CSCI 209: Software Development

Sara Sprenkle

sprenkles@wlu.edu

https://cs.wlu.edu/~sprenkles/cs209

https://pollev.com/sprenkle

How did you get to CSCI209: Software Development?

- CSCI111 and CSCI112:
  Fundamentals of *Programming* I and II
  - Recently renamed to Introduction to Computer Science and Data Structures, respectively

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"

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?

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?

What is the most important characteristic of good software?

Nobody has responded yet.

Hang tight! Responses are coming in.

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

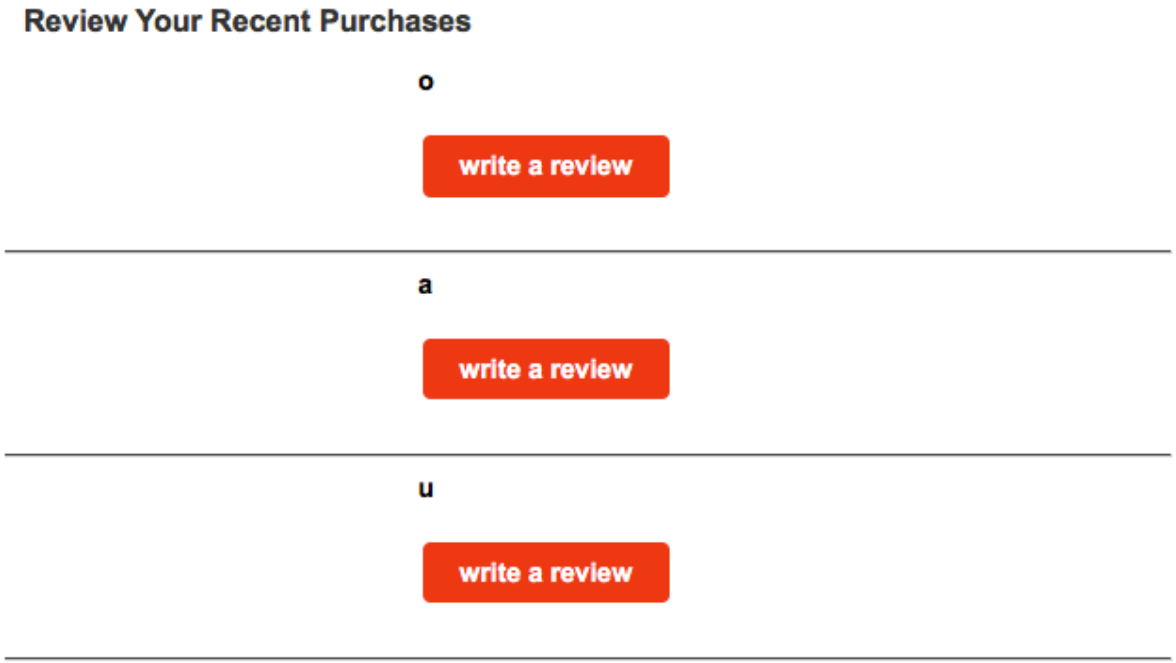
➔ Metrics of quality referred to as the **ilities*

My Research Interests

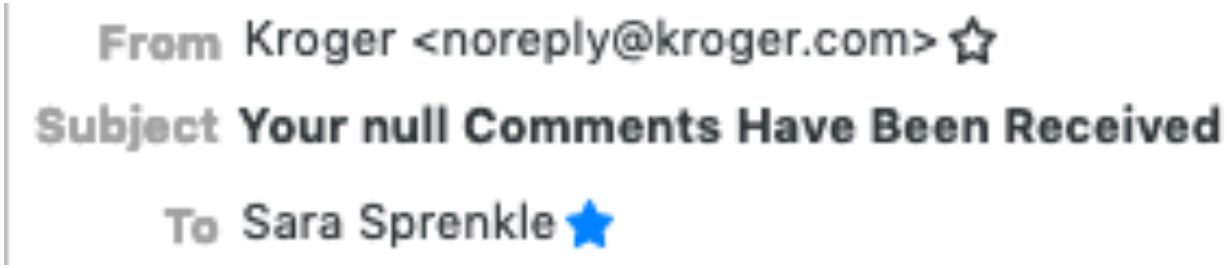
- General: Software engineering
- Automated testing of web applications
  - Develop algorithms
  - Implement in tools
  - Empirical studies
    - Try ideas out, see what actually happens, analyze
- Subject applications: Ancient Graffiti Project, ChemTutor
- CSCI335: Software Engineering via Web Applications

