# Why Git and GitHub?

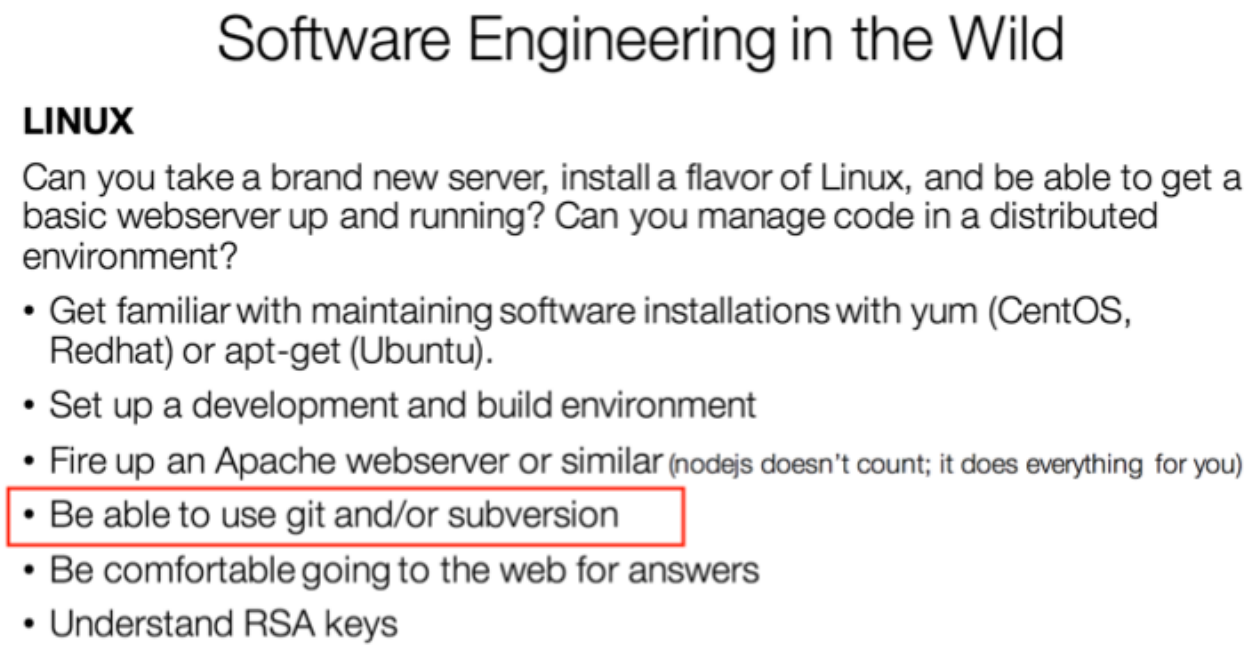
Ming Chow
Senior Lecturer, Tufts University Department of Computer Science
mchow@cs.tufts.edu
Twitter: @0xmchow
SIGCSE 2018

# Why Git?

- Keep *track* of changes and work
- *Documenting* changes and *communicating* changes to others (e.g., in a team)
- *Undo* changes
- *Collaboration*
- A basic skill in tech

# “An Industry Guide to Becoming a Software Engineer"

- Presented by Bill Langenberg, Technical Manager at TripAdvisor
- Slides: https://tuftsdev.github.io/WebProgramming/notes/blangenberg.pdf



Software Engineering in the Wild

**LINUX**

Can you take a brand new server, install a flavor of Linux, and be able to get a basic webserver up and running? Can you manage code in a distributed environment?

- Get familiar with maintaining software installations with yum (CentOS, Redhat) or apt-get (Ubuntu).
- Set up a development and build environment
- Fire up an Apache webserver or similar (nodejs doesn't count; it does everything for you)
- Be able to use git and/or subversion
- Be comfortable going to the web for answers
- Understand RSA keys

# Why Git? (continued)

- The following question was deleted from StackOverflow as "primarily opinion-based": *Most CS programs these days do not teach skills such as source control, configuration management, integration (and continuous integration), code readability (AKA how to comment correctly), programming methodologies, bug tracking. These topics are considered "easy enough" to be taught on-the-job (OTJ), even though mastering them can be very complex. Should these skills be taught in universities? Can a real-world programmer really do without these? Is it sufficient to learn them OTJ, as part of a first-year programming experience?* https://stackoverflow.com/questions/1117018/should-universities-be-teaching-scm-methodology-skills/1117285#1117285
- In 2009, Norman Ramsey responded: "*Universities should teach **distributed source control** starting with the first programming course. It is a fine way to distribute code to students; they can get bug fixes easily. In later courses they can use it to help support pair programming. We should treat it as easy, make it a habit, and not make a big fuss about it.*"
  - Full text: https://www.cs.tufts.edu/~nr/students/scm.html

