(CDS 264, Tuesdays and Thursdays, 1:30-3:15pm)

Course Policy

Administrative

- *Instructor*: Ari Trachtenberg (trachten@bu.edu, 617-353-2811, PHO 427, Zoom),
- *TA:* Sina Moayed Baharlou (baharlou@bu.edu, PHO 307)
- *Graders:*
    - Zane Mroue (zanem@bu.edu)

- Varsha Athreya (vathreya@bu.edu)

Office Hours

These are our standard office hours. Please pay attention to the announcements for any changes.

Day	Time	Staff	Location
Mondays	2-3pm	AriTrachtenberg	PHO 427
	7:30-8:30pm	VarshaAthreya	PHO 305
Wednesdays	3-4pm	AriTrachtenberg	PHO 427
	4:45-5:45pm	ZaneMroue	PHO 305
Thursdays	4:30-5:30pm	SinaMoayedBaharlou	PHO 305
Fridays	12-1pm	SinaMoayedBaharlou	PHO 305

Course Content

Scaling software to many developers, modules, or machines requires a fundamentally different skill-set than writing short prototype code. Software engineers working on large-scale web, financial and healthcare systems, or even multiplayer games must synthesize a wide variety elements at all abstraction layers, ranging from hardware and operating system constraints through full-stack considerations of distributed execution, security, databases, and front-end development.

This course will focus on the principles of scalable and full-stack software engineering, including:

- **Design** – how to design code for efficiency, modularity, interoperability, security, and extensibility;
- **Distribution** – the issues inherent in efficient and reliability distributed processing over many machines;
- **Optimization** – the effects of design and distribution on performance, and optimization through all abstraction layers;
- **Security** – fundamental elements of secure software design and implementation.

Prerequisite

The course will include significant amounts of programming in a number of different languages, primarily JavaScript, Java, C, and C++ and variants thereof. Though there may be short tutorials on languages, as they are encountered, students will be expected to learn language elements and software tools on the fly, as needed, and to collaborate on increasingly complex and collaborative programming projects. A fundamental background in software programming is thus required, although a background in data structures and algorithms may also be helpful (but is not required).

Resources

Textbooks

Required

- **[EJ]** Bloch, *Effective Java* (Third Edition), Addison-Wesley Professional, 2018: A well-informed guide to design best-practices for professional Java code. The eBook version is also acceptable.

References

- **[EFC]** Meyers, *Effective C++* (Third Edition). Addison-Wesley Professional, 2005: A C++ counterpart to the course text, but with a different focus.
- **[COR]** Grimm, *C++ Core Guidelines Explained* .2022: Best practices for modern C++.
- **[ESTL]** Meyers. *Effective STL* . Addison-Wesley Professional, 2001: Design guide for using C++'s Standard Template Library.
- **[CJV2]** Horstmann. *Core Java Volume II, Advanced Features* , 2022: A good reference for some of Java's more advanced features.
- **[DND]** Deitel and Deitel. *C++20 for programmers* , 2023: A basic reference for those new to C++.
- **[EMC]** Meyers. *Effective Modern C++*. O'Reilly, 2014. An updated version of [EFC].
- **[JC]** Goetz. *Java Concurrency in Practice*. Pearson Education, 2006: Discusses concurrency issues in Java.
- **[DP]** Gamma et al. *Design Patterns: Elements of Reusable Object-Oriented Software*, Pearson, 1994: A core reference on software design patterns.
- **[MEFC]** Meyers. *More Effective C++*. Addison-Wesley Professional, 1996: An older version of **[EFC]**.
- **[JP]** Bloch et al. *Java Puzzlers*. Addison-Wesley Professional, 2005: Some interesting ideosynchracies of Java.

Tutorials

In-class tutorials

The following tutorials are recommended for students who want more experience with material of relevance to the class.

