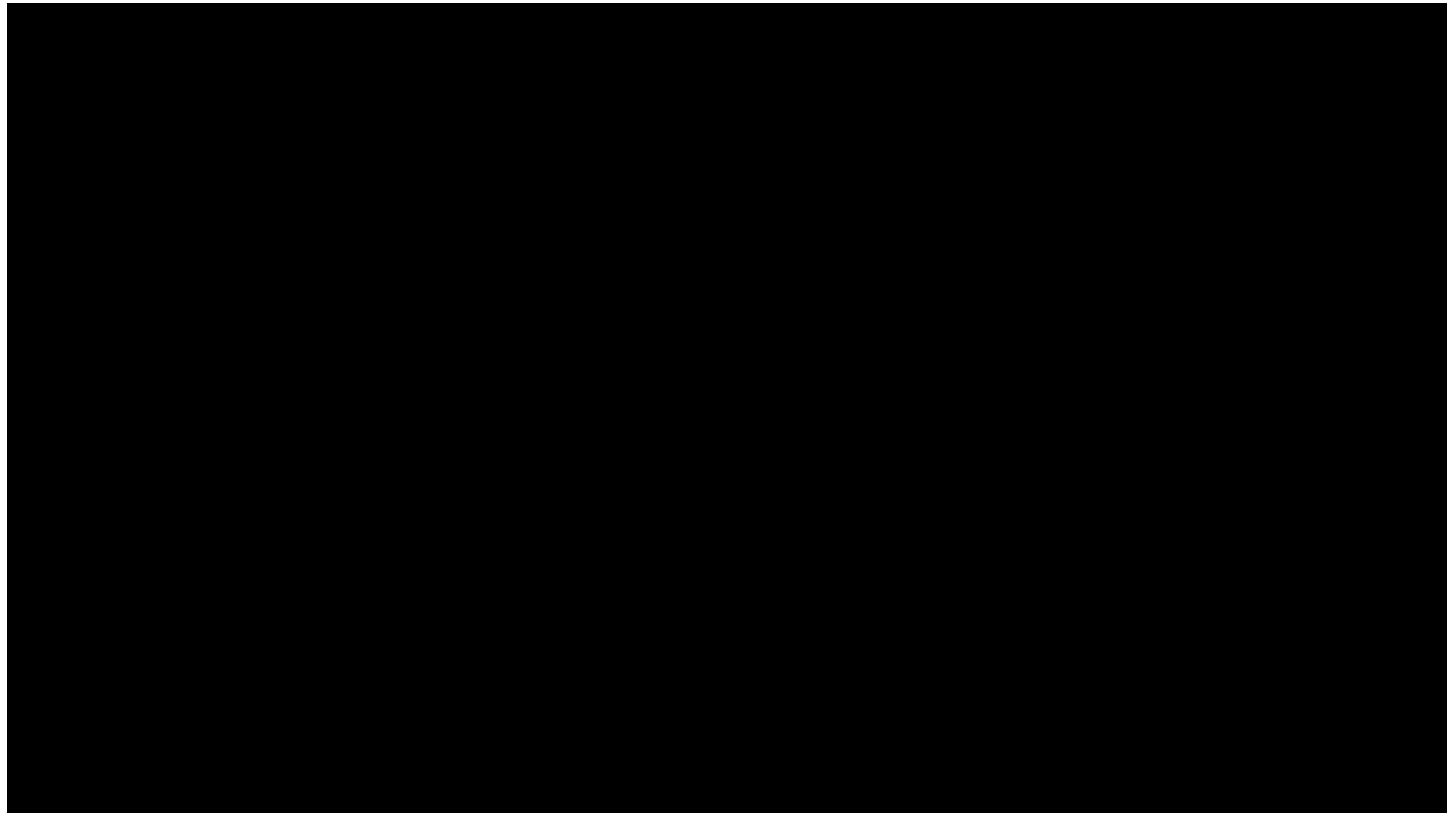
Your workflow (2)



