Source Control 101

Question:

If a team of developers is collaborating on a project why can't you use something like Dropbox to keep in sync?

Question:

If a team of developers is collaborating on a project why can't you use something like Dropbox to keep in sync?

Answers:

- If two people are editing different files at once it is likely to break the code
- If two people edit the same file it is easy for one save to destroy changes made by an earlier save

Source control is syncing help for developers

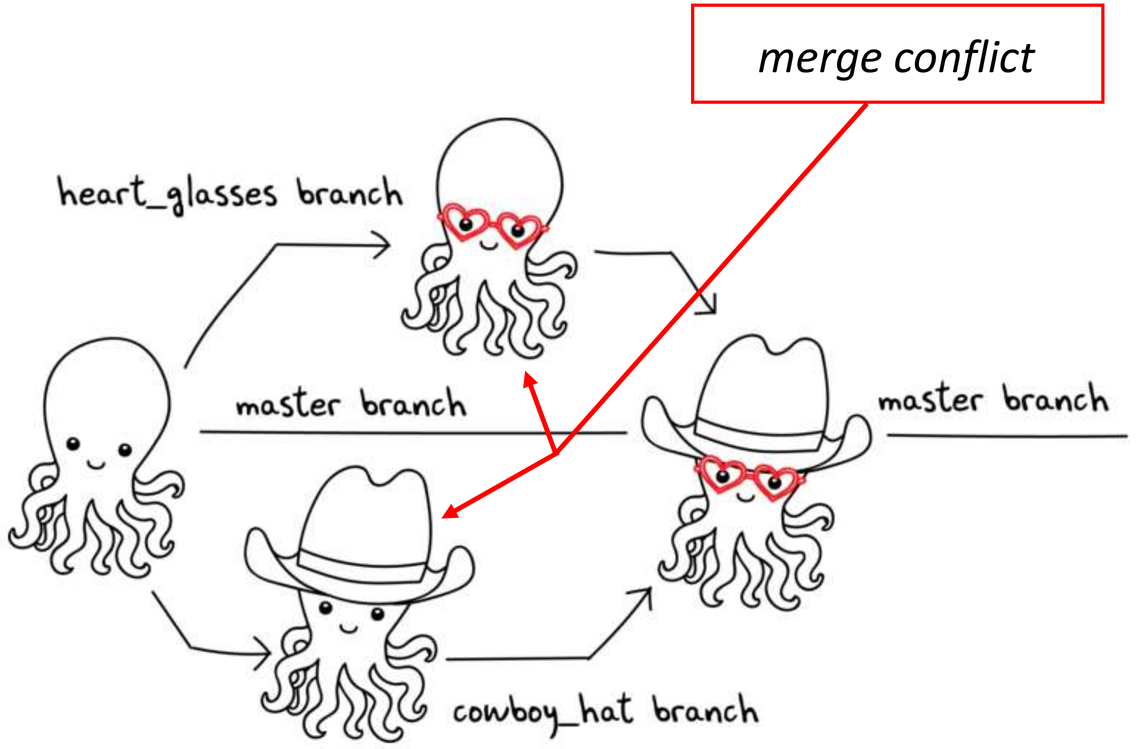
- Your changes are broken into "chunks" called commits that can change multiple files at one time on a local copy
- You only sync with the shared repository when you intend to
- Source controls system allows you to go back to any historical version
- The source control system will prevent you from accidentally destroying someone else changes by overwriting files they changed

Git *basic* workflow

1. Before you start work, you PULL, syncing your local system with any changes other developers have made to the shared repository since you last worked
2. On your local copy, you add some code, make a feature, etc.
3. When you are finished, you combine (ADD) all your changes into a COMMIT and PUSH that commit to the shared repository

Merge conflicts

A merge conflict happens when two developers edit the same file at (approximately) the same time



Git *more complicated* workflow

1. Before you start work, you PULL, syncing your local system with any changes other developers have made to the shared repository since you last worked
2. On your local copy, you add some code, make a feature, etc.
3. You *try* to COMMIT and PUSH, but it fails because your code was not up to date
   - On the PUSH, git usually gives an error reading: ''not fast forward"
4. So you PULL again
5. On your local system, git potentially marks several of the code files to be in conflict, you fix the conflicts in your code by making more edits
6. You test your code
7. You stage the changed files (ADD)
8. You COMMIT and PUSH

Fixing conflicts – what it looks like

SomeFile.java

```
SomeCode
>>>>>>>>> HEAD

YourCode();
==========

TheirCode();
<<<<<<<<<<
MoreCode
```

- You will see eclipse errors – do not get weirded out
- Realize that you are merging two branches of the code
- Edit the code to fix the problems
- Do not forget to test

To Reduce the Occurrence of Conflicts

- Always PULL before you begin coding
- Quickly COMMIT/PUSH after you finish
- Pair program on one computer so each member of the team is not modifying the same code at the same time

What Is Pair Programming?

- Two programmers work side-by-side at a computer, continuously collaborating on the same design, algorithm, code, and/or test
- Enable the pair to produce higher quality code than that produced by the sum of their individual efforts

Pair Programming

- Working in pairs on a single computer
  - The *driver*, uses the keyboard, talks/thinks out-loud
  - The *navigator*, watches, thinks, comments, and takes notes
  - Person who really understands should start by navigating ☺

- For hard (or new) problems, this technique
  - Reduces number of errors
  - Saves time in the long run

Pair programming video

- https://www.youtube.com/watch?v=rG_U12uqRhE

Practice

- With your team, follow the instructions to get your repo setup
- https://ada.csse.rose-hulman.edu/201920-csse220/arcade-game-teamXX/blob/master/GitlabInstructions.md replacing **teamXX** with your team info
- You will submit your ArcadeGame from your repo