Date	Time	Place	Topic	Instructor	More
1/22 - Wed	4:45-5:45pm	PHO 305	Foswiki and markdown	ZaneMroue	Slides
1/24 - Fri	12-1pm	PHO 305	Tools: git/linux	SinaMoayedBaharlou	Slides
1/27 - Mon	7:30-8:30pm	PHO 305	JavaScript/CSS/HTML	VarshaAthreya	Slides
1/29 - Wed	4:45-5:45pm	PHO 305	Java	ZaneMroue	Slides
2/ 3 - Mon	7:30-8:30pm	PHO 307	C++	VarshaAthreya	Slides
2/ 7 - Fri	12-1pm	PHO 305	debugging	SinaMoayedBaharlou	Slides
2/10 - Mon	7:30-8:30pm	PHO 305	IDEs	VarshaAthreya	

External tutorials

- BU Research Computing has various training modules and online courses: linux, C, C++, and SQL.
- The Java tutorial: https://download.oracle.com/javase/tutorial/ - “A practical guide for programmers”
- The Java language specification: https://docs.oracle.com/javase/specs/ -The formal specification for Java.

References

- *How To*
  - WorkInLab - how to work with the lab machines (PHO 305/307).
  - Code signing - how to cryptographically sign gdb so that it can be used on Mac machines
  - Another course: lecture notes from MIT's core course in software engineering
- *Languages*
  - JavaScript school - some good documentation on **JavaScript** ; also includes a sandbox for trying things out.
- *Online sandboxes:*
  - JavaScript - CodeHS
  - Java - CodeHS
  - C++ - CodeHS, replit
- *Platforms*
  - Documentation for the IntelliJ IDE: https://www.jetbrains.com/idea/documentation/
  - The Clion IDE: https://www.jetbrains.com/clion/
  - Android Studio for developing Android applicaitons.
  - Documentation for the NetBeans IDE: https://netbeans.org/kb/
  - Tutorial for the Eclipse IDE: http://www.vogella.com/tutorials/Eclipse/article.html

Electronic resources

Online

Note that for security reasons, class servers can only be accessed from the BU network (or VPN).

We will utilize a variety of professional web-based technologies in this course.

Course wiki

https://agile.bu.edu/ec531. You are responsible for checking the course wiki page regularly. It will contain handouts, homeworks, and related material. The wiki will also eventually allow you to check your grades.

Gitlab

https://agile.bu.edu/gitlab - This will server as our class git server for version control. We will also use it for issue tracking and peer-review.

Questions & Answers

https://agile.bu.edu/q2a. You may post questions about course material on our class q2a site. Extra credit problems and solutions will also be selected through this page.

Collaboration tools

You can use Zoom's whiteboard to collaborate with other students on projects.

Course Lab

As part of this course, you also have access to the Signet/VLSI Labs (PHO 305/307), which contain Linux workstations:

- All workstations should have a variety of IDEs installed (IntelliJ, Netbeans, Eclipse, Clion …), which you may use in assignments or your project.
- If you do not have door access to either PHO 305 or PHO 307, please let us the course staff.
- PHO 307 is available for shared lab hours every weekday from 6:30-8:30pm.

Unless otherwise stated, you should utilize C++17 and Java 17, for widest compatibility.

Alternatives

Several alternatives exist for accessing lab software:

1. Citrix - remote access to our licensed applications, although I don't vouch for its security posture.
2. Remote lab access: Connect remotely to the lab machines. You can use export windows on the lab machine to your own display.
3. Your own device. Much of our software has freely-available community editions. Please keep in mind that I will ***not*** provide support for your own installations - it is up to you to properly maintain and install the software. If you choose this path, you may want to install the following software:
    - The Java Development Kit (jdk 17) for most platforms, which includes the java compiler
    - IntelliJ, a well-designed professional *Integrated Development Environment* (IDE) that also sports a free Community version.
    - Clion - an IntelliJ-based C++ IDE.