My Research Interests: Automatically Finding Bugs

- In an email:



My Research Interests: Automatically Finding Bugs



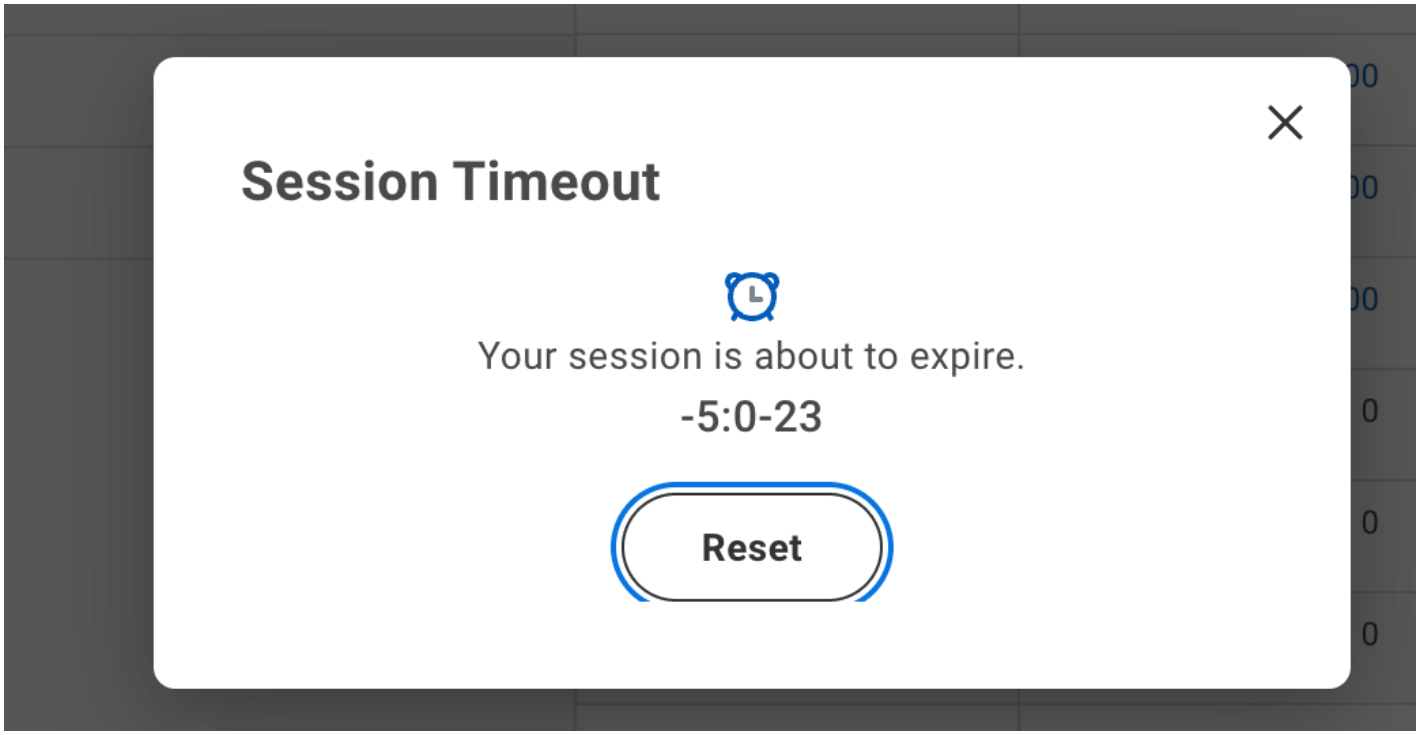
Sometimes, Errors are More Obvious

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!!”

