Course Syllabus

Performant Software Systems with Rust

Instructor

Baochun Li, Professor (bli@ece.toronto.edu)

Overview

This course aims to develop a practical and in-depth understanding on best practices of building modern software systems that are memory safe, highly performant, and secure, using the Rust programming language.

This course covers most of the basics on developing a standalone command-line utility using Rust, including important features in Rust, such as ownership and borrowing, lifetimes, traits, generics, error handling, functional programming, and smart pointers. After covering the basics, the course focuses on two advanced topics: **(1)** it covers asynchronous programming in Rust, and shows how it can be used to develop performant servers with asynchronous concurrency. **(2)** it introduces useful design patterns that can be used to build large software systems with Rust, using a performant discrete event simulator for network simulations as a case study.

Due to the practical nature of this course, much of the learning will be applied in hands-on programming assignments. Students will also complete a course project in teams (of 2-3 students in each team), on building a system framework of their own choosing. There will be a strong emphasis on best practices and design patterns for building performant software systems in practice.

Lecture Topics

As this is the first time this course is offered, the topics are subject to change. The following is a tentative list of topics that will be covered in this course:

1 Introduction to the Course

- About me
- Goals and the course structure (4 assignments, 1 project)
- Course website
- Textbook
- Grading policy
- Advantages of the Rust programming language

2 Learning Rust the Hands-On Way

- Reading the errors carefully
- Installing and using the Rust toolchain
- Creating, building, and running the first Rust project
- Implementing a guessing game

3 Basic Programming Concepts

- Mutable and immutable variabls
- Static types
- Scalar data types
- Compound data types
- Functions
- If expressions
- Loops

4 Ownership and the String Type

- Ownership: a unique feature in Rust towards memory safety
- The `String` type
- References and borrowing
- String slices

5 Structs and Enums

- Defining and instantiating structs
- Methods
- Enums
- Options
- Pattern matching
- The power of enums
- Characteristics of object-oriented languages

6 Error Handling

- Recoverable and unrecoverable errors
- Handling unrecoverable errors with `panic!`
- Handling recoverable errors with `Result`
- To `panic!` or not to `panic!`

7 Vectors and Hash Maps

- Storing lists of values in vectors
- Storing key-value pairs in hash maps

8 Generics and Traits

- Generic data types

- Traits: defining shared behaviour across structs

9 Lifetimes

- Providing hints to the compiler using lifetimes

10 Functional Rust

- Closures: anonymous functions
- Processing a collection of items with iterators

11 Smart Pointers

- Using `Box<T>` to allocate memory on the heap
- Trait objects
- Dynamically sized types and the `Sized` trait
- Wide pointers
- The `Deref` trait and deref coercion
- The `Drop` trait
- `Rc<T>`, the reference-counted smart pointer
- `RefCell<T>` and the interior mutability pattern

12 Fearless Concurrency — Threads

- Multi-threaded programming
- The actor model
- Message passing using channels
- `Send` + `Sync` marker Traits
- Atomic types from the Rust standard library

13 Asynchronous Rust

- Asynchronous Rust with stackless coroutines
- Asynchronous runtimes
- Futures and `async` / `.await`

14 Best Practices and Idiomatic Rust

- Using Clippy for static checking
- Writing and running tests using `cargo test`
- Unit vs. integration tests
- Managing large projects with packages, crates, and modules
- Design patterns and the *singleton* design pattern
- Idiomatic Rust

Assignments

Four sets of hands-on programming assignments are given in this course. Each assignment should be completed on an individual basis, using Rust as the programming language.

1 Reversi Board Game

In this assignment, students are expected to build a command-line utility to allow two human players to play the Reversi board game to completion. Its implementation uses basic types and control flows, and interacts with the user.

2 Search Utility

In this assignment, students are asked to build a simple command-line search utility that implements the basic functionality of the UNIX `grep` command.

3 Web Client

In this assignment, students are expected to implement a command-line utility that makes HTTP requests, similar in spirit to the `curl` UNIX utility. This assignment will help students gain more experiences on using external crates in the Rust ecosystem as dependencies, and on handling errors.

4 Web Server

In this assignment, students are expected to implement a simple web server to maintain a personal music library. This assignment will help students gain experience developing a more complex program that utilizes multiple CPU cores concurrently, implements persistent data storage, and incorporates external crates from the Rust ecosystem as dependencies.

Course Project

The course project represents a more innovative and time-consuming piece of work than the course assignments. There are no prescribed requirements for the nature of your course project: it can be a new software framework similar to any of the frameworks on crates.io, a new standalone command-line utility, a new web application that involves a web backend, a new distributed system service, or even a new pull request that adds a feature to — or fixes a critical issue in — an existing open-source project. The only requirement is that the source code of your course project should be 100% written in Rust. It would also be desirable for most third-party frameworks that the project uses to be mostly written in Rust as well. With the prevalence of large language models designed specifically for coding assistance, you may be able to build a project that is quite ambitious and novel within a matter of weeks.

The course project involves three stages: building a team, writing a project proposal, and completing the project by the due date. We will soon publish some inspiring project ideas that may help you define your own project. However, your mileage may vary as teams will have very different backgrounds and skill sets. Your team's project may be substantially less (or more) challenging than these project ideas.

