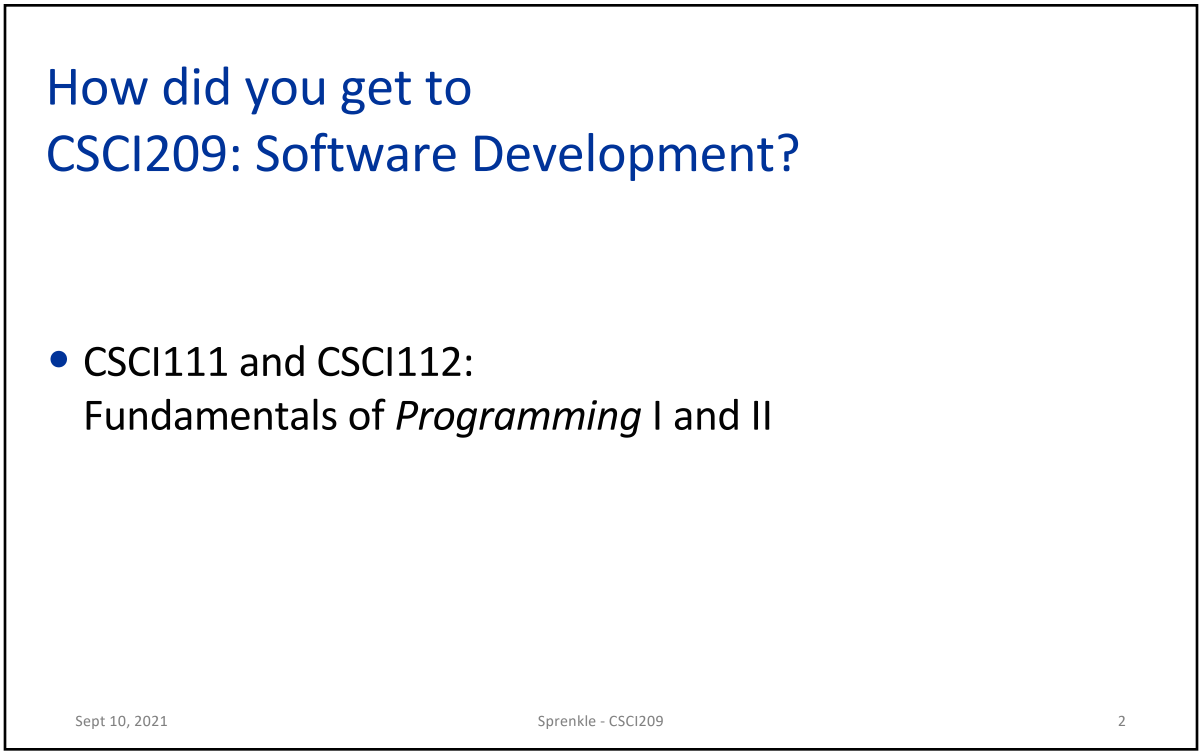
CSCI 209:
Software Development

Sara Sprenkle
sprenkles@wlu.edu

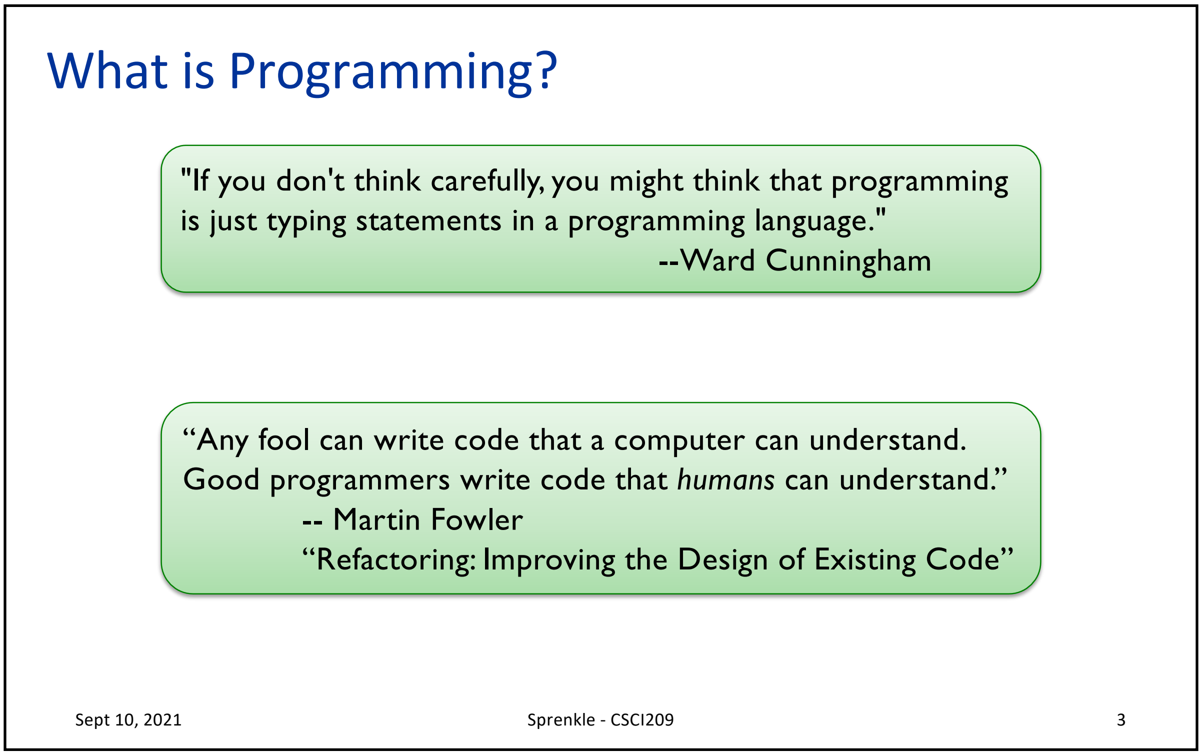
http://cs.wlu.edu/~sprenkle/cs209

https://pollev.com/sprenkle



How did you get to CSCI209: Software Development?

- CSCI111 and CSCI112:
  Fundamentals of *Programming* I and II

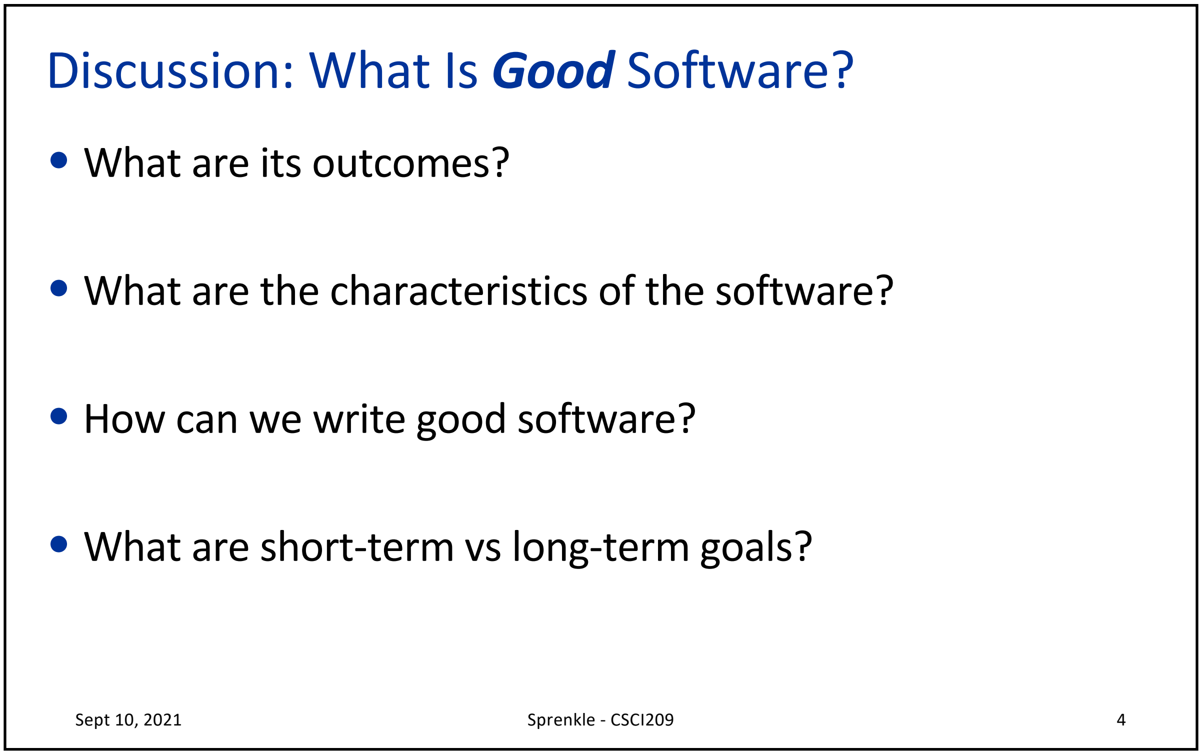


What is Programming?