Workday!



Course Content

- Software Design Principles
- Java programming language
  - **Statically** typed, compiled
- Software development, productivity tools
  - Version Control Systems (git)
  - 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

Practical Objectives

- Prepare you for 300-level classes, interviews, internships, jobs
  - Programming skills
  - Proficiency with environment, tools
  - Comfort/confidence

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."

Feedback from Yet Another Alumnus

"One advantage that I'm grateful for from W&L is that I've been able to make a lot of impact in the front-end and technical *writing spaces* of my projects.
I think the liberal-arts education paired with thinking about *design*/practicing writing documentation in your classes has given me a niche that's been really rewarding."

Class Details

- Course Web Site `https://cs.wlu.edu/~sprenkles/cs209`
  - Example code, slides, readings, resources, assignments
- One required online textbook
- Plentiful online resources
- Participation
  - Class discussions

Class Details

- Classes recorded via Zoom
  - Recordings posted on Canvas after class
- You can join class on Zoom (see Canvas), but it shouldn't be the common case
- Zoom Office Hours links in Canvas (TBD)
  - Please email for appointments

Communication from me

- My schedule is in flux
- If I'm not in the office: often, short responses from my phone
  - Please excuse the brevity/informality
  - Prefer a short response to no response

Where Should I Go For…?

Course Web Site

- Slides
- Assignments
- Example code
- Readings
- Resources

Canvas

- Class recordings
- Some assignment submissions
- Zoom office hours links
- Zoom class links (e.g., when sort of sick)

Class Details

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

Course Dynamics

- Professor's Responsibilities:
  - Be prepared for class
  - Make material clear
  - Challenge and encourage students
  - Provide constructive feedback to students
  - Treat students with respect
- 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

Changes from 111/112 to 209

- No lab! No student assistants!
  - Need to work on your own
  - Recommendation: Schedule your own “lab time” to work
- Emphasis changes
  - Not just about correctness and efficiency but about maintainability and human readability
- Python → Java

New! Embedded Tutor: Vincent Ziccardi!

- Class of 2026
- Majors: Computer Science and Spanish
- Extracurriculars
  - Wrestling
  - Cyber Security Club
  - Outing Club
- Favorite activities
  - Rock climbing at the Outing Club Barn
  - Playing guitar
- CSCI209 Favorite: the final project, as it was the largest coding project I've worked on and the most rewarding to complete.