Grading

- Assignments 45%
  - 10% each for the first three assignments
  - 15% for Assignment 4
- Project 55%
  - Proposal (10%)
  - Video Presentation (10%)
  - Video Demo (5%)
  - Final Deliverables and Report (30%)

Accommodations

The University of Toronto supports accommodations for students with diverse learning needs, which may be associated with mental health conditions, learning disabilities, autism spectrum, ADHD, mobility impairments, functional/fine motor impairments, concussion or head injury, visual impairments, chronic health conditions, addictions, D/deaf, deafened or hard of hearing, communication disorders and/or temporary disabilities, such as fractures and severe sprains, or recovery from an operation. If you have a learning need requiring an accommodation the University of Toronto recommends that students register with Accessibility Services as soon as possible. We know that many students may be hesitant to reach out to Accessibility Services for accommodations. The purpose of academic accommodations is to support students in accessing their academics by helping to remove unfair disadvantages. We can assess your situation, develop an accommodation plan with you, and support you in requesting accommodation for your course work. The process of accommodation is private; we will not share details of your needs or condition with any instructor.

If you feel hesitant to register with us, we encourage you to reach out for further information and resources on how we can support. It may feel difficult to ask for help, but it can make all the difference during your time here.

Phone: 416-978-8060

Email: accessibility.services@utoronto.ca

Equity, Diversity and Inclusion

U of T Engineering strives to create equitable and inclusive learning environments for all individuals. Each person has their own lived experiences and socio-cultural identities that shape their worldview and their experiences with privilege and oppression.

Looking for community? Feeling isolated? Not being understood or heard?

You are not alone. You can talk to anyone in the Faculty that you feel comfortable approaching, anytime — professors, instructors, teaching assistants, department academic advisors, student leaders or the Assistant Dean of Diversity, Inclusion and Professionalism.

You belong here. In this class, the participation and perspectives of everyone is invited and encouraged. The broad range of identities and the intersections of those identities are valued and create an inclusive team environment that will help you achieve academic success. You can read the evidence for this approach here.

You have rights. The University Code of Student Conduct and the Ontario Human Rights Code protect you against all forms of harassment or discrimination, including but not limited to acts of racism, sexism, Islamophobia, antisemitism, homophobia, transphobia, ableism, classism and ageism. Engineering denounces unprofessionalism or intolerance in language, actions or interactions, in person or online, on- or off-campus. Engineering takes these concerns extremely seriously and you can confidentially disclose directly to the Assistant Dean for help here.

Support for Indigenous Students

If you are an Indigenous engineering student, you are invited to join a private Discord channel to meet other Indigenous students, professors, and staff, chat about scholarships, awards, work opportunities, Indigenous-related events, and receive mentorship. Email Professor Bazylak or Darlee Gerrard if you are interested.

Indigenous students at U of T are also invited to visit Nations House's (FNH) Indigenous Student Services for culturally relevant programs and services. If you want more information on how to apply for Indigenous specific funding opportunities, cultural programs, traditional medicines, academic support, monthly social events or receive the weekly newsletter, go to the FNH website, email or follow FNH on social media: Facebook, Instagram, or TikTok. A full event calendar is on the CLNX platform. Check CLNX often to see what new events are added!

As part of the Faculty's commitment to improving Indigenous inclusion, all U of T Engineering faculty, staff and students are called upon to start or continue their personal journeys towards understanding and acknowledging Indigenous peoples' history, truths and culture. The Faculty's 2018 Blueprint for Action report identified many actions to (re)building relationships between U of T Engineering and Indigenous peoples. These actions also align with the Truth and Reconciliation Commission of Canada's Calls to Action.

One action is for non-Indigenous people to make a Land Acknowledgement in lectures and/or before important meetings. For reference, here are examples from the University and the COU: University of Toronto's Statement of Acknowledgement of Traditional Land and Council of Ontario Universities' Acknowledgement of Traditional Land.

To combat the fear that land acknowledgements have become largely performative and rote, we advise that the acknowledgement is done with sincerity and reflection on how you benefit from using the land, and on your commitments to Truth and Reconciliation. Take this training to learn more.

As an environmentalist, I have always been appreciative of the fact that right now we are in the territory covered by the Dish With One Spoon Wampum Belt. This is a treaty originally by the Anishinaabe, Mississaugas, and the Haudenosaunee. It is not a rights-based agreement, but rather a responsibility-based agreement. Hundreds of years ago the First Nations People in this area recognized our responsibility to share and protect the land. Since that first wampum belt many other First Nations and Europeans have joined in this treaty. The First Nations close ties to the land enabled them to see the needs of the land when others could not or choose not to see. For that I am thankful to the Indigenous People of this land we work and study on.

Looking for help personalizing the land acknowledgement? Professor Jason Bazylak is the Dean's Advisor on Indigenous Inclusivity. He is willing to chat with anyone on how to go about presenting and/or discussing this with your class. He can be reached at jason.bazylak@utoronto.ca (after January 2024). In 2023, you can contact Darlee Gerrard in the Outreach Office at darlee.gerrard@utoronto.ca or Marisa Sterling, P.Eng. in the Dean's Office at marisa.sterling@utoronto.ca.

Wellness and Mental Health Support

As a University of Toronto Engineering student, you have a Departmental Undergraduate Advisor or a Departmental Graduate Administrator who can support you by advising on personal matters that impact your academics. Other resources that you may find helpful are listed on the U of T Engineering Mental Health & Wellness webpage, and a small selection are also included here:

U of T Engineering's Mental Health Programs Officer

Accessibility Services

Graduate Engineering Council of Students' Mental Wellness Commission Visit the SGS Grad Hub and click on Resources & Supports. Navigate to Health & Wellness: Graduate Wellness Services at SGS

Academic Success

We encourage you to access these resources as soon as you feel you need support; no issue is too small. If you find yourself feeling distressed and in need of more immediate support, consider reaching out to the counsellors at U of T Telus Health Student Support or visiting U of T Engineering's Urgent Support — Talk to Someone Right Now.

Last updated on November 14, 2025