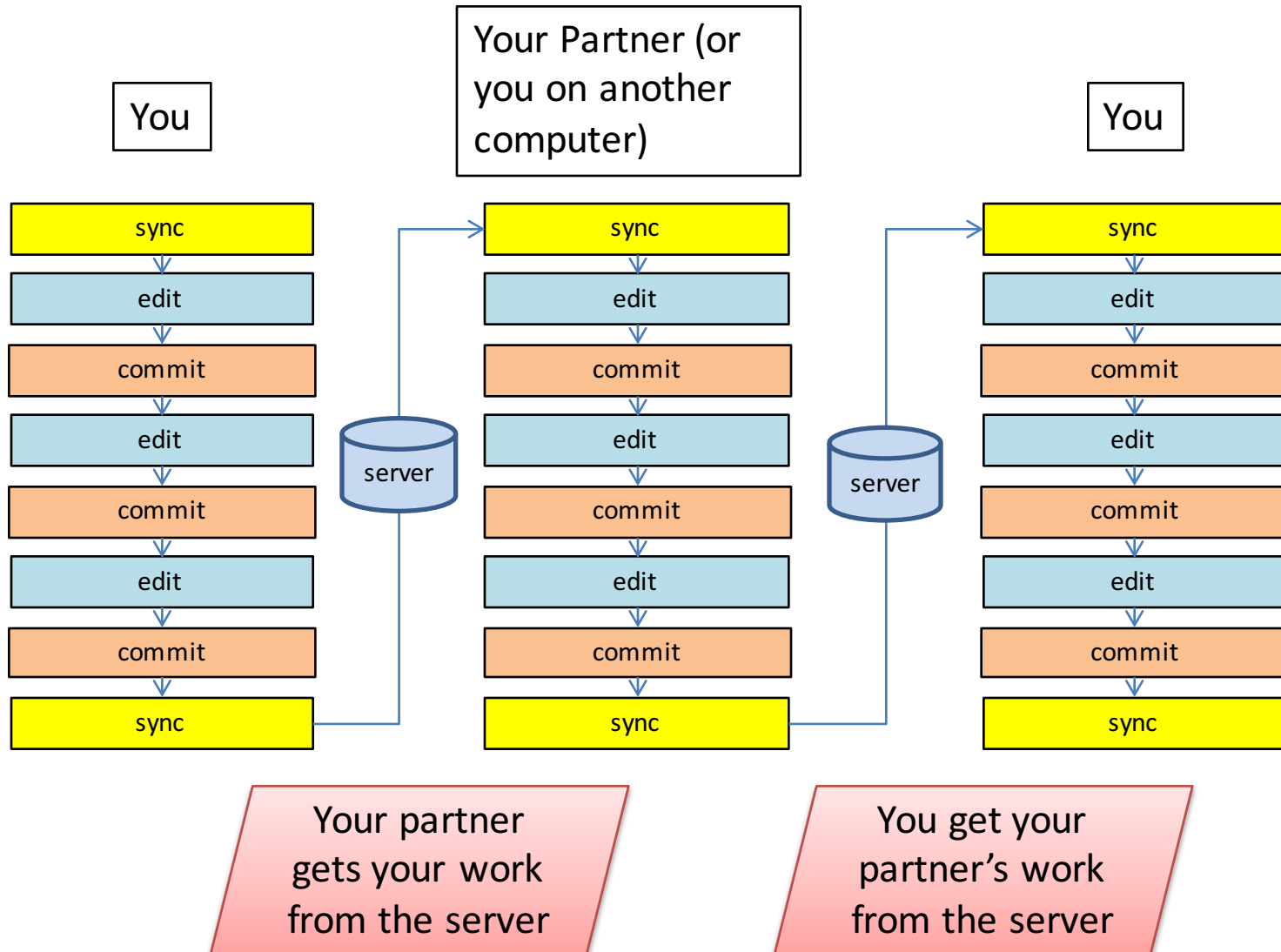
Best practice: commit your work whenever you've gotten one part of your problem working, or before trying something that might fail.

If your new stuff is screwed up, you can always "revert" to your last good commit. (Remember: always "revert", never "roll back")