My Bio

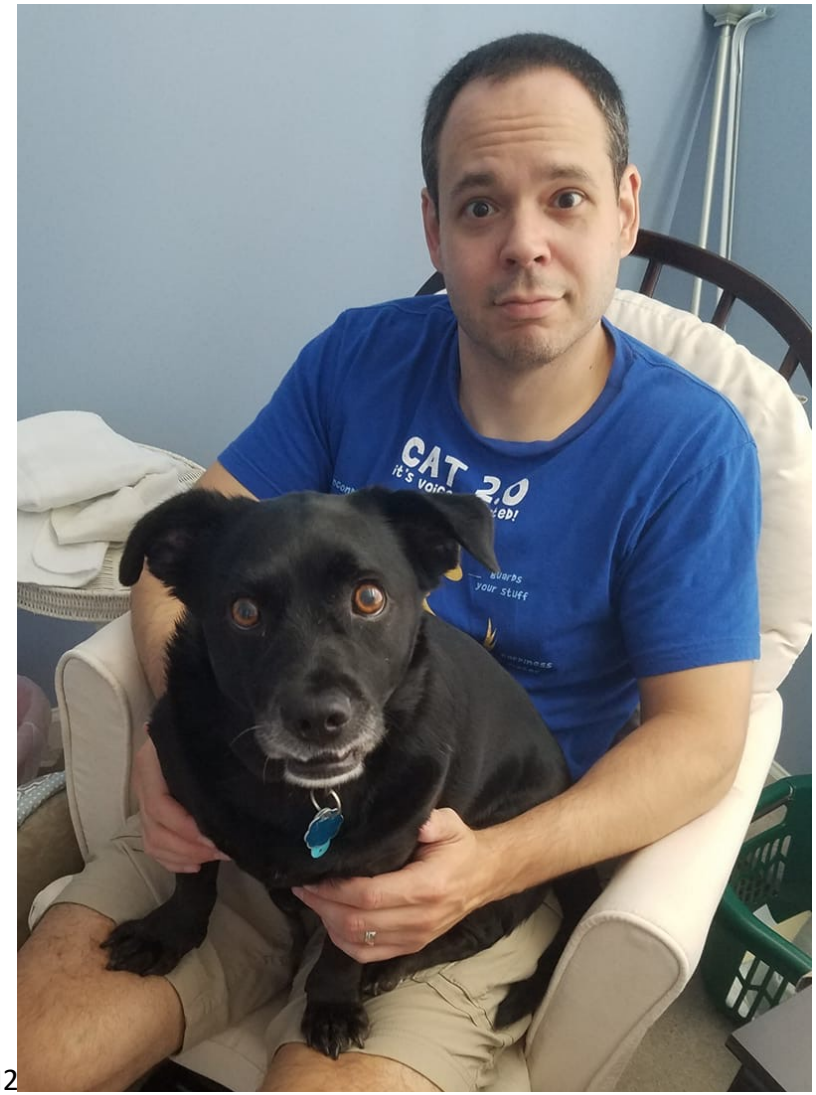
- From Dallastown, PA
- Education:
  B.S., Gettysburg College;
  M.S., Duke University;
  Ph.D., University of Delaware
- For fun: Xander, pets, family


