CSCI 209: Software Development

Sara Sprenkle
sprenkles@wlu.edu

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

`https://pollev.com/sprenkle`

Aug 24, 2020   Sprenkle - CSCI209   1

1

CSCI209: Software Development

- How did we get here?
  - ➢ CSCI111 and CSCI112 - Fundamentals of Programming I and II

Aug 24, 2020   Sprenkle - CSCI209   2

2

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?

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
  - ➢ Robust, reliability, stability
- Code is easy to read, extend, maintain
  - ➢ Readability, extensibility, maintainability
- Application is easy to use
  - ➢ Usability
- Efficiency
- Scalability

➔ Referred to as the *ilities*

Aug 24, 2020   Sprenkle - CSCI209   6

6

3

Course Content

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

Aug 24, 2020   Sprenkle - CSCI209   7

7

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

Aug 24, 2020   Sprenkle - CSCI209   8

8

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

9

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."

10

Class Details

- Course Web Site `http://www.cs.wlu.edu/~sprenkle/cs209`
  - ➤ Example code, slides, readings, resources
- One required online textbook
- Plentiful online resources
- Participation
  - ➤ Class discussions

11

Class Details

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

12

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

Aug 24, 2020   Sprenkle - CSCI209   13

13

My Bio

August 2020

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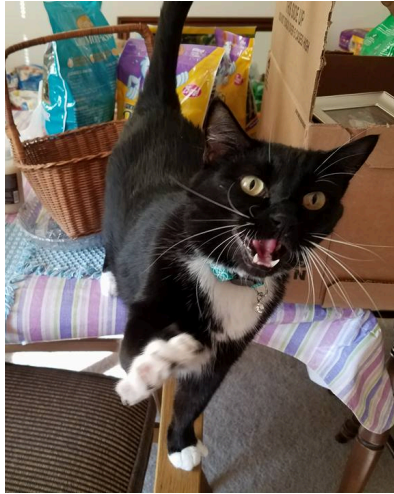- For fun: pop culture, gardening, Rockbridge Animal Alliance

October 2019

Aug 24, 2020   Sprenkle - CSCI209

14

My Roommates

Aug 24, 2020   Sprenkl

15

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
- Digital Humanities
  - ➤ Ancient Graffiti Project

CSCI335: Software Engineering via Web Applications
Spring 2021

16

My Research Interests

- Email received recently

Review Your Recent Purchases

o

write a review

a

write a review

u

write a review

17

From Kroger <noreply@kroger.com> ☆
Subject **Your null Comments Have Been Received**
To Sara Sprenkle ⭐

18

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")

SUBMIT

Aug 24, 2020   Sprenkle - CSCI209   19

19

Student Survey

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

Aug 24, 2020   Sprenkle - CSCI209   20

20

Here We Go!

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

21

VERSION CONTROL: GIT

22

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?

Aug 24, 2020   Sprenkle - CSCI209   23

23

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?

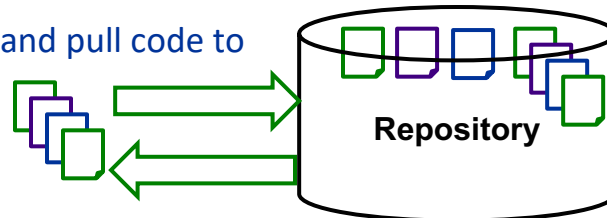Aug 24, 2020   Sprenkle - CSCI209   24

24

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

Repository

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

25

Version Control Benefits

- Backup and Restore
  - ➢ Files are saved as they are edited
  - ➢ Revert to a specific version/checkpoint
- Synchronization
  - ➢ Lets people share files
  - ➢ Stay up-to-date with the latest version
- Track changes to code
  - ➢ Save comments explaining why change happened
  - ➢ Stored in the VCS, not the file
  - ➢ Track how, why a file evolves over time
- Track Ownership
  - ➢ Tags every change with the name of the person who made it

26

Version Control Benefits

- Short-term undo
  - Messed up a file? Go back to the last **good** version
- Long-term undo
  - Created a bug a year ago? Jump back to see change you made.
- Sandboxing
  - Making a big change? Make temporary changes in isolated area, test, work out kinks before "checking in" your changes
- Branching and merging
  - Branch a copy of your code into a separate area, modify it in isolation (tracking changes separately)
  - Later, merge work into common area.