- NetBeans, a mature IDE for Java sporting a project management system, debugger, and profiler, together with a graphical layout manager:
- Eclipse, an alternative open-source Integrated Development Environment for java with a slightly sleeker (and possibly easier to use) interface than Netbeans:
- Android Studio - an IntelliJ-based IDE for designing and building Android apps.

GRADES

All grades will be curved according to the class median. Thus, it is your relative score (compared to the rest of the class) that really matters, rather than your objective score. For a course at this level, I expect to center the median at a B/B+, but the final grade will depend on my assessment of the class as a whole.

Composition

Raw scores will be computed based on the following *approximate* weights:

- Best practice work [20%]
- ~10 In-class labs [20%]
- ~4 Quizzes [30%]
- ~4 Homeworks [30%]

Best practice presentation

Each student will be responsible for presenting some best practices from our course text *Effective Java* :

- Presentations will be done in groups of 1-3 students and last up to 30 minutes.
- After each presentation, the class will take a 15-minute quiz, written by the instructor based on relevant best practice items.

Student presentation grades will be distributed evenly between the following:

- The normalized performance of the class on the quiz.
- The student's performance on other presentation quizzes.

In-class labs

Much of the theoretical material in class will be accompanied by an in-class lab. Labs will typically be completed with assigned lab partners.

Quizzes

Quizzes will be administered at the end of each conceptual section of the course. The best way to prepare for these quizzes is to keep up to date on the course material and get help from the course staff, where needed.

Unless otherwise stated, quizzes ***must*** be completed individually without any external aid (*e.g.,* books, the Internet or other people).

Homework

We will have several homework projects of increasing sophistication. Your homework will consist of several components:

- **Groups:** The homeworks will be completed in groups of various sizes.
- **Peer review:** You will be expected to review code of other students, both within your group and outside your group.
- **Revisions:** Each homework project will evolve over time, adding requirements and features as directed by the instructor.

Homework solutions will be graded based on correctness and the course foci (design, distribution, optimization, and security). Your reviews and revisions will also be graded as part of your homework.

Extra credit

Extra credit may be gained by asking or answering questions on the course q2a site that selected by the instructional staff. You will need to register on the q2a site in order to answer questions.

Collaboration

I take cheating and plagiarism very seriously. At the same time, for a course of this type it is appropriate to have a very broad collaboration policy.

With exception of Quizzes (which must be done individually and without external references) and unless otherwise stated:

- You may use external references (books, web, other people) to produce work in this class, but:
  - You may only examine static web references, limited to:
    - books and research papers
    - web material that was created before the first day of class
    - material publicly accessible from the course server (agile.bu.edu).
  - You may talk to other humans (students, professors) about your work, but:
    - You ***may only*** look at code of those in your group:
      - Best practice - your writeup partner(s)
      - Labs - *your* lab partner(s)
      - Quizzes - no one
      - Homeworks - *your* staff-designated homework partners
  - You may use Generative Artificial Intellegence tools (*e.g.,* ChatGPT, Bard, Copilot) only for the best practice presentation and homeworks, but ***not*** for best practice evaluations, labs, or

quizzes, subject to the following:

- For any document in which generative AI is used, you are *required* to include a statement, at the beginning of the document, clearly and completely explaining how it was used.
- Any code taken verbatim from a genAI solution must be tagged with appropriate comments.

In any event:

- You must clearly acknowledge *all* your sources near the locations where they are used.
  - This includes any github repositories or web sites that inspired you in completing your work.
- You must make clear what code is your own:
  - All borrowed or adapted code *must* be referenced clearly in comments within the code.
  - External libraries can only be added in source form. Libraries whose source code is not available or cannot be submitted with your solution may not be used.
- You are responsible for being able to fully explain *all code* that you submit upon demand (and I will demand it!).

Failure to meet any of the above conditions will constitute a violation of academic integrity in this class. If you are not sure whether something is permitted by the course policy, ASK THE INSTRUCTOR! - it is much more awkward to explain your actions after the fact to the college disciplinary committee.