Using Github for Windows/Mac



Your workflow with a partner

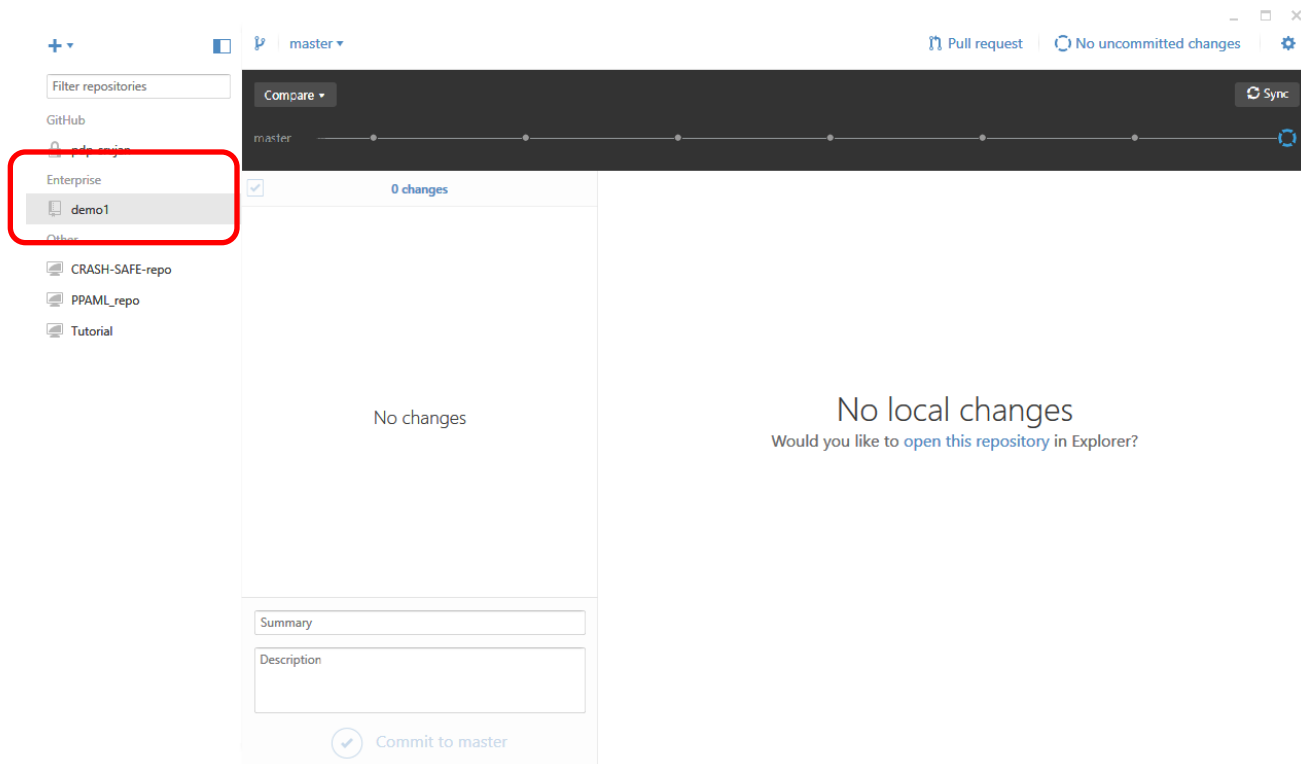


The new github desktop

- In the next few slides, we'll give you the updates for 2015.
- We won't be using github.com. Instead we will be using "Github for Enterprise" at <https://github.ccs.neu.edu>
- The user interface for GHFW/GHFM has changed. It's now called "Github Desktop".
- In the next few slides, we'll show you how your daily workflow looks with new interface.

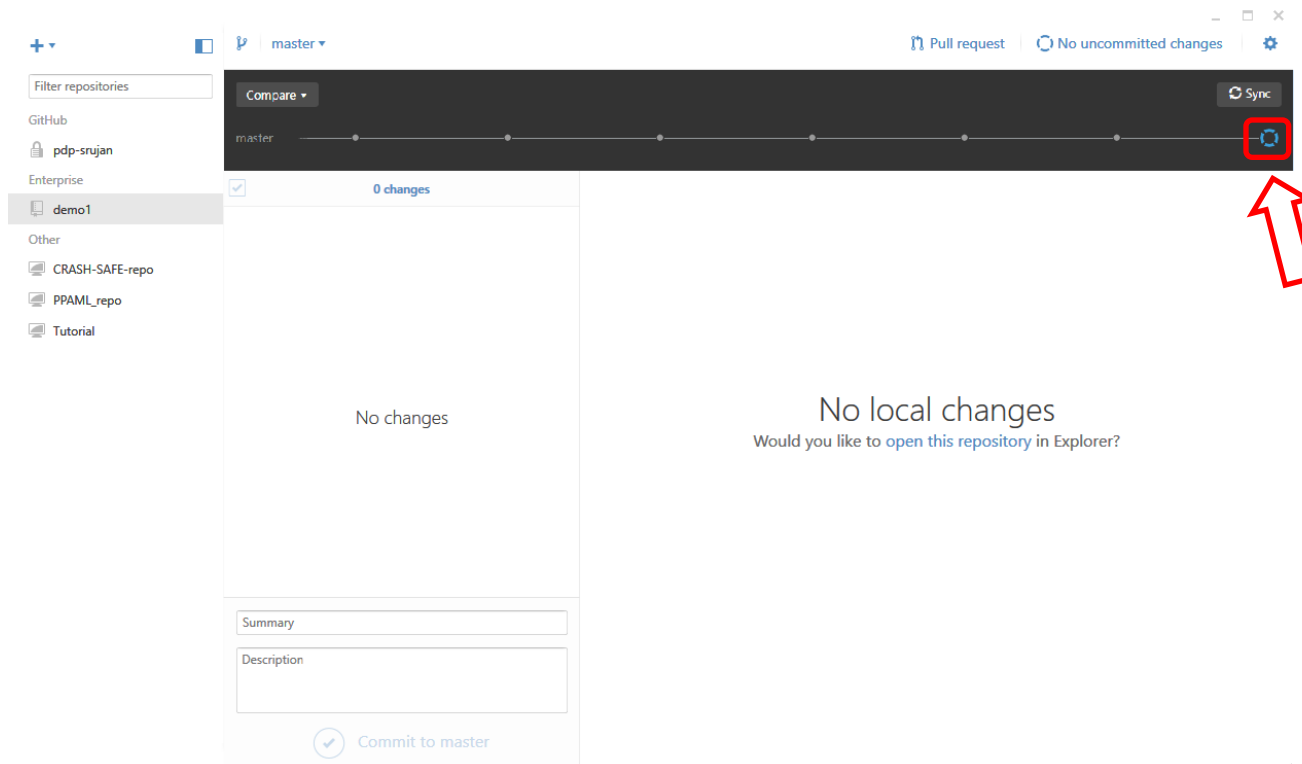
Starting your work session

- Here's what your Github Desktop should look like when you open it up. Observe that your repos will be in the section labeled "Enterprise".