"If you don't think carefully, you might think that programming is just typing statements in a programming language."
--Ward Cunningham

“Any fool can write code that a computer can understand. Good programmers write code that *humans* can understand.”
-- Martin Fowler
“Refactoring: Improving the Design of Existing Code”

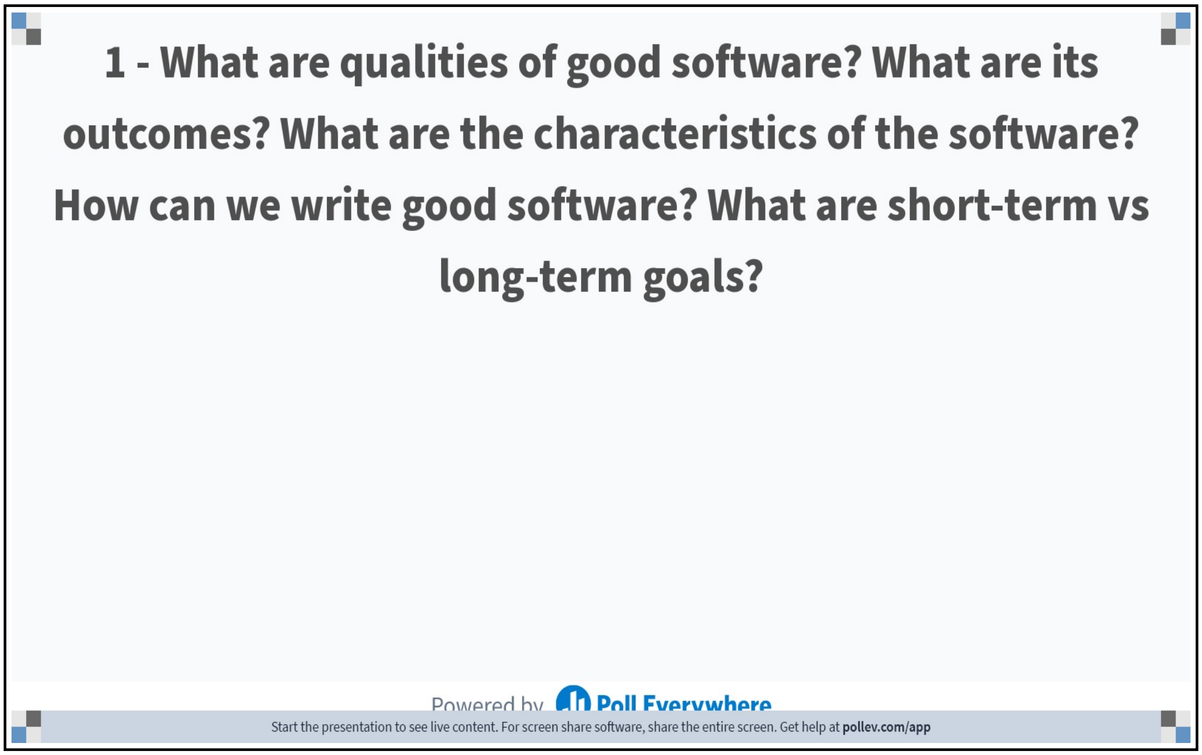
3



Discussion: What Is ***Good*** Software?

- What are its outcomes?
- What are the characteristics of the software?
- How can we write good software?
- What are short-term vs long-term goals?

4



1 - What are qualities of good software? What are its outcomes? What are the characteristics of the software? How can we write good software? What are short-term vs long-term goals?

Powered by Poll Everywhere

Start the presentation to see live content. For screen share software, share the entire screen. Get help at **pollev.com/app**

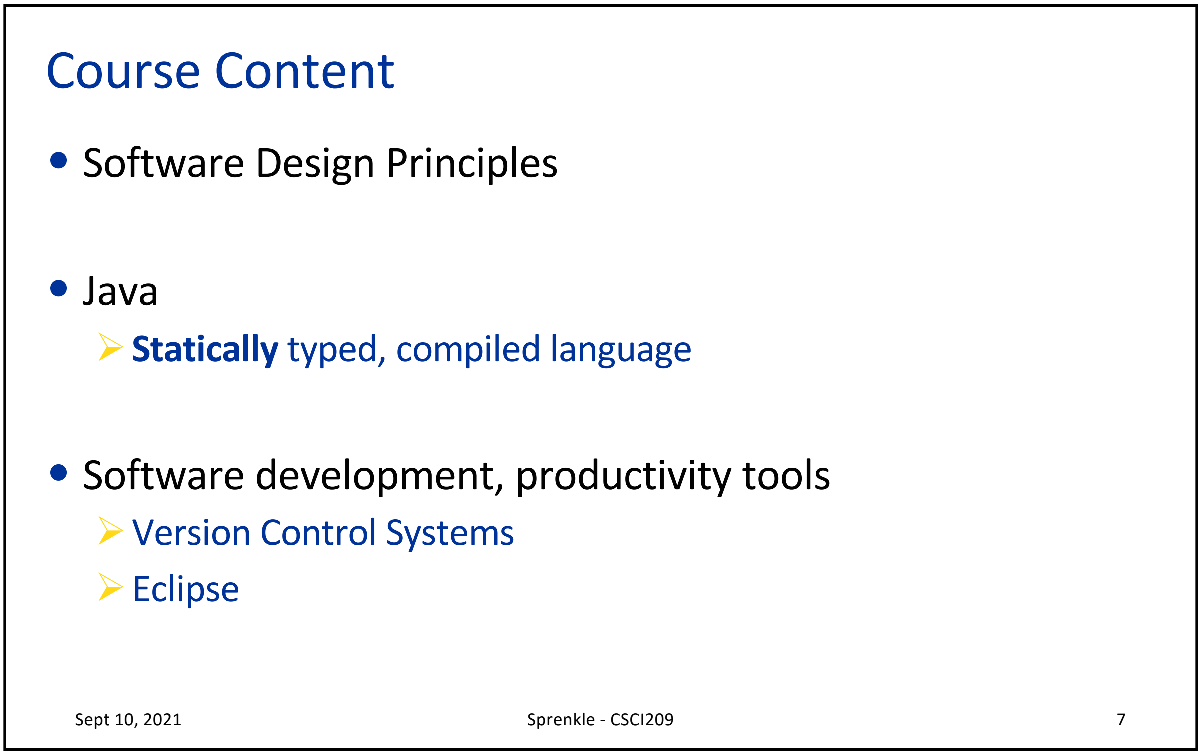
5

Characteristics of ***Good*** Software?

- Free of bugs (does what it is supposed to do)
  - Robust, reliability, stability
- Code is easy to read, extend, maintain
  - Readability, extensibility, maintainability
- Usability: Application is easy to use
- Efficiency
- Scalability

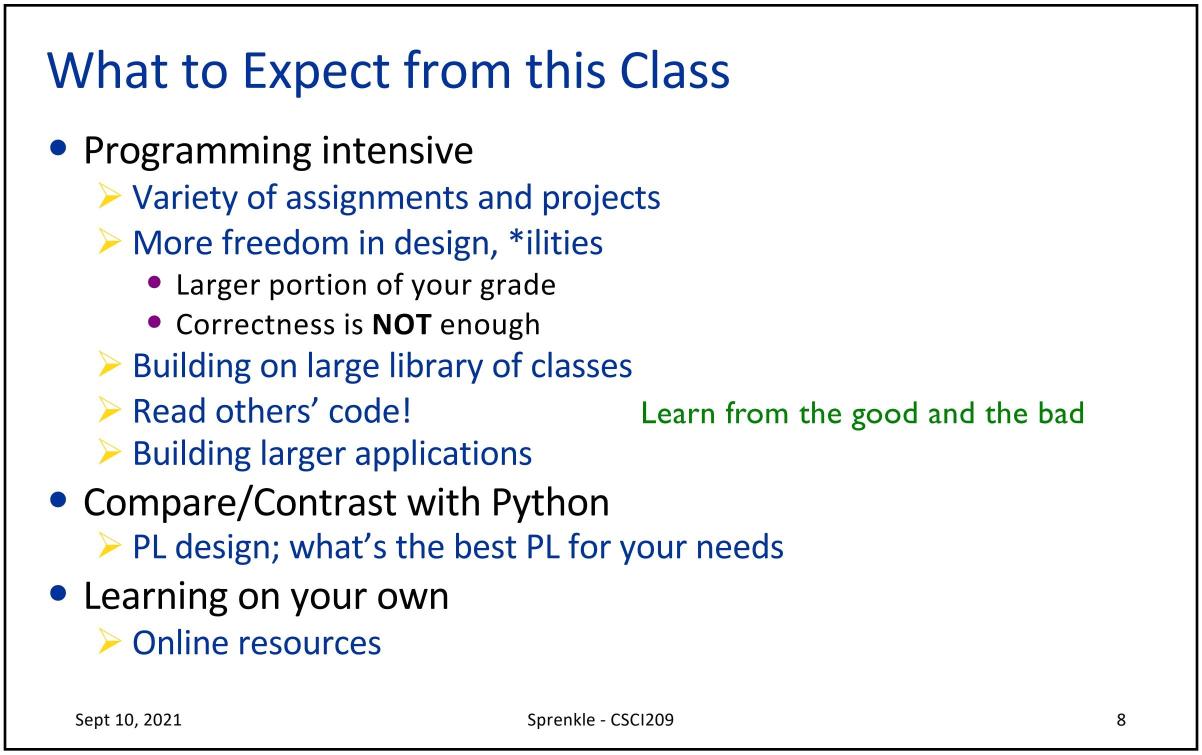
➔ Metrics of quality referred to as the **ilities*