27

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

28

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/

Aug 24, 2020　　　　　Sprenkle - CSCI209　　　　　29

29

Using Git

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

Users

Repo

GitHub Repository

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

- Keeps "public" copy of code

Repo

Aug 24, 2020　　　　　Sprenkle - CSCI209　　　　　30

30

Using Version Control: `clone`
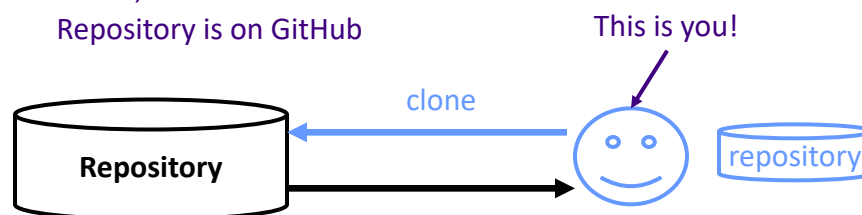
- To start, need to **clone** the repository
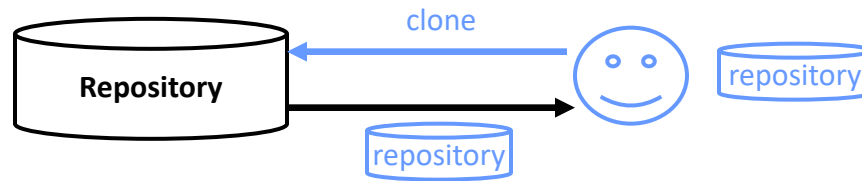
clone

Repository

repository

repository

31

Using Version Control: `clone`

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

For us,
Repository is on GitHub

This is you!

clone

Repository

repository

32

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*

commit

Local Repository ← comments? → ☺ Code
  comments ←

- Updates each modified file
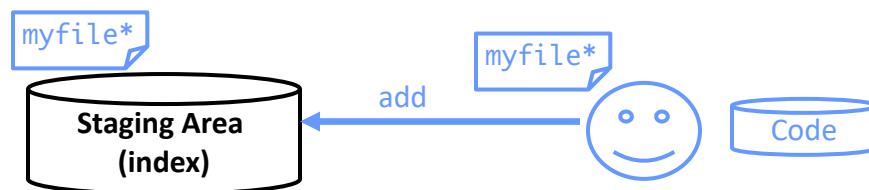- Records comments with updated files

Aug 24, 2020   Sprenkle - CSCI209   33

33

Using Version Control: `add, delete`

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

myfile*   myfile*

Staging Area (index) ← add ☺ Code

- 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

Aug 24, 2020   Sprenkle - CSCI209   34

34

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 on to bigger and better projects

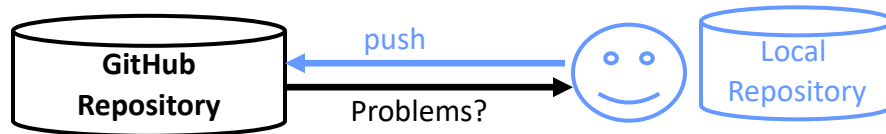| COMMENT	DATE
CREATED MAIN LOOP & TIMING CONTROL	14 HOURS AGO
ENABLED CONFIG FILE PARSING	9 HOURS AGO
MISC BUGFIXES	5 HOURS AGO
CODE ADDITIONS/EDITS	4 HOURS AGO
MORE CODE	4 HOURS AGO
HERE HAVE CODE	4 HOURS AGO
AAAAAAAA	3 HOURS AGO
ADKFJSLKDFJSDKLFJ	3 HOURS AGO
MY HANDS ARE TYPING WORDS	2 HOURS AGO
HAAAAAAAAANDS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

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

Aug 24, 2020   Sprenkle - CSCI209   35

35

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

GitHub Repository ← push → ☺ Local Repository

Problems?

- Updates each modified file
- Records comments with updated files

☺ Local Repository'

Other people's code doesn't change
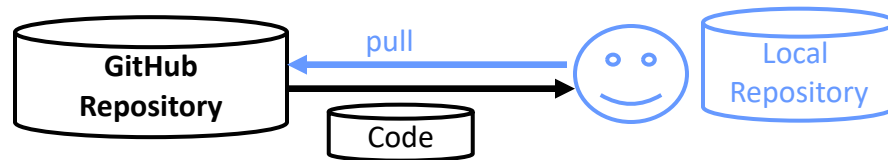
Aug 24, 2020   Sprenkle - CSCI209   36

36

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)

37

Using Git: Branches

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

38

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*

Got it?
Let's practice!

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/

Aug 24, 2020

39

39

Looking Ahead

- Rest of today: working on Git Lab (Lab 0)
  - Due before class Wednesday
- Before Wednesday's class
  - 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
  - 3 more volunteers to be remote Wed
- Wednesday: Java!

Aug 24, 2020   Sprenkle - CSCI209   40

40