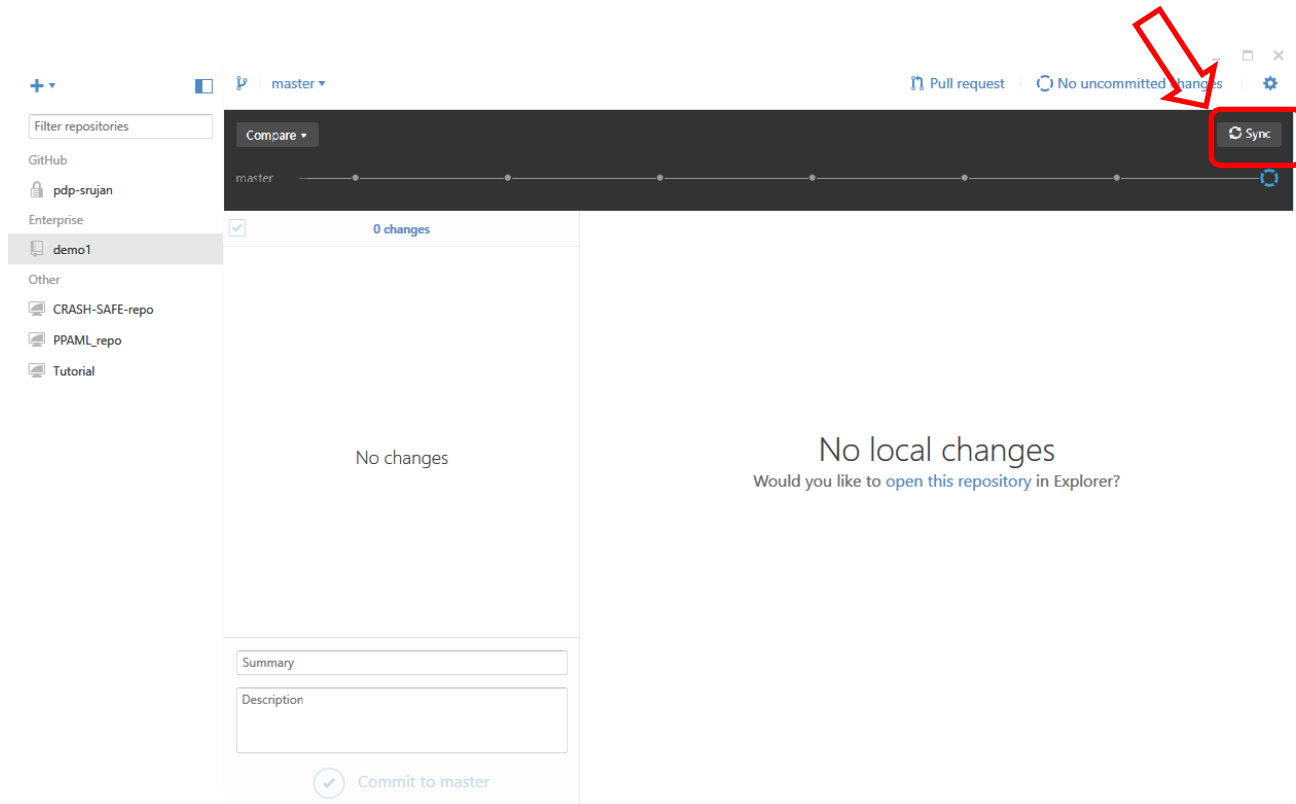
Where am I?

- The open blue circle indicates that you are looking at the most recent local files