6



Course Content

- Software Design Principles
- Java
  - **Statically** typed, compiled language
- Software development, productivity tools
  - Version Control Systems
  - Eclipse



What to Expect from this Class

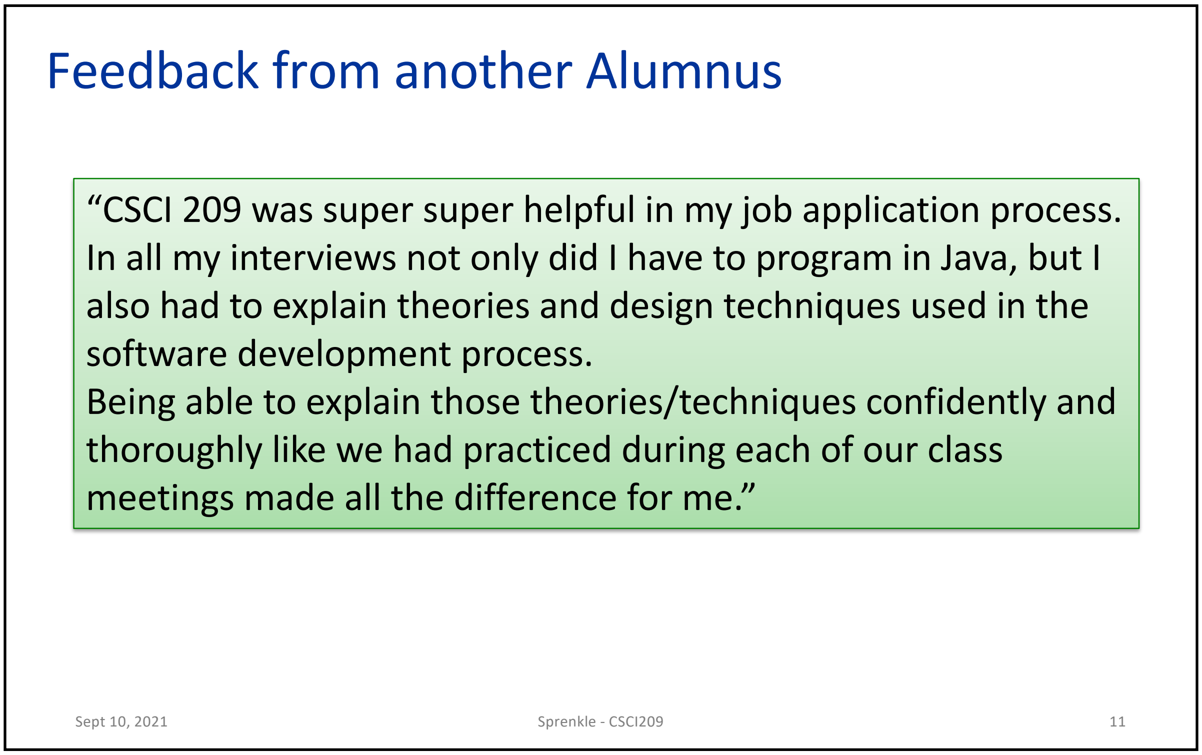
- Programming intensive
  - Variety of assignments and projects
  - More freedom in design, *ilities
    - Larger portion of your grade
    - Correctness is **NOT** enough
  - Building on large library of classes
  - Read others’ code! Learn from the good and the bad
  - Building larger applications
- Compare/Contrast with Python
  - PL design; what’s the best PL for your needs
- Learning on your own
  - Online resources

Learning Objectives

- Discuss software development and practices knowledgably, using appropriate terminology
- Design, implement, test, and document efficient applications of increasing size and complexity
- Understand the designs and implementations of others
- Use a version control system
- Use many of the capabilities of the Eclipse IDE
- Test and debug large applications systematically, using standard tools
- Understand design principles such as DRY and shy
- Discuss the benefits and limitations of a statically typed language

Feedback from an Alumnus

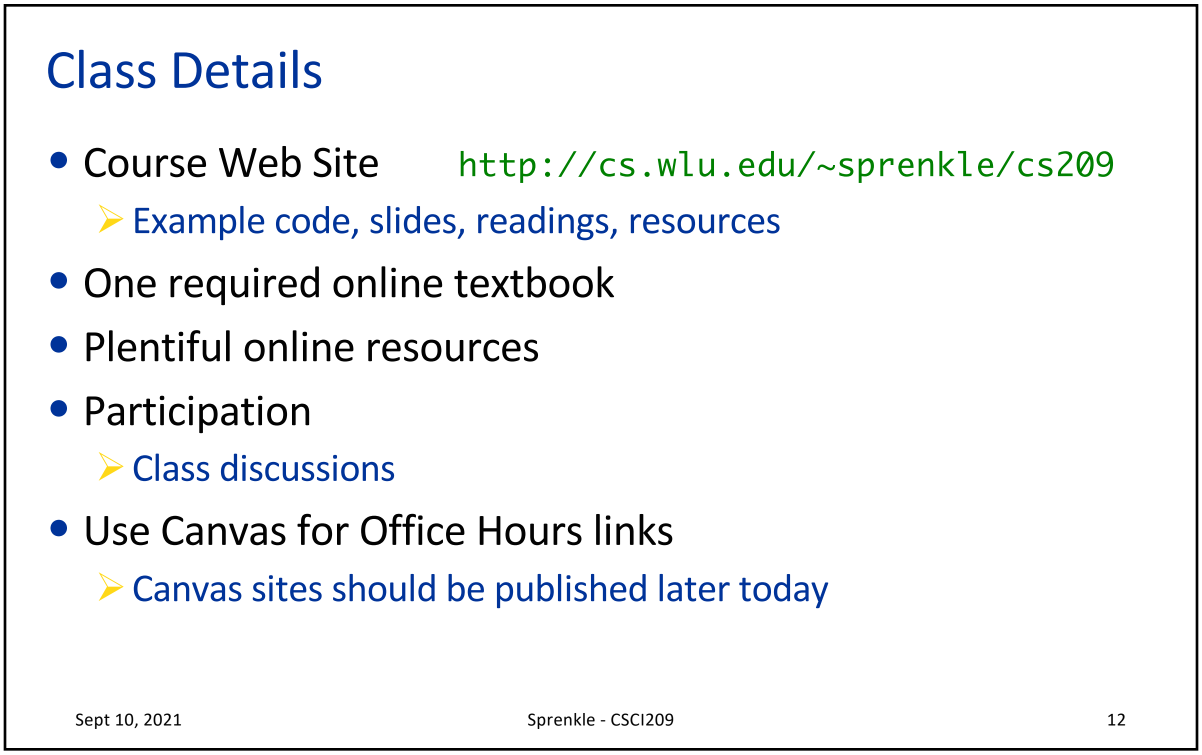
"I am schooling everyone at work on OO design and Java.
Seriously, keep pounding OO design principles in.
It is incredibly practical.
I'm teaching CS majors and Computer Engineering grads about this.
It's crazy how some (good) technical schools don't stress this more."



Feedback from another Alumnus

> “CSCI 209 was super super helpful in my job application process. In all my interviews not only did I have to program in Java, but I also had to explain theories and design techniques used in the software development process.
> Being able to explain those theories/techniques confidently and thoroughly like we had practiced during each of our class meetings made all the difference for me.”