# Why GitHub?

- Network effects
- Social and code review capabilities
- De facto resume / portfolio

# FW: Feedback from an employer re: Github accounts

Academic x

**Kahan, Robin R.** 11/10/14

to Ming

I know you have your students use a Github account to showcase their work. I thought maybe through your Piazza site though we could print this commentary from an employer to ensure that students see its importance. I will make sure I am checking resumes for the Github

Employer Feedback from ETCIC

We didn't observe any candidates that had a github repository mentioned on their resume, nor when questioned did any of them have one to offer. If you aren't familiar with github (in addition to being a tool most development teams use internally to manage projects) it is the coders/programmers/developers version of an artists or designers "portfolio".

A verbal interview with a code writer only goes so far, and being able to have a peek at examples of code they have written previously and then shared via github allows hiring teams a much more effective method of evaluation.

I suggested to each candidate that they investigate setting up a github account and post any examples of work there. Even if they are just pet projects or works in progress. We will be paying much more attention to those that follow thru on this.

One candidate mentioned that they used github for a school project, but only privately. We would recommend that in general... sharing public examples of code (github) & any open-source contributions (sourceforge) is one of the best ways for young CS grads to begin to make their mark in this world.

This doesn't mean that we didn't have some good conversations, it just would be an excellent finishing touch for these school programs to encourage their students to participate in the coding community in this way. It wasn't just a gap with Rochester students, we saw this across the board.

Cheers,
Scott

# How is Git and GitHub **Used** in COMP 20: Web Programming at Tufts?

- Since 2012
- Course website: https://tuftsdev.github.io/WebProgramming/
- Required for submitting labs and assignments
  - Private GitHub repository give to students two weeks into the course for lab and assignment submissions
- Required for semester group project
- The first assignment: Portfolio (personal website) https://tuftsdev.github.io/WebProgramming/assignments/a1.html
- The third lab: Practice Using Git and GitHub https://tuftsdev.github.io/WebProgramming/assignments/lab-git.html
- *End of course: private GitHub repo returned to students as learning portfolio to keep*

note

stop following 50 views

Actions

# My Gift to You, Take Ownership of Your Private GitHub Repo (DEADLINE, December 31, 2017)

***[Lab 8 has been graded]***

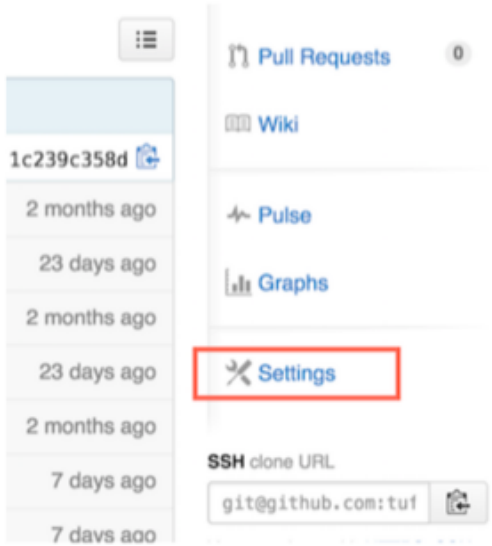
***An important outcome of COMP 20: your portfolio, your private GitHub repository.***

You can now take ownership of your private repo. You can do anything that you want with it: show it off to potential employers and friends, use the material as reference, continue working with it. Once you take ownership of your private repo, it is yours, it will no longer be under "tuftsdev". *I have made everyone's private GitHub repo, public. This is necessary in order to proceed with the transfer of ownership.*

To go ahead and transfer your private GitHub repo to you"

1. Log in to GitHub

2. Go to the "Settings" of your private repo (or https://github.com/tuftsdev/comp20-*****/settings --replace the ***** with your username).