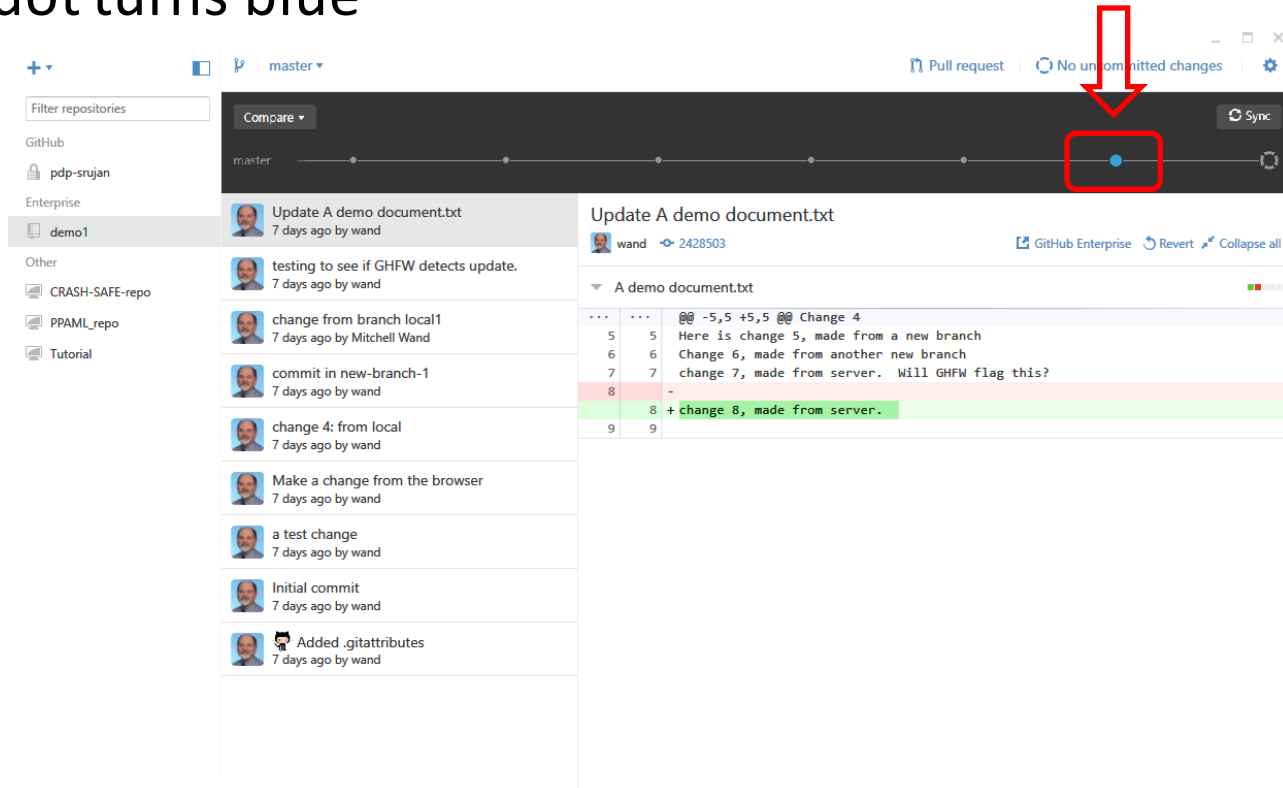
Always start by syncing

- This will download any changes that you or your partner have made on other machines



Click on a dot to see a commit

- Clicking on the last dot will show you what was in your last commit
- The dot turns blue



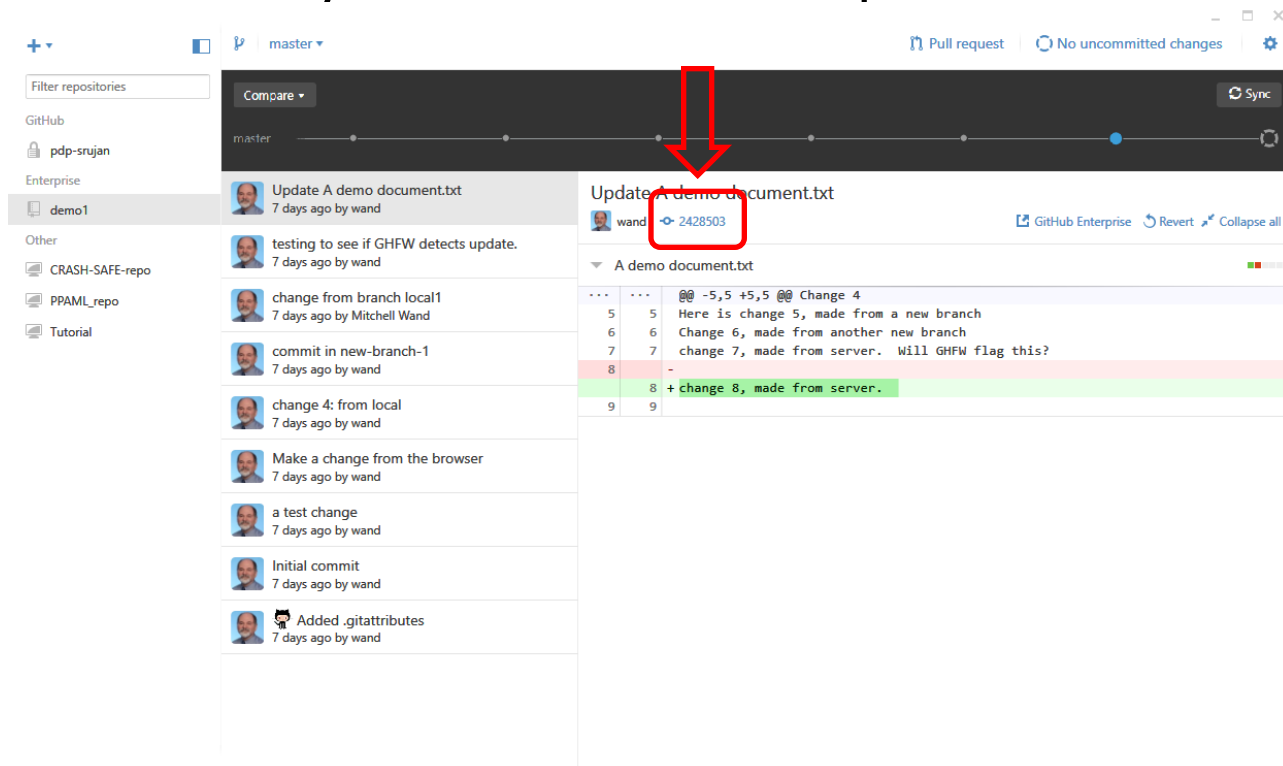
The screenshot shows the GitHub interface for a repository. The 'Compare' dropdown is open, displaying a list of commits. The most recent commit, 'Update A demo document.txt', is highlighted with a blue dot. A red arrow points to this dot, and a red box highlights it. The commit details for 'Update A demo document.txt' are shown on the right, including the commit message and a diff view.

