CISC 3115 TY3

C02c: Programming Environment

Hui Chen

Department of Computer & Information Science

CUNY Brooklyn College

Outline

- Last class (08/28)
  - Get around in Operating Systems
    - Unix-like: Unix, Linux, OS X
    - Windows
  - Terminal and command line
  - JRE and JDK
- Just discussed
  - Authoring Java programs
  - Compiling and running Java programs
  - CodeLab and Blackboard online exercise
- Git and individual & team assignment submission
- Assignments

Git and Github Classroom

- The instructor uses Github Classroom to manage team work, and weekly programming assignments not in CodeLab

Git

- Version Control System (VCS)
- Source Code Management system (SCM)

Version Control System (VCS)

- Why do we need it?
  - https://stackoverflow.com/questions/1408450/

"

Have you ever:

Made a change to code, realised it was a mistake and wanted to revert back?
Lost code or had a backup that was too old?
Had to maintain multiple versions of a product?
Wanted to see the difference between two (or more) versions of your code?
Wanted to prove that a particular change broke or fixed a piece of code?
Wanted to review the history of some code?
Wanted to submit a change to someone else's code?
Wanted to share your code, or let other people work on your code?
Wanted to see how much work is being done, and where, when and by whom?
Wanted to experiment with a new feature without interfering with working code?

Working with VCS

- VCS provides a "centralized" location to store project files
  - Versioned code, configuration files, build scripts …
- VCS tracks each contributors' individual changes
- VCS helps prevent concurrent work from conflicting

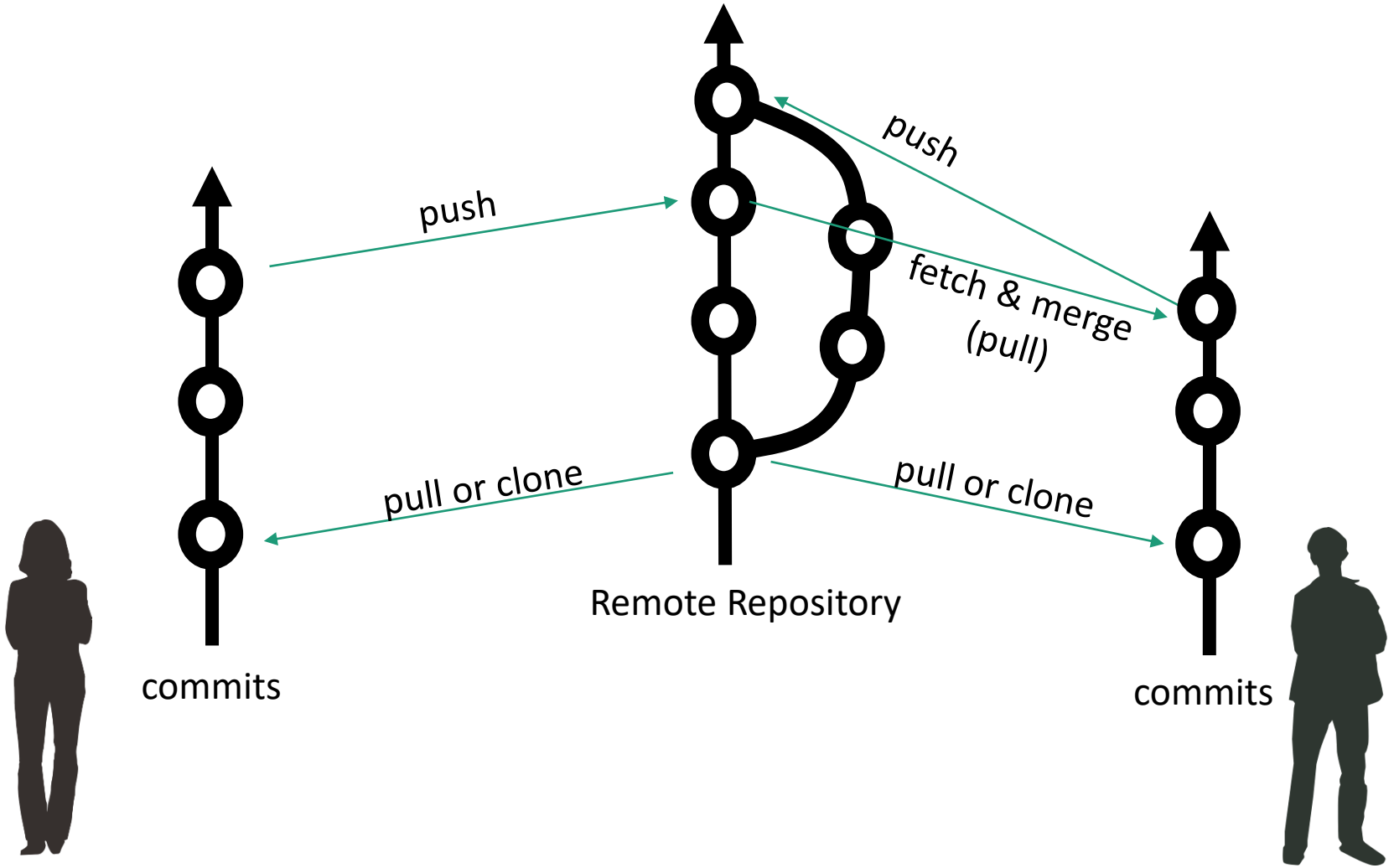
Basic VCS Operations

- Check out/update: copying the repository to the machine you are working at
- Check in/Commit: copying the changes you made to the repository and creating a new version
- Branch: create a new "child" development from a state of the repository