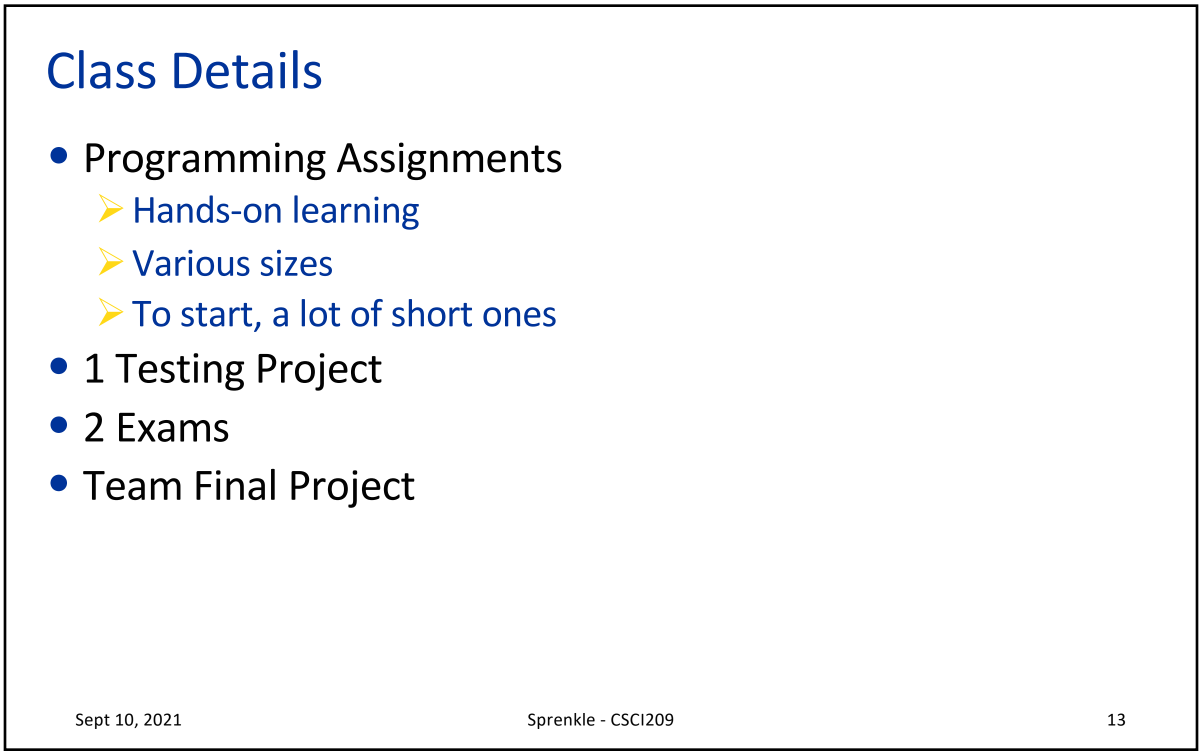
11



Class Details

- Course Web Site `http://cs.wlu.edu/~sprenkle/cs209`
  - Example code, slides, readings, resources
- One required online textbook
- Plentiful online resources
- Participation
  - Class discussions
- Use Canvas for Office Hours links
  - Canvas sites should be published later today

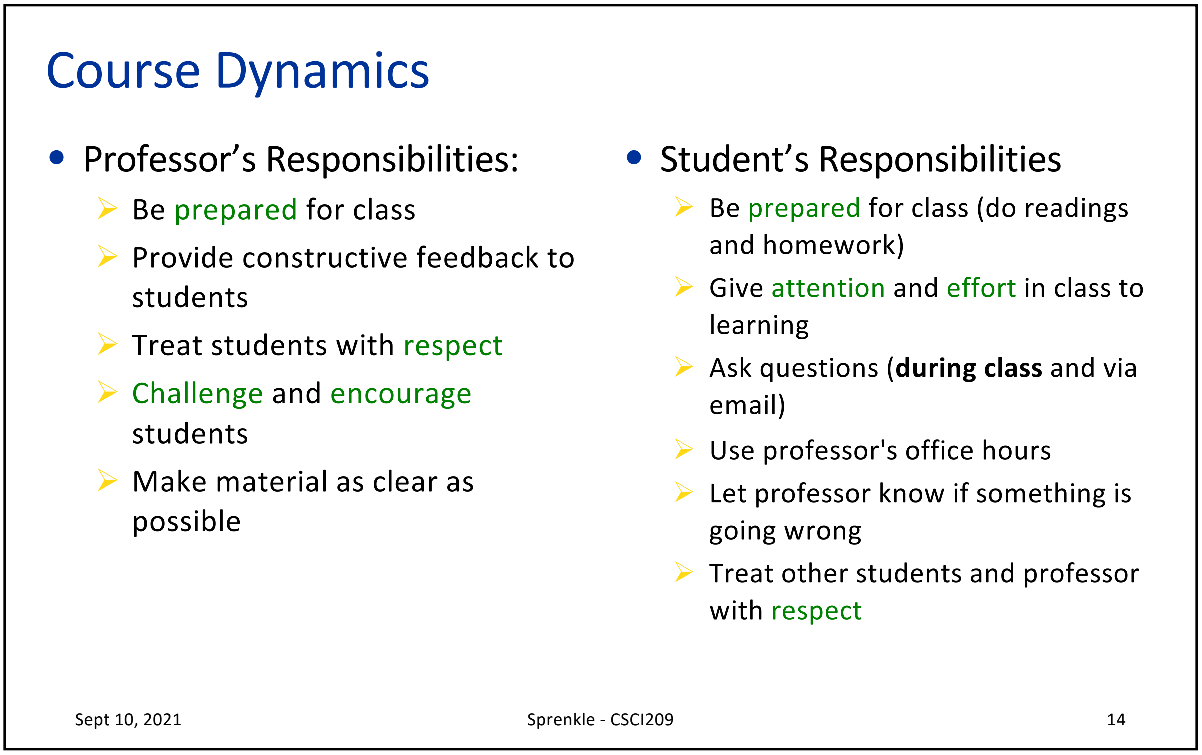
12



Class Details

- Programming Assignments
  - Hands-on learning
  - Various sizes
  - To start, a lot of short ones
- 1 Testing Project
- 2 Exams
- Team Final Project

13



Course Dynamics

- Professor’s Responsibilities:
  - Be prepared for class
  - Provide constructive feedback to students
  - Treat students with respect
  - Challenge and encourage students
  - Make material as clear as possible
- Student’s Responsibilities
  - Be prepared for class (do readings and homework)
  - Give attention and effort in class to learning
  - Ask questions (**during class** and via email)
  - Use professor's office hours
  - Let professor know if something is going wrong
  - Treat other students and professor with respect

14

My Bio

- From Dallastown, PA
- Education: B.S., Gettysburg College; M.S., Duke University; Ph.D., University of Delaware
- For fun: pop culture, gardening, Rockbridge Animal Alliance