Commit	Author	Message
Update A demo document.txt	wand	7 days ago by wand
testing to see if GHFW detects update.	wand	7 days ago by wand
change from branch local1	Mitchell Wand	7 days ago by Mitchell Wand
commit in new-branch-1	wand	7 days ago by wand
change 4: from local	wand	7 days ago by wand
Make a change from the browser	wand	7 days ago by wand
a test change	wand	7 days ago by wand
Initial commit	wand	7 days ago by wand
Added .gitattributes	wand	7 days ago by wand

```
@@ -5,5 +5,5 @@ Change 4
5 5 Here is change 5, made from a new branch
6 6 Change 6, made from another new branch
7 7 change 7, made from server. Will GHFW flag this?
8 -
8 + change 8, made from server.
9 9
```

This shows your commit SHA

- In this view, you can see the first 6 characters of the unique identifier (“the SHA”) for this commit
- You’ll need it for your Worksession Report

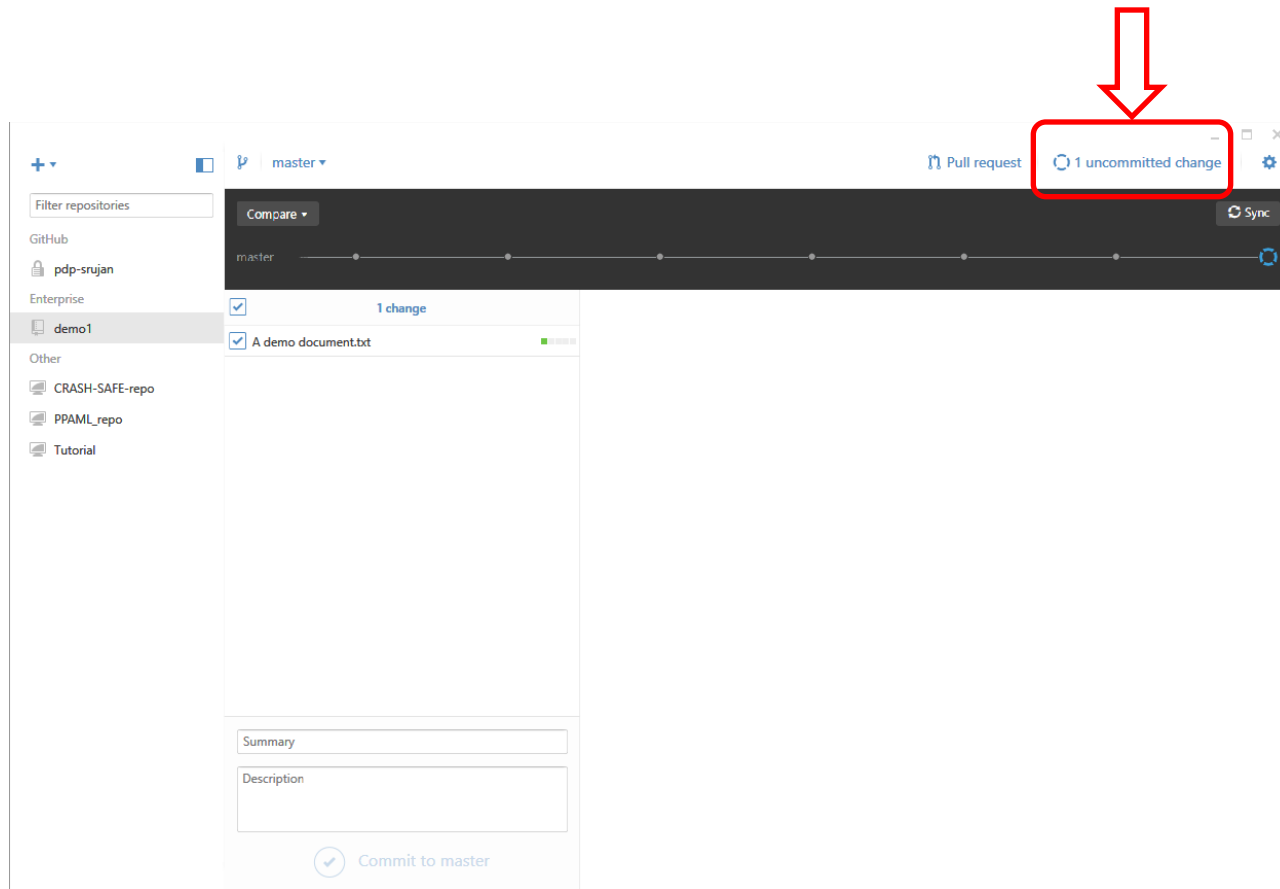


The screenshot shows a GitHub commit page for the file "A demo document.txt". The commit is titled "Update A demo document.txt" and was made by user "wand" 7 days ago. The commit SHA is "2428503", which is highlighted with a red box and a red arrow. The commit message is "Update A demo document.txt". The diff shows changes to the file, with the most recent change being "+ change 8, made from server.".

```
@@ -5,5 +5,5 @@ Change 4
5 5 Here is change 5, made from a new branch
6 6 Change 6, made from another new branch
7 7 change 7, made from server. Will GHFW flag this?
8 -
8 + change 8, made from server.
9 9
```