3. Under "Danger Zone", click on "Transfer Ownership" and type in the name of your repo (e.g., "comp20-******"). *I cannot a transfer repo directly to you, you must initiate the request yourself!*

# How is Git and GitHub **Taught** in COMP 20: Web Programming at Tufts?

- *The goal is to get into the **habit** of using revision control, not mastery*
- *Students are thrown into the wolves*
- Focus on the basics: add, commit, push, pull
- Tutorials:
  - The third lab: Practice Using Git and GitHub https://tuftsdev.github.io/WebProgramming/assignments/lab-git.html
  - Git cheat sheets
  - My Asciinema: https://asciinema.org/a/35316
  - Connor Taylor's tutorial on Git: https://tuftsdev.github.io/WebProgramming/notes/git-ctaylor.pdf
    - Connor Taylor (Class of 2015) was the head TA for the course in 2014-2015

# Habit, Not Mastery

Source:
https://twitter.com/ErrataRob/status/707327952158121984

# Challenges

- 20% of the questions from students throughout the course are Git-related
  - Permission denied (SSH key) issues
  - Merge conflicts
  - Students didn’t read the error message carefully to do a `git pull` first
- Students go on and use Git and GitHub in other courses, which becomes a double-edge sword…

github and student work Academic x

**Norman Ramsey**

to mchow,

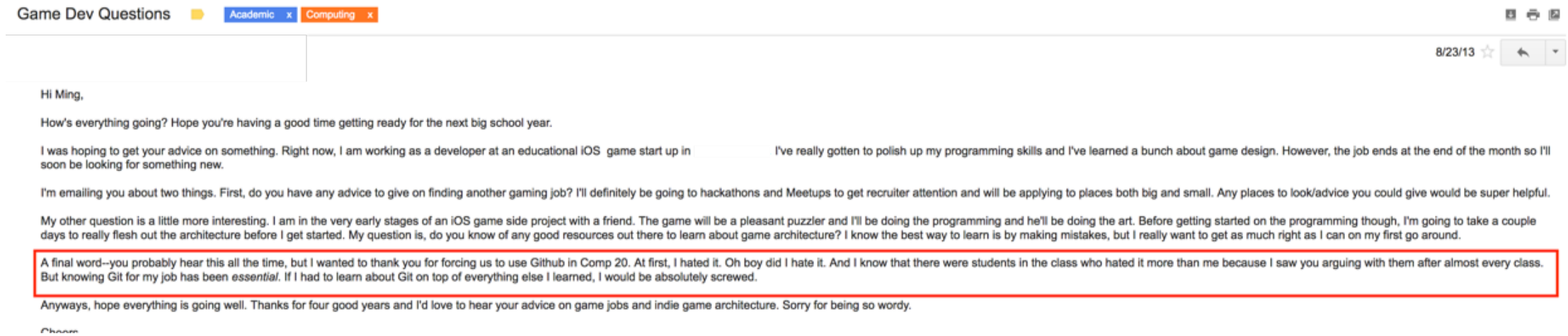
Ming,

A couple of my less thoughtful students have created *public* github repositories for their comp 40 projects. I'm sure that I don't need to explain to anyone why it's a bad idea for students to make their source code public. On the other hand, I really would like to encourage students to use git for their projects, and I think git is an ideal tool to enable collaboration among students who are working in groups.

I know you are promoting the use of github in your classes. Have you found a way to resolve this problem? If so, how? If not, do you have any suggestions regarding comp 40 policy?

...

# Students and Alumni Stories



Game Dev Questions Academic x Computing x

8/23/13

Hi Ming,

How's everything going? Hope you're having a good time getting ready for the next big school year.

I was hoping to get your advice on something. Right now, I am working as a developer at an educational iOS game start up in I've really gotten to polish up my programming skills and I've learned a bunch about game design. However, the job ends at the end of the month so I'll soon be looking for something new.

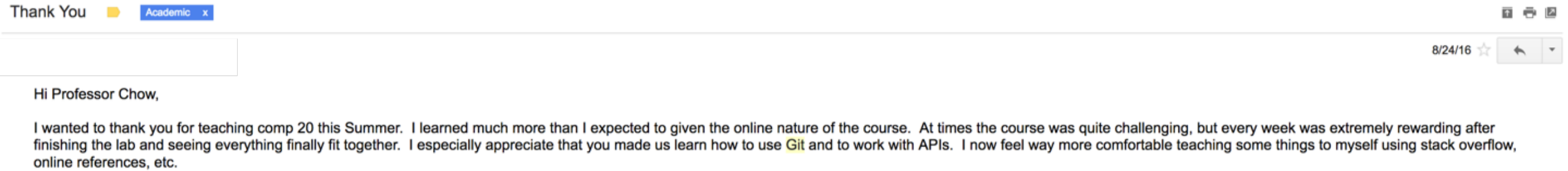
I'm emailing you about two things. First, do you have any advice to give on finding another gaming job? I'll definitely be going to hackathons and Meetups to get recruiter attention and will be applying to places both big and small. Any places to look/advice you could give would be super helpful.