July 2021

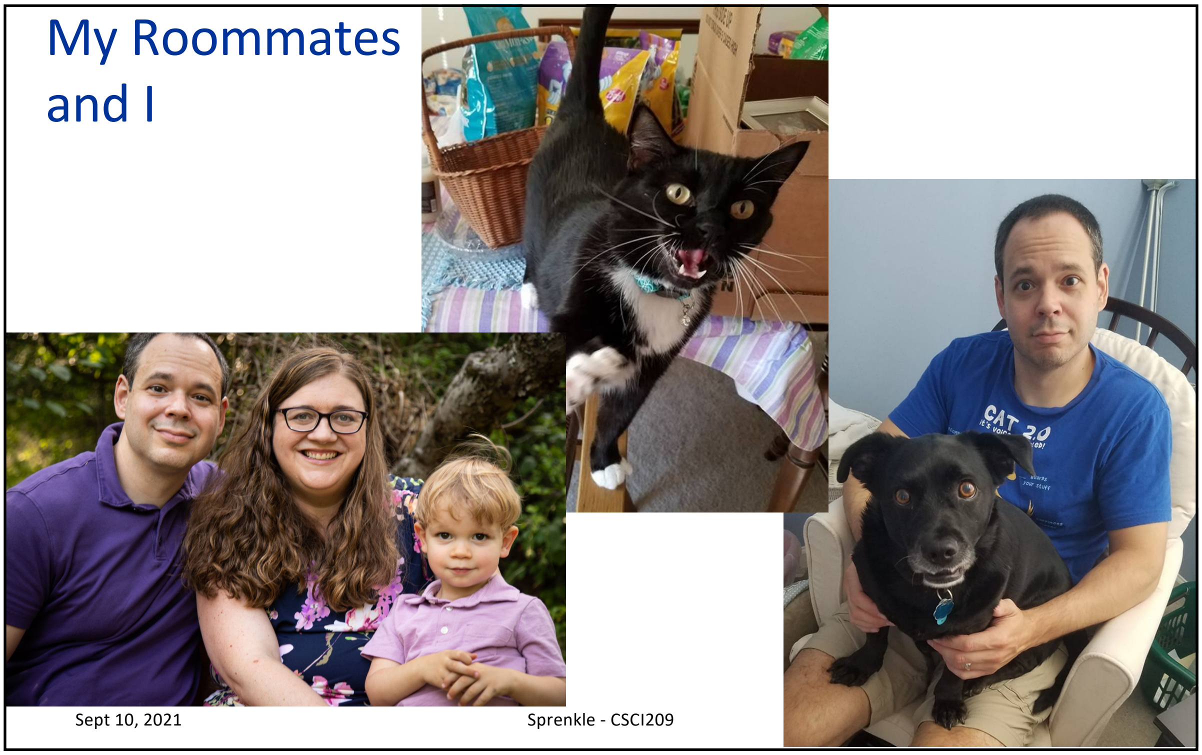
15

October 2019

August 2020

September 2021

16



My Roommates and I

17

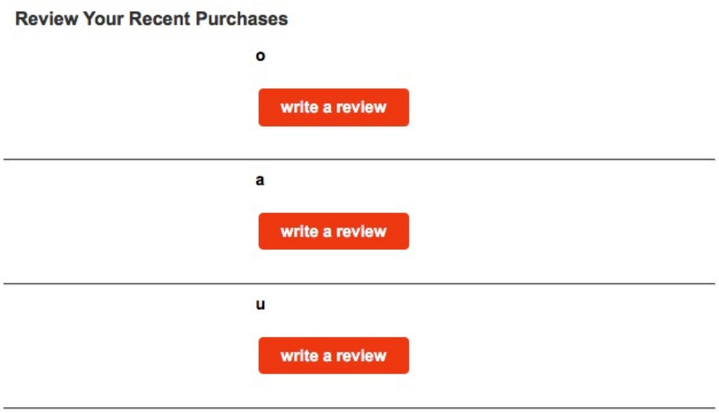
My Research Interests

- General: Software engineering
- Automated testing of web applications
  - Develop algorithms
  - Implement in tools
  - Empirical studies
    - Try ideas out, see what actually happens, analyze
- Subject applications: Ancient Graffiti Project, ChemTutor

18

My Research Interests: Automatically Finding Bugs

- In an email:



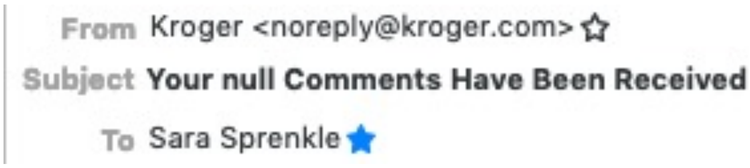
Review Your Recent Purchases

o

a

u

My Research Interests: Automatically Finding Bugs

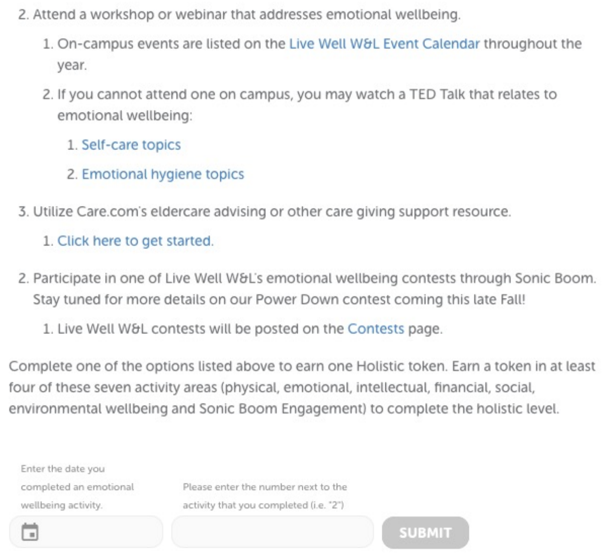


From Kroger <noreply@kroger.com> ☆

Subject **Your null Comments Have Been Received**

To Sara Sprenkle ★

My Research Interests: Automatically Finding Bugs



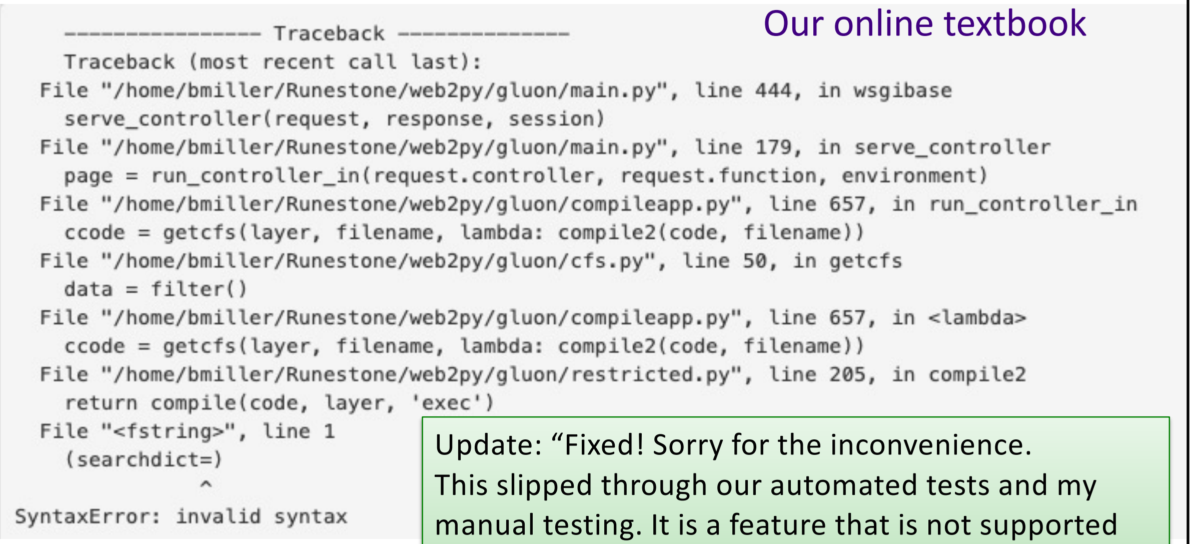
2. Attend a workshop or webinar that addresses emotional wellbeing.
   1. On-campus events are listed on the Live Well W&L Event Calendar throughout the year.
   2. If you cannot attend one on campus, you may watch a TED Talk that relates to emotional wellbeing:
      1. Self-care topics
      2. Emotional hygiene topics
3. Utilize Care.com's eldercare advising or other care giving support resource.
   1. Click here to get started.
2. Participate in one of Live Well W&L's emotional wellbeing contests through Sonic Boom. Stay tuned for more details on our Power Down contest coming this late Fall!
   1. Live Well W&L contests will be posted on the Contests page.

Complete one of the options listed above to earn one Holistic token. Earn a token in at least four of these seven activity areas (physical, emotional, intellectual, financial, social, environmental wellbeing and Sonic Boom Engagement) to complete the holistic level.

Enter the date you completed an emotional wellbeing activity.

