CS 695 / SWE 699: Programming Tools

Fall 2023 **Wednesday, 4:30pm-7:10pm, Exploratory Hall L0003**

Grades • Syllabus, Schedule, and Slides • Announcements, Assignments, Discussion (Piazza)

Course Overview

This course will provide a comprehensive introduction to the design of programming tools, examining how developers program and the ways in which tools can make programming faster, easier, and less error-prone. The course will examine the specific challenges developers face when programming and how a wide range of programming tools may address these challenges.

This course will offer a broad survey of the new types of programming tools that researchers have envisioned to fundamentally reshape the programming experience, including tools for debugging, editing code, navigating code, visualizing code, editing code, detecting defects, reusing code, and synthesizing programs. This will include features within mainstream development environments such as Visual Studio Code, Eclipse, Android Studio, IntelliJ, and WebStorm, programming tools which are being developed and commercialized through startups such as Replay.io, https://natto.dev/, https:/stately.ai/, https://www.plasmic.app/, https://snyk.io/, https://www.codesee.io/, https://deepsource.com/, and tools that currently exist only as research prototypes.

A particular focus in Fall 2023 will be on the impact Large Language Models (LLMs) such as ChatGPT may have on the programming experience.

Weekly readings will be taken from a forthcoming textbook on Programming Tools as well as research papers published in major conferences and journals.

In-class tech talks by students will offer an up to date survey of the current state of commercial and open source programming tools, the ways in which these tools may improve programming practice today, and the challenges in using these tools to improve programming productivity.

Throughout the course, students will learn HCI methods and how to apply these methods to understanding the needs of developers and to design and evaluate programming tools. Students will evaluate existing tools, learn to apply needfinding methods to identify and characterize opportunities for new tools, and design prototypes of new programming tools. Throughout the semester, students will work in small groups on a project, where they apply the concepts of the course to study programming practice and existing tools and design a new tool prototype.

There will be no exams in the course. Grades will come from written responses to readings, class activities and tech talks, and an extended group project.

This course is cross-listed as CS 695 and SWE 699. Students enrolled in the Computer Science Ph.D. program who enroll in CS 695 will receive credit for an advanced graduate course. Students enrolled in the MS CS program who enroll in CS 695 will receive credit for an advanced course in the area of Software Engineering and Programming languages. MS SWE program who enroll in SWE 699 will receive credit for a software engineering related course.

Course Staff

Instructor: Prof. Thomas LaToza (tlatoza@gmu.edu)

Office hours by appointment.

Teaching Assistant: Emad Aghayi (eaghayi@gmu.edu)

Office hours: TBA.

Project

The homework in this course will be in the form of a project. All project work will occur in groups with up to four members.

HW0: Project Proposal

The project proposal should describe a specific aspect of software development that your project will focus on. The project proposal should clearly identify a specific challenge software developers experience in programming work, including a scenario describing a situation a developer might face. The project proposal should also include (1) a brief description of the type of study you will perform to understand this challenge better and (2) an initial idea of how a tool might address this challenge.

HW1: Review of Literature

In this assignment, you will read several papers related to your proposed project idea. You will summarize each of these papers, describing the similarities and differences with the approach you are envisioning. Based on what you learn from these readings as well the feedback you received on HW0, you will prepare a revised plan for your project.

HW2: Study of Current Practice

The study of current practice will be a small study examining a specific challenge software developers face in their programming work. Several types of study are possible. You might choose to examine posts on StackOverflow or another online repository. You might choose to perform a simple think aloud usability study and observe 4 or 5 participants perform a programming task. Or you might choose to survey professional software developers about their activities. In any case, the outcome of the study should be a better understanding of a challenge that software developers face in their programming work.

HW3: Tool Sketch

Based on the challenge identified in HW1, you should create a sketch of a potential solution. Your sketch should not contain any implementation of your tool. Instead, the sketch should provide a storyboard, depicting a series of screenshots describing the behavior of your tool on two or more examples.