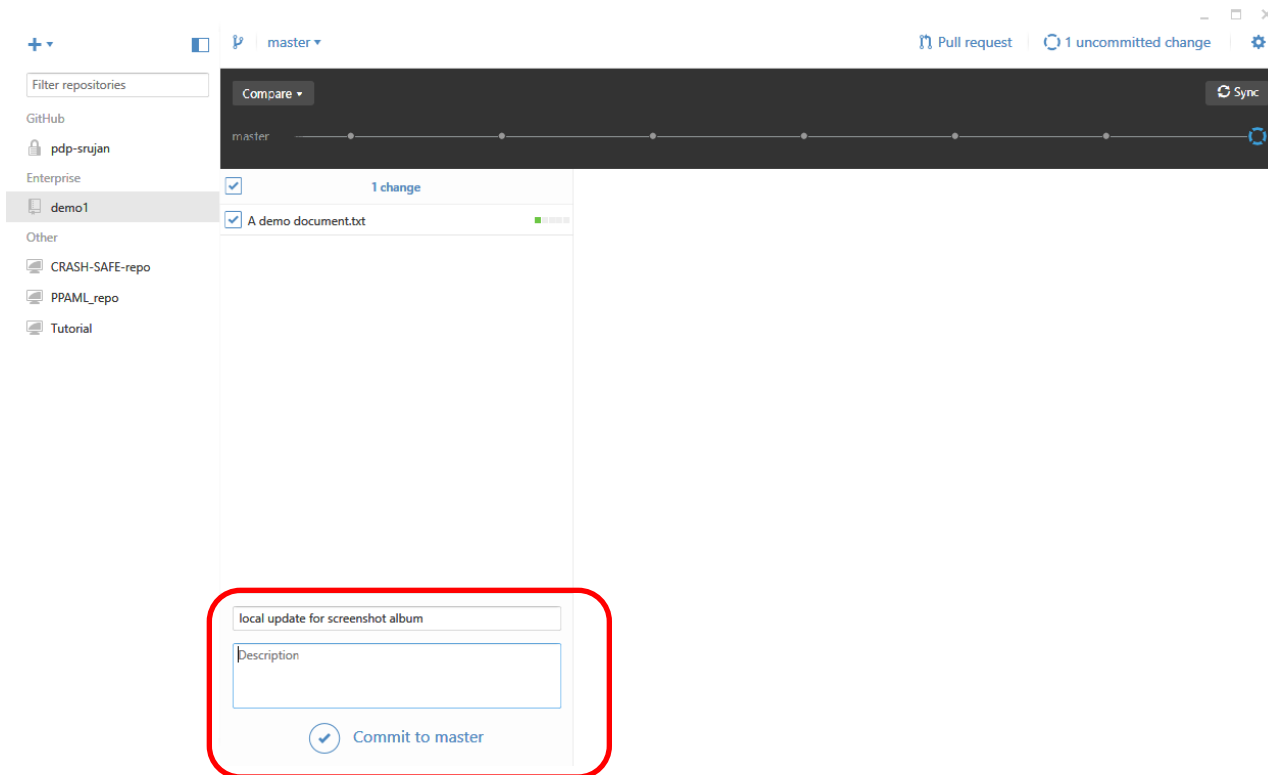
Now let's work on our file

- Now the screen shows an uncommitted change.