Please enter the number next to the activity that you completed (i.e. "2")

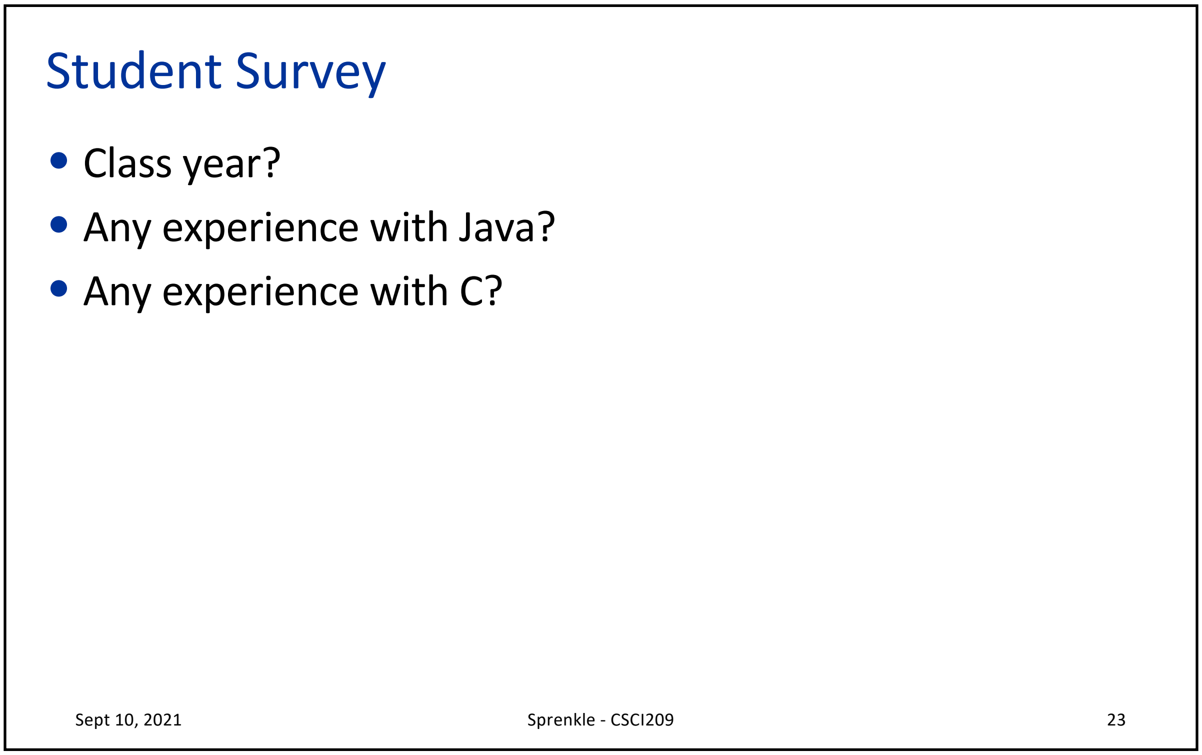
Sometimes, Errors are More Obvious



Our online textbook

```
  ---------------- Traceback --------------
  Traceback (most recent call last):
File "/home/bmiller/Runestone/web2py/gluon/main.py", line 444, in wsgibase
  serve_controller(request, response, session)
File "/home/bmiller/Runestone/web2py/gluon/main.py", line 179, in serve_controller
  page = run_controller_in(request.controller, request.function, environment)
File "/home/bmiller/Runestone/web2py/gluon/compileapp.py", line 657, in run_controller_in
  ccode = getcfs(layer, filename, lambda: compile2(code, filename))
File "/home/bmiller/Runestone/web2py/gluon/cfs.py", line 50, in getcfs
  data = filter()
File "/home/bmiller/Runestone/web2py/gluon/compileapp.py", line 657, in <lambda>
  ccode = getcfs(layer, filename, lambda: compile2(code, filename))
File "/home/bmiller/Runestone/web2py/gluon/restricted.py", line 205, in compile2
  return compile(code, layer, 'exec')
File "<fstring>", line 1
  (searchdict=)
             ^
SyntaxError: invalid syntax
```

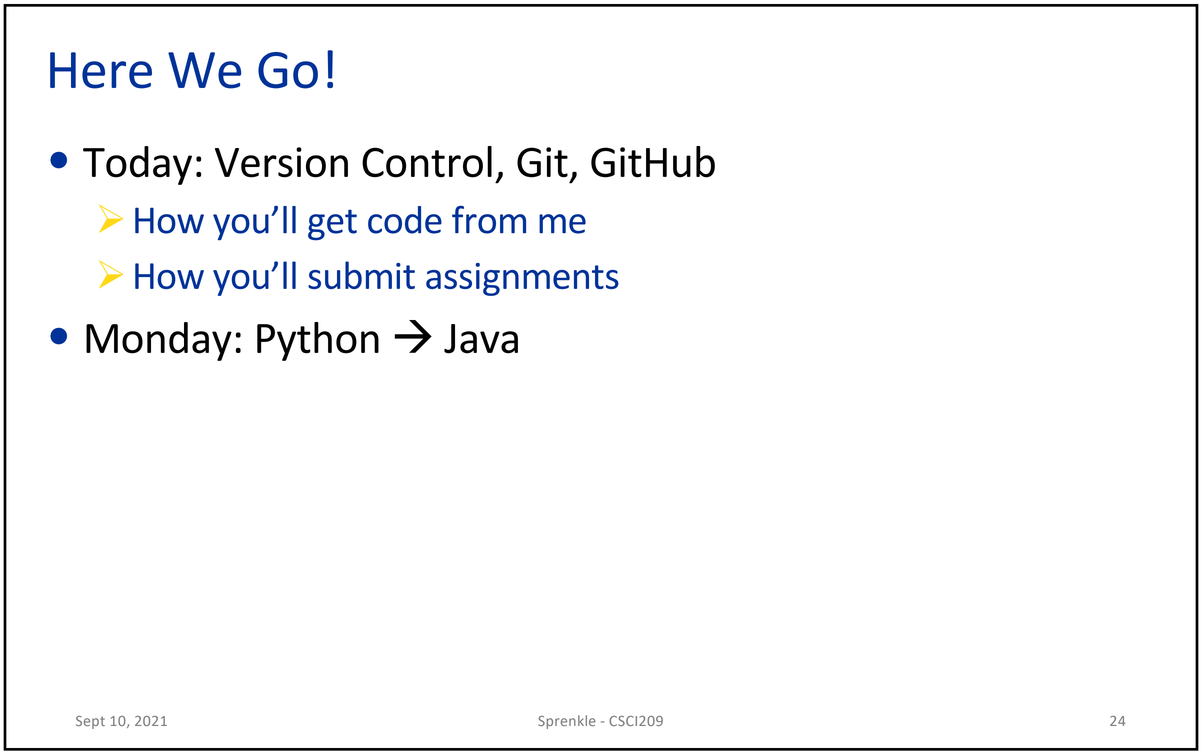
Update: "Fixed! Sorry for the inconvenience. This slipped through our automated tests and my manual testing. It is a feature that is not supported by python3.7 but is supported by python 3.8 and 3.9. Time to get all of Runestone's servers off of 3.7!!"



Student Survey

- Class year?
- Any experience with Java?
- Any experience with C?

23



Here We Go!

- Today: Version Control, Git, GitHub
  - ➢ How you’ll get code from me
  - ➢ How you’ll submit assignments
- Monday: Python → Java

24

VERSION CONTROL: GIT

Motivating Version Control

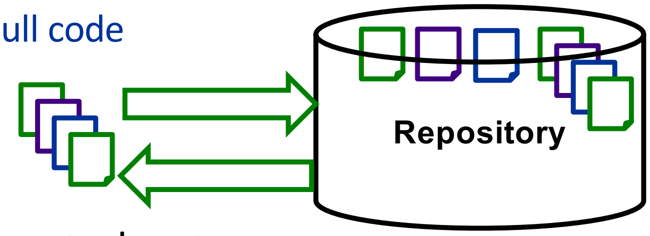
- Do you name your documents like the following?
  - Paper-final.pdf
  - Paper-final2.pdf
  - Paper-final_real.pdf
  - Paper-FINAL-final.pdf
- Do you sometimes break your code so badly and really want to go back to a previous state?
- Do you forget why you made a code change?
- Do you want to just try something out and, if it doesn't work, revert back?

Motivating Version Control: Collaboration

- How do you share documents with teammates?
  - Emailing with conflicting updates?
  - Google Docs/Box Notes → not meant for code
  - Merging contributions
  - Who has the *real* version of the documents?

Version Control Systems Can Help!