Additionally, a description and high-level overview of the tool's design and implementation should be included.

HW4: Tool Prototype

Based on your sketch, you will implement a small prototype of your tool, building a plugin for an existing development environment such as VS Code to implement your idea.

Readings

Each lecture will have assigned weekly readings to complete before lecture. You will post your reactions to each reading on Piazza by answering questions about the reading.

Tech Talks

In groups of 4, you will explore a commercial, open source, or research programming tool, installing and learning the tool and using the tool in your own work. You'll then give a talk sharing your experiences with the class: what was it like using the tool, how did it fit into your programming workflow, where did you find it helpful, and where did it break down.

Tentative Schedule

1. Course Overview and Conducting Studies (8/23)

Req readings: none

2. Design Process (8/30)

HWs: HW0 due

Req readings:

- B. A. Myers, A. J. Ko, T. D. LaToza and Y. Yoon, "Programmers Are Users Too: Human-Centered Methods for Improving Programming Tools," in Computer, vol. 49, no. 7, pp. 44-52, July 2016. Summary by Thomas LaToza
- Ko, A. J., LaToza, T.D., and Burnett, M. M. (2013). A practical guide to controlled experiments of software engineering tools with human participants. Empirical Software Engineering (ESE), Sept. 2013, 1-32.

3. Problem Solving (9/6)

Req readings: TBA

4. Structured Editors (9/13)

Req readings: TBA

HWs: HW1 due

5. Program Transformation (9/20)

Req readings: TBA

6. GUI Builders & No Code (9/27)

HWs: HW2 due

7. Program Synthesis (10/4)

Req readings: TBA

8. Live Programming (10/11)

HWs: HW3 due

9. Computational Notebooks (10/18)

Req readings: TBA

10. Reusing Code (10/25)

Req readings: TBA

11. Navigating Code (11/1)

Req readings: TBA

12. Software Visualization (11/8)

Req readings: TBA

13. Detecting Defects (11/15)

Req readings: TBA

(11/22) NO CLASS - THANKSGIVING BREAK

14. Debugging (11/29)

HWs: HW4 due

Req Readings: TBA

Course Policies

Resources

This course will use Piazza for posting the schedule and all assignments and announcements. Additionally, we will use Piazza for a discussion board. Grades will be available through Blackboard.

Makeups

Unless arrangements are worked out in advance, missed exams **cannot** be made up. As each of the assignments build on prior assignments, it is important to submit each assignment in a timely fashion. 10% will be deducted for late HW assignments and late HW assignments will only be accepted for 24 hours after the due date. **HW assignments submitted more than 24 hours late will receive a zero.** If you're worried about being busy around the time of a HW submission, please plan ahead and get started early.

Grading

Responses to readings: 20%

Tech Talk: 10%

Project: 70%

Honor Code

GMU is an Honor Code university; please see the Office for Academic Integrity for a full description of the code and the honor committee process, and the Computer Science Department's Honor Code Policies regarding programming assignments. The principle of academic integrity is taken very seriously and violations are treated gravely. What does academic integrity mean in this course? Essentially this: when you are responsible for a task, you will perform that task. When you rely on someone else's work in an aspect of the performance of that task, you will give full credit in the proper, accepted form. Another aspect of academic integrity is the free play of ideas. Vigorous discussion and debate are encouraged in this course, with the firm expectation that all aspects of the class will be conducted with civility and respect for differing ideas, perspectives, and traditions. When in doubt (of any kind) please ask for guidance and clarification.

Accommodations for Disabilities

If you have a documented learning disability or other condition that may affect academic performance you should: 1) make sure this documentation is on file with the Office for Disability Services to determine the accommodations you need; and 2) talk with me to discuss your accommodation needs.

Privacy

Students must use their Mason email account to receive important University information, including messages related to this class. See Mason Email Systems for more information.

Other Useful Campus Resources

Writing Center

University Libraries

Counseling and Psychological Services (CAPS)

University Policies

GMU Academic Calendar