August 2023

Sept 5, 2025

209

My Roommates



LEARNING HOW TO LEARN COMPUTER SCIENCE

The Learning Space

Not knowing how to do something

Being able to do something successfully

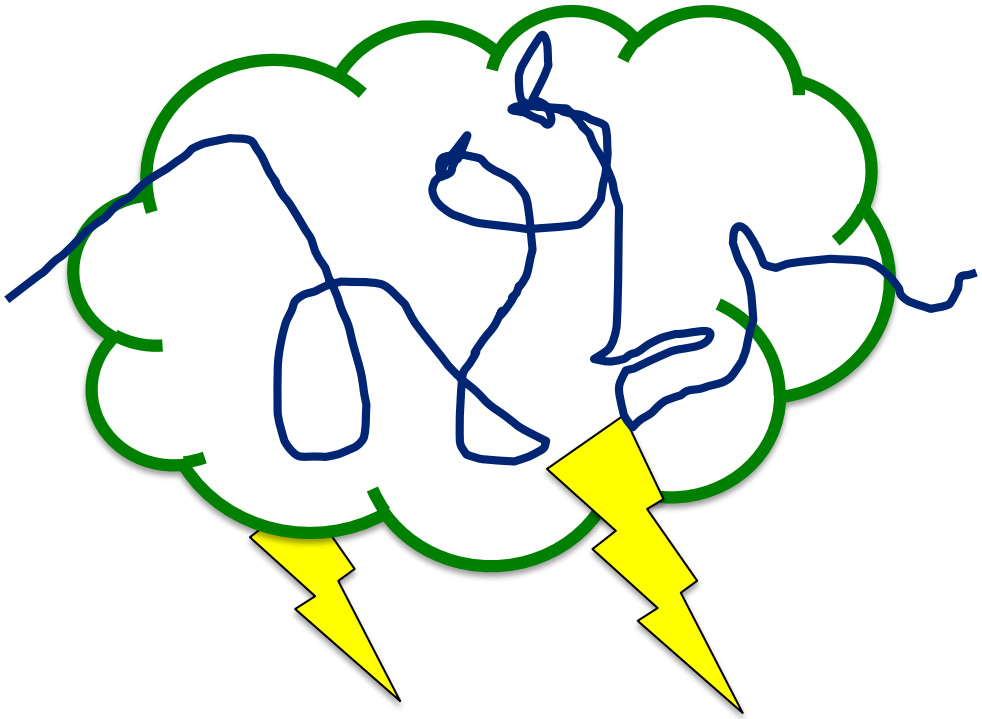
The Learning Space

Not knowing how to do something

Being able to do something successfully

The Learning Space

Not knowing how to do something

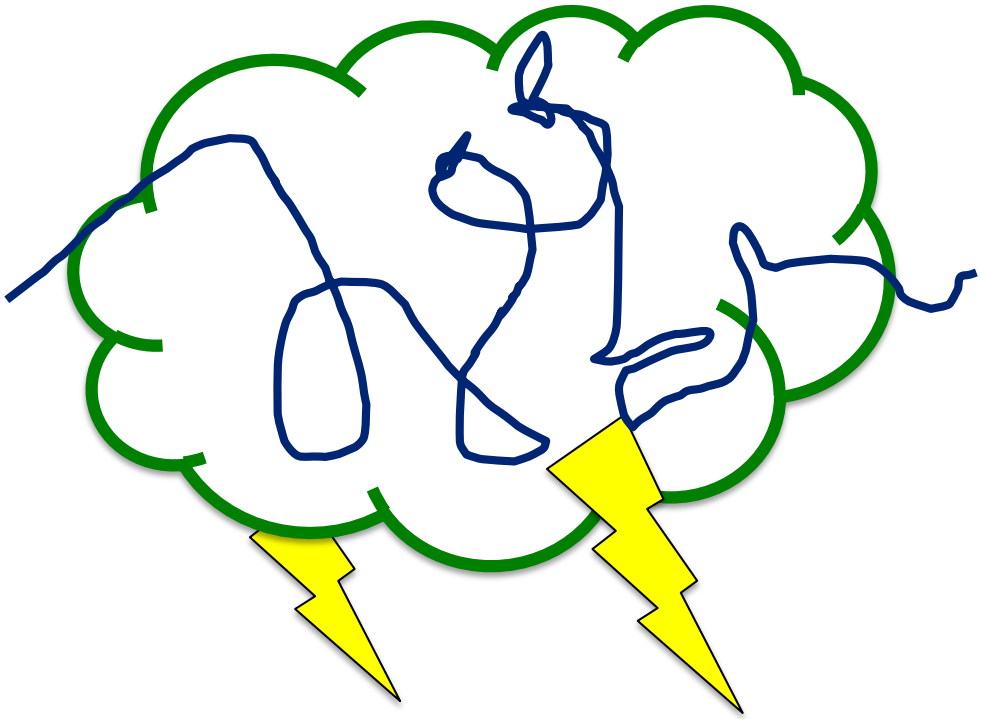


Being able to do something successfully

The path is not an easy, straight line.
Often frustration, uncertainty, and doubt in your abilities

The Learning Space

Not knowing how to do something



Being able to do something successfully

This process is where learning happens and resilience is built

The Learning Space

Not knowing how to do something

Being able to do something successfully

Towards Being a Power User Assignment

- Goals
  - 111/112 → 209
  - Comfort with Unix and the Command-Line
  - Know your computer
- Iterative process:
  1. Try to recall in your own mind
  2. Confirm/check with neighbors, internet

Collaborating at Pods

In parallel, I will go around and meet everyone.

From the Lab Computers

- Automatically mirrored

From Your Computer

- USB-C
- HDMI

- Adjust the monitors, using TWO hands!
- Read the handout at the table
- Practice: Take turns projecting the lab machines and connecting your laptops and projecting
  - Check out the CS209 web site and look at today's assignments on schedule page
    - *How will you find the web site?*
  - Start on tasks for Monday

Looking Ahead: Before Monday's Class

- Bookmark course schedule page
- Register for the online textbook
- Complete the "Towards Being a Power User" assignment
  - Due Monday before class

Expected to take less than 2 hours.
Make sure you're using resources well.
If more than two hours, stop and contact me.