Distributed Version Control

- Git is a distributed version control system
- Possible to commit locally without upsetting the others
- Allow more flexibility and support different kinds of workflow

Example: Distributed Workflow



Merge Conflict

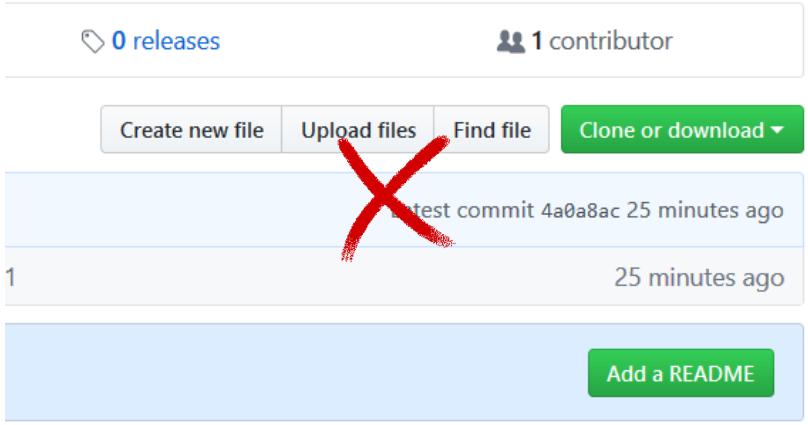
- A merge conflict may occur, e.g., when two branches contains edits to the same file
- Using this reference, see if you can resolve a merge conflict
  - https://help.github.com/articles/resolving-a-merge-conflict-using-the-command-line/.

A Simple Git Workflow

- Assume without merging
  - git clone … (clone the remote repository to your computer)
  - … Do something
  - git add … (Add file contents to the index and to prepare the content staged for the next commit)
  - git commit (Stores the current contents of the index in a new commit along with a log message from the user describing the changes.)
  - git push (Updates remote repository using local repository)
  - git pull (Updates remote refs using local refs, while sending objects necessary to complete the given refs. Attempt this when git push fails; do it before you start working …)
  - … Do something
  - git add …
  - git commit …
  - git push …

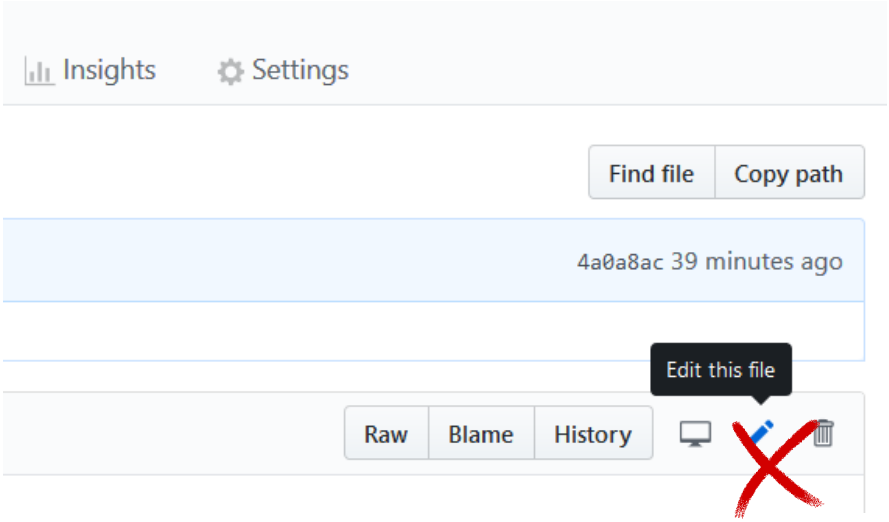
Add Files to Git Repository

- Hey, there is an "upload files" button, I am going to …
- **Don't use it. Under no circumstances, you should use it in this class**



Edit Files in Git Repository

- Hey, there is an "edit this file" button, I am going to …
- **Don't use it. Under no circumstances, you should use it in this class**



In-Class Exercise C03b-1

- Warmup exercise for an individual

In-Class Exercise C03b-2

- Warmup exercise for a team/group
- Now is the time to complete the "Team Composition form"

Questions?

- These are difficult. I have many questions …
- There are too many exercises … (you and you team should complete them in your own time)