My other question is a little more interesting. I am in the very early stages of an iOS game side project with a friend. The game will be a pleasant puzzler and I'll be doing the programming and he'll be doing the art. Before getting started on the programming though, I'm going to take a couple days to really flesh out the architecture before I get started. My question is, do you know of any good resources out there to learn about game architecture? I know the best way to learn is by making mistakes, but I really want to get as much right as I can on my first go around.

A final word--you probably hear this all the time, but I wanted to thank you for forcing us to use Github in Comp 20. At first, I hated it. Oh boy did I hate it. And I know that there were students in the class who hated it more than me because I saw you arguing with them after almost every class. But knowing Git for my job has been *essential*. If I had to learn about Git on top of everything else I learned, I would be absolutely screwed.

Anyways, hope everything is going well. Thanks for four good years and I'd love to hear your advice on game jobs and indie game architecture. Sorry for being so wordy.

Cheers



Thank You Academic x

8/24/16

Hi Professor Chow,

I wanted to thank you for teaching comp 20 this Summer. I learned much more than I expected to given the online nature of the course. At times the course was quite challenging, but every week was extremely rewarding after finishing the lab and seeing everything finally fit together. I especially appreciate that you made us learn how to use Git and to work with APIs. I now feel way more comfortable teaching some things to myself using stack overflow, online references, etc.

I'm on the HBase team (big data - woohooo) and so far my tasks have consisted of adding to/modifying fork of Apache HBase (which is one hell of a code base.) I'm learning about NoSQL column-store databases, optimization algorithms as work arounds for NP-complete problems, concurrency, Java8, and distributed systems. It's a lot to learn and feels a little overwhelming at times, but I'm having a wonderful learning experience and at the end of the day I'm so grateful that even if I'm feeling overwhelmed, I have a basic understanding of the tools and skills I need to know in order to actually get to the meat of my tasks. That is, I know how to use git/github, look at other people's commits, create branches, google effectively, use stackoverflow, read documentation, and read other people's code! I mean seriously this stuff is step0 for working here (and my guess is anywhere in industry.) . The other thing that astounds me is if I hadn't taken comp20 and comp120 with you, I would have no concept of a distinction between client and server. It's not an easy mental model at first and I remember since day1 of comp20 you drew that same client-server picture over and over again until it finally clicked with the class.

Post-grad Academic x Computing x

8/4/17

Hey Ming,

Just wanted to give you a quick update and a word of thanks.

After many months of a frustrating job search, I started working at Specifically, I'm part of a small team that builds statistical models for this website. One of my projects is building a model in JavaScript that shows visitors to the site the policy we think they're most likely to want to buy (it's actually a pretty groundbreaking concept in the insurance world – tailoring coverage to the customers' needs and budgets instead of making them pick from our plans. Seems kind of obvious, in my opinion).

While most of my work is done with R and SQL (languages I had never worked with before…), I know for a fact I wouldn't have gotten the job if it weren't for you. Here's way:

I met with the guy who is now my manager for a purely informational talk. I just wanted to learn about the systems he was using and what kinds of stuff he found interesting. We talked a little stats shop, and a lot about tools (the company uses TFS instead of GitHub, etc.). I left, thinking that would be the last time I saw him.

He emailed me a week later asking me to join his team. I was so floored. Everyone else we work with has a masters in stats. I found out later from one of my teammates that the manager said there were two things about me that told him that I was serious and able to learn:

1. I use a non-standard editor (Atom)
2. I was passionate about the virtues of GitHub

Both of these are attributable to you. For about a year before I took comp 20, I hadn't really been coding, and then I needed to I was content using Vim. But turned me onto Atom once he saw me coding for 20. And of course, I had used GitHub before taking your class, but I sure didn't understand Git, and concepts of networking and version control were not high on my list of interests.

So, Ming, I really owe all of that to you. Already I've deployed direct knowledge from Comp 20 in discussions, and I can feel that I've become a more efficient learner and stack overflow user from having begun to tackle web development.

# References


- https://github.com/blog/2343-why-version-control-is-required-for-comp-20-at-tufts-university
- https://clutch.co/it-services/failings-cybersecurity-education-interview-professor-ming-chow