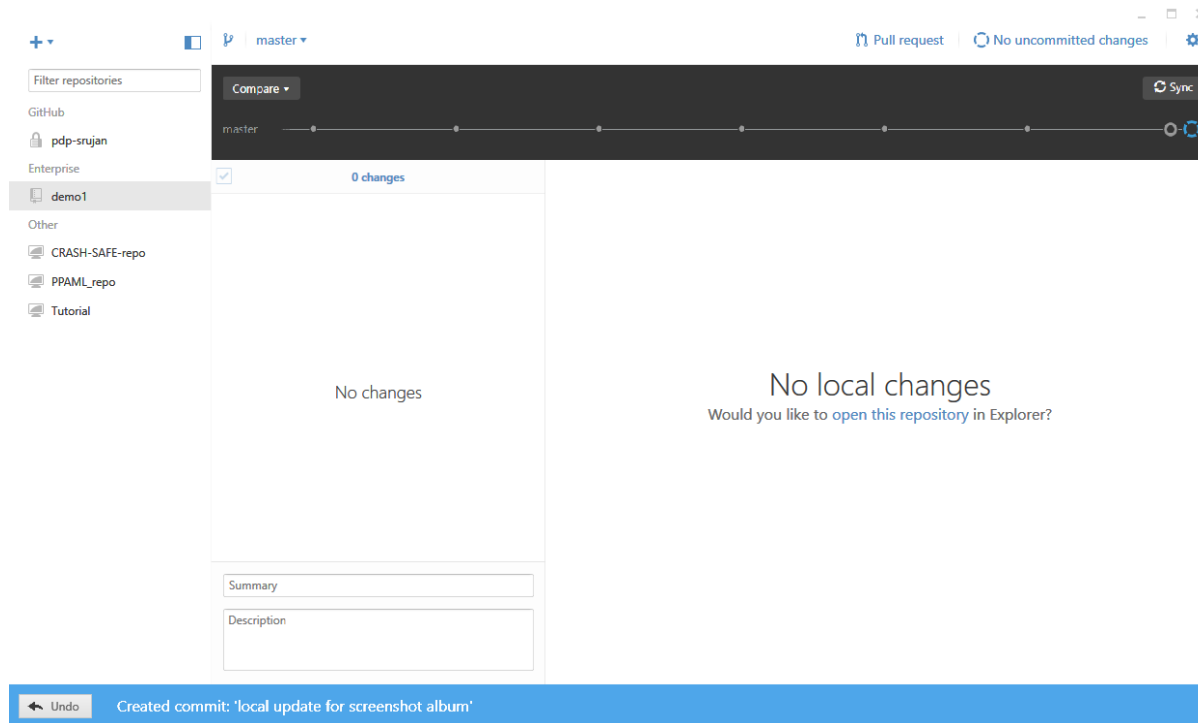
Next, we commit our work

- We write a commit message. Then we'll click on “Commit to Master”



Here's what you'll see after a commit

- Now it says “No uncommitted changes” again.
- You can also undo the commit if you want.



Be sure to record the commit SHA

- Click on the open circle to see what was in your commit, and to record the commit SHA. Here's that screen again:

The screenshot shows the GitHub interface for a repository named 'demo1'. The commit history is displayed on the left, and the details of the selected commit 'Update A demo document.txt' are shown on the right. A red arrow points to the commit SHA '2428503' in the commit details. The diff shows changes to 'A demo document.txt' with lines 5 through 9. Change 8 is highlighted in green.

```
@@ -5,5 +5,5 @@ Change 4
5 5 Here is change 5, made from a new branch
6 6 Change 6, made from another new branch
7 7 change 7, made from server. Will GHFW flag this?
8 -
+ change 8, made from server.
9 9
```

Be sure to sync!!!

- Your work is not saved on the server until you sync.

The screenshot shows the GitHub web interface for a repository. The top navigation bar includes a 'Pull request' button, a 'No uncommitted changes' indicator, and a 'Sync' button. The 'Sync' button is highlighted with a red box and a red arrow pointing down to it. The main content area shows a commit history on the left and a diff view on the right. The diff view shows a file named 'A demo document.txt' with several lines of code. The most recent change is highlighted in green and labeled '+ change 8, made from server.'.

Filter repositories

Compare

master

Update A demo document.txt
7 days ago by wand

testing to see if GHFW detects update.
7 days ago by wand

change from branch local1
7 days ago by Mitchell Wand

commit in new-branch-1
7 days ago by wand

change 4: from local
7 days ago by wand

Make a change from the browser
7 days ago by wand

a test change
7 days ago by wand

Initial commit
7 days ago by wand

Added .gitattributes
7 days ago by wand

Update A demo document.txt
wand 2428503

GitHub Enterprise Revert Collapse all

A demo document.txt

```
@@ -5,5 +5,5 @@ Change 4
5 5 Here is change 5, made from a new branch
6 6 Change 6, made from another new branch
7 7 change 7, made from server. Will GHFW flag this?
8 -
+ change 8, made from server.
9 9
```

Submit a Work Session Report

- At the end of your work session, submit a work session report via the web.
- The URL for the work session report will appear in each problem set.
- The report will ask for the SHA of your last commit. You can get this from the Github Desktop, as we've shown you.

Work Session Report (PS 01)

In order to keep track of the time you spend on each question, we ask you to fill out this form at the end of each work session. If you work on multiple questions during a single work session, please fill out separate reports for each question.

NOTE: we will have a fresh form with a fresh URL for each problem set.

* Required

Your CCS email address *

Example: shriram@ccs.neu.edu

Which question were you working on during this session? *

If you worked on more than one question, please fill out a separate report for each question.

- Question 1 (distance-to-origin)
- Question 2 (string-first)
- Question 3 (image-area)
- Question 4 (string-insert)
- Question 5 (string-delete)

How many hours did you work during this session? *

You need only give the time up to the nearest 15 minutes or so. Ignore the "seconds" field.

Hrs : Mins : Secs

Did you commit AND PUSH your work at the end of this session? *

Remember to commit your work and push it to [github.ccs.neu.edu](https://github.com/ccs.neu.edu) at the end of each session. If you have not done so, do it now.

- Yes
- No

Please record the first six characters of the commit SHA

Sample answer: ec633f

Summary

- In this lesson you have learned
 - that git creates a mini-filesystem in your directory
 - what commit, push, pull, and sync do
 - the elements of the basic git workflow
 - how git allows you to work across multiple computers
 - how git allows you and a partner to work together