- Main idea: ***repository*** holds the code and all changes to it
  - Need to push and pull code to and from the repository



- Centralized version control systems
  - E.g., CVS, Subversion, …
- Distributed version control systems
  - E.g., Git, Mercurial, …

Git & GitHub

- We're going to use Git
  - Distributed version control system
- Our repositories will be hosted by GitHub
  - How you'll get code from me
  - How you'll submit assignments

GitHub's Octocat

Common Git Commands

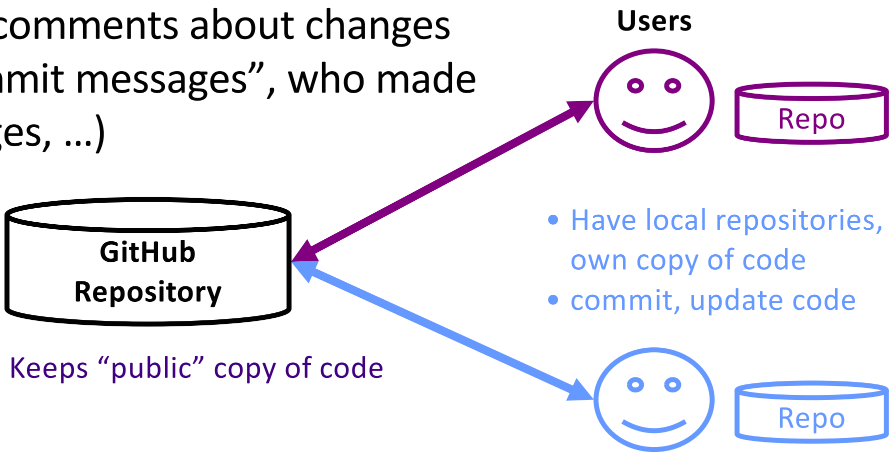
Command	What it does
clone	Clones a repository – sets up your repository so that you can coordinate
add <*file*>	Adds the *file* to the staging area
commit	Commits all the staged files (locally)
push	Push all your changes to the remote → You need your code to be pushed so that I can see it.
branch	List all local branches
branch <*name*>	Creates a new branch named *name*
checkout <*name*>	Switches to the branch named *name*

THIS IS GIT. IT TRACKS COLLABORATIVE WORK ON PROJECTS THROUGH A BEAUTIFUL DISTRIBUTED GRAPH THEORY TREE MODEL.
COOL. HOW DO WE USE IT?
NO IDEA. JUST MEMORIZE THESE SHELL COMMANDS AND TYPE THEM TO SYNC UP. IF YOU GET ERRORS, SAVE YOUR WORK ELSEWHERE, DELETE THE PROJECT, AND DOWNLOAD A FRESH COPY.

https://xkcd.com/1597/

Using Git

- Git is a *distributed* VCS
- ***Repositories*** store all versions of all files, comments about changes ("commit messages", who made changes, …)

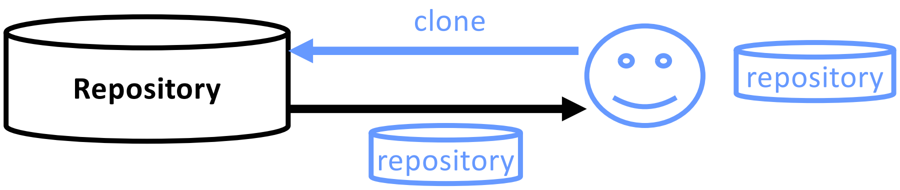


- Have local repositories, own copy of code
- commit, update code

- Keeps "public" copy of code

Using Version Control: `clone`

- To start, need to **clone** the repository

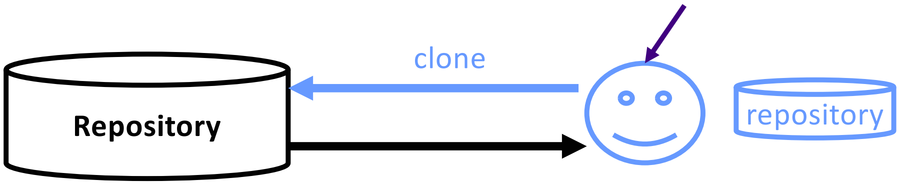


Using Version Control: `clone`

- To start, need to **clone** the repository

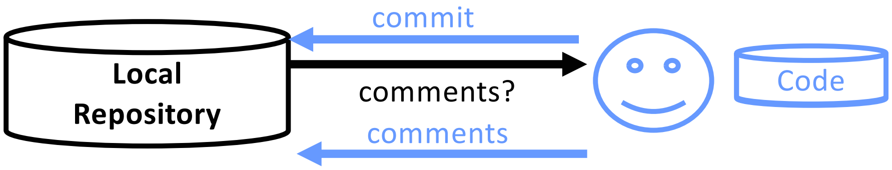
For us,
Repository is on GitHub

This is you!



Using Version Control: `commit`

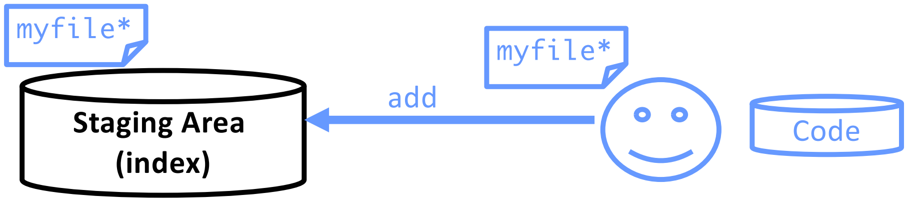
- After you make changes that you want to document, **commit** your version
  - Include comments about what changes you made and ***why***



- Updates each modified file
- Records comments with updated files

Using Version Control: **`add, delete`**

- You need to **add** and **delete** files and directories to the *staging area*, then **commit**

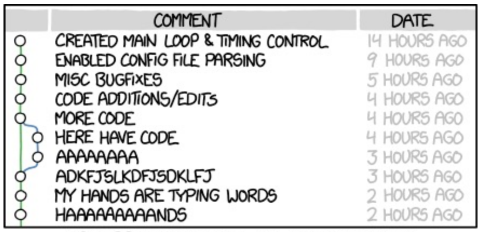


- Marks the files that will be part of the next commit
- When you commit, these files are added to your local repository

- Add, delete files and directories

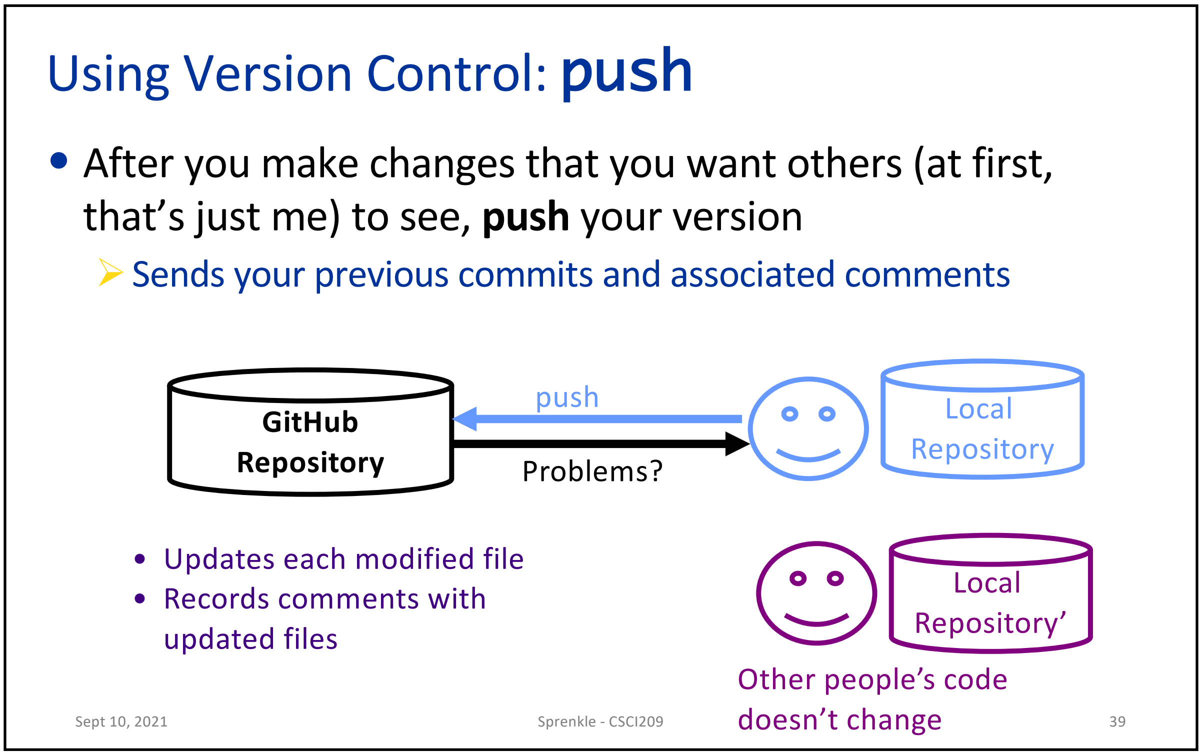
Using Version Control: Commit Messages

- Many different conventions
- Make your messages meaningful and descriptive
  - Emphasis on the *why*
  - Your future self and contributors will thank you
  - Especially as you move to bigger projects with more collaborators



COMMENT	DATE
CREATED MAIN LOOP & TIMING CONTROL	14 HOURS AGO
ENABLED CONFIG FILE PARSING	9 HOURS AGO
MISC BUGFIXES	5 HOURS AGO
CODE ADDITIONS/EDITS	4 HOURS AGO
MORE CODE	4 HOURS AGO
HERE HAVE CODE	4 HOURS AGO
AAAAAAAA	3 HOURS AGO
ADKFJSLKDFJSDKLFJ	3 HOURS AGO
MY HANDS ARE TYPING WORDS	2 HOURS AGO
HAAAAAAAANDS	2 HOURS AGO

AS A PROJECT DRAGS ON, MY GIT COMMIT MESSAGES GET LESS AND LESS INFORMATIVE.

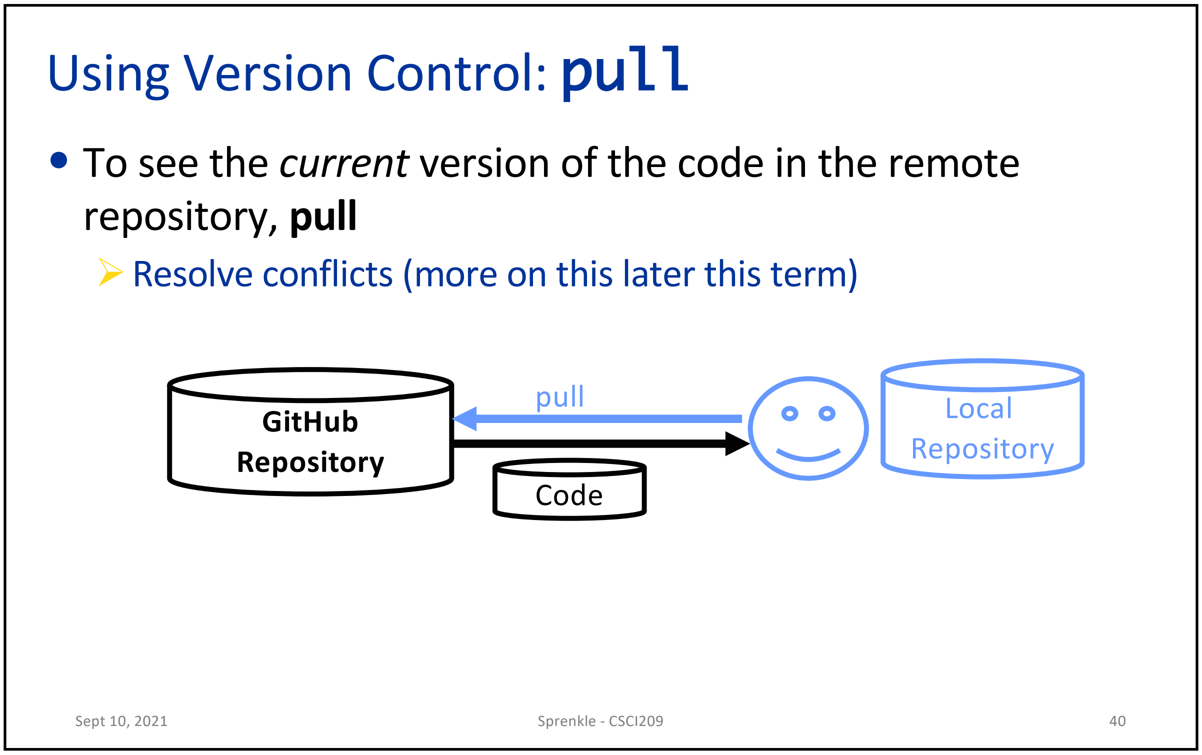


Using Version Control: **push**

- After you make changes that you want others (at first, that’s just me) to see, **push** your version
  - ➢ Sends your previous commits and associated comments

- Updates each modified file
- Records comments with updated files

Other people’s code doesn’t change

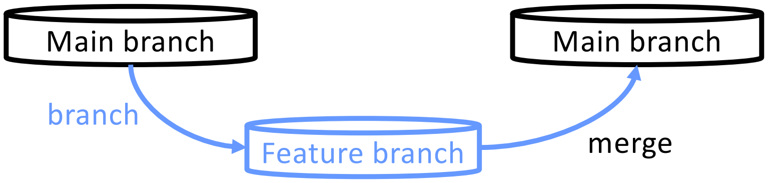


Using Version Control: **`pull`**

- To see the *current* version of the code in the remote repository, **pull**
  - ➢ Resolve conflicts (more on this later this term)

Using Git: Branches

- We create branches when we want to create a new "sandbox" to play in
  - New functionality
  - Bug fixes
  - Different approach



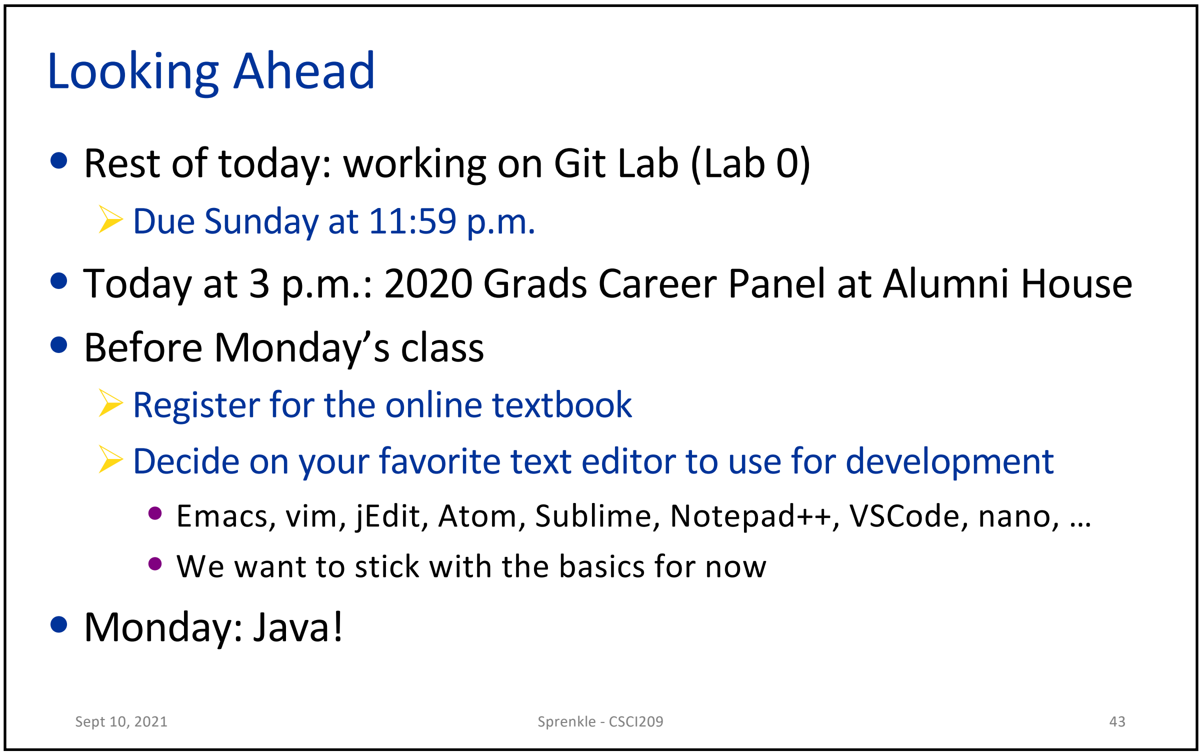
Common Git Commands

Command	What it does
clone	Clones a repository – sets up your repository so that you can coordinate
add <*file*>	Adds the *file* to the staging area
commit	Commits all the staged files (locally)
push	Push all your changes to the remote → You need your code to be pushed so that I can see it.
branch	List all local branches
branch <*name*>	Creates a new branch named *name*
checkout <*name*>	Switches to the branch named *name*

https://xkcd.com/1597/

THIS IS GIT. IT TRACKS COLLABORATIVE WORK ON PROJECTS THROUGH A BEAUTIFUL DISTRIBUTED GRAPH THEORY TREE MODEL.
COOL. HOW DO WE USE IT?
NO IDEA. JUST MEMORIZE THESE SHELL COMMANDS AND TYPE THEM TO SYNC UP. IF YOU GET ERRORS, SAVE YOUR WORK ELSEWHERE, DELETE THE PROJECT, AND DOWNLOAD A FRESH COPY.

Got it?
Let's practice!



Looking Ahead

- Rest of today: working on Git Lab (Lab 0)
  - ➢ Due Sunday at 11:59 p.m.
- Today at 3 p.m.: 2020 Grads Career Panel at Alumni House
- Before Monday’s class
  - ➢ Register for the online textbook
  - ➢ Decide on your favorite text editor to use for development
    - Emacs, vim, jEdit, Atom, Sublime, Notepad++, VSCode, nano, …
    - We want to stick with the basics for now
- Monday: Java!