COMP 4501A (Winter 2026)

Advanced Facilities for Real-Time Games

Instructor: Oliver van Kaick

Gender pronouns: he/him/his (learn more)

Email: Oliver.vanKaick at carleton.ca

Office Location: HP5348, Herzberg Building

Best Ways to be in Touch: in class, right after class, during student hours, via the course's Discord server and Discord DMs, or via email

Student hours: student hours will be listed in Brightspace once the course starts.

Teaching Assistants: A list of teaching assistants and their contact/office hours information will be available in Brightspace once the course starts.

Class Location: Please check Carleton Central for the room location.

Lecture Times: Tuesdays and Thursdays, 1:05pm to 2:25pm

No tutorials.

Course Website:
https://brightspace.carleton.ca/d2l/home/364870

Course Discord server: you will receive an invitation to join the Discord server by email in the first week of classes. If you don't receive an invitation, you can find an invitation link in Brightspace.

Important dates and deadlines can be found here: Dates, Deadlines, and Regulations—Registrar's Office, including class suspension for fall, winter breaks, and statutory holidays.

Course Summary

COMP4501 covers the use of game engines for the development of computer games, and advanced techniques relevant to games, such as 3D rendering, animation, and the simulation of physics. Assignments consist of programming/game development tasks including a project.

Course Calendar Description

A practical course on the design and implementation of modern game engines and advanced facilities provided by these engines. Such facilities include systems for rendering 3D scenes; simulating physics; playing animations; game AI; and enabling multi-player games. Students will undertake a significant game development project.
Includes: Experiential Learning Activity
Prerequisite(s): COMP 3501.
Lectures three hours a week.

Learning Material(s) and Other Course/Lab-Related Resources

Students are not required to purchase textbooks or other learning materials for this course.
We do not have an assigned textbook, as the course draws topics from a variety of areas. The following books are useful as references for the main topics discussed in the course:

- **Game engines:** Jason Gregory, "Game Engine Architecture", Second Edition, CRC Press, 2015.
- **Computer graphics:** Peter Shirley, Steve Marschner, "Fundamentals of Computer Graphics", Third Edition, CRC Press, 2009.
- **Real-time rendering, physically-based rendering:** Tomas Akenine-Möller, Eric Haines, and Naty Hoffman, "Real-time rendering", Third Edition, A. K. Peters, 2008.
- **Animation:** Rick Parent, "Computer Animation: Algorithms and Techniques", Third Edition, Morgan Kaufmann, 2012.

Learning Material	Options for Purchasing	Approximate Cost (New)
Gregory, "Game Engine Architecture"	Amazon, used	CAD $142
Shirley and Marschner, "Fundamentals of Computer Graphics"	Amazon, used	CAD $167
Akenine-Möller, Haines, and Hoffman, "Real-time rendering"	Amazon, used	CAD $157
Parent, "Computer Animation: Algorithms and Techniques"	Amazon, used	CAD $100

The programming assignments and course project will be based on the freely-available Godot Game Engine (https://godotengine.org/) using GDScript (https://docs.godotengine.org/en/stable/tutorials/scripting/gdscript/gdscript_basics.html). There is a wealth of books and on-line tutorials specific to programming in Godot. I would advise to start by checking the free tutorials provided in Godot's website.

Topics Covered

- Architecture of games and game engines.
- Advanced rendering techniques: deferred rendering, global illumination heuristics, illumination models, programming of spatial, vertex, and fragment shaders.
- Animation: key-frame animation, mesh animation, character animation, locomotion.
- Rigid-body physics: collision detection, animation based on physical simulation.
- Introduction to soft-body physics.
- Shape modeling and acquisition.
- Networking, AI, pathfinding.

For a detailed list of the topics covered per week, please refer to the calendar available in Brightspace.

Learning Outcomes

At the end of this course, students will be able to:

- Design the software architecture for a game of reasonable complexity, using different software engineering paradigms useful for games.

- Summarize the main components that typically compose a game engine, explaining how these are integrated into a coherent software architecture, and how they can be used for game development.
- Explain the principles behind common techniques used for the creation of games, such as rendering, animation, and physical simulation. This includes the mathematical concepts and algorithms related to these techniques.
- Identify the most suitable techniques that can be used to add a specific functionality or effect to a computer game.
- Implement a game of reasonable complexity in the Godot engine, using 3D graphics.

Assessment Scheme

Grading scheme (the specific deadlines and descriptions can be found in Brightspace):

Component	**Percentage**	**Date**
Assignments	30%	Three assignments; approximately every two weeks
Project	30%	Three project milestones; approximately every two weeks
Final exam	40%	Centrally-scheduled in-person exam, during exam week

Note that you need to obtain a passing grade in the final exam to pass the course.

Late and Missed Work Policies

Late Work

Assignment deadlines are strict. The following scheme is applied to late submissions (which includes assignments and the final course project):
- 3 hours late: no penalty
- 3 to 12 hours late: -10%
- 12 to 24 hours late: -20%
- More than one day late: assignment receives a grade of zero.

If you encounter extenuating circumstances that prevent you from submitting the assignments on time, read more about academic consideration forms and long term accommodation.

Assignment submissions are handled electronically (i.e., through Brightspace). Technical problems do not exempt you from submitting on time. So, if you wait until the last minute and then have issues with your connection, you will receive a deduction according to the scheme above. Consequently, you are advised to:

- Periodically upload you progress (e.g., upload your progress to Brightspace after each major change; we will only grade your last submission).
- Submit your final work at least one hour in advance of the due date and time.
- Store backups of your assignments in the cloud, e.g., OneDrive, Dropbox, a private GitHub repository. However, your assignment has to be submitted to Brightspace so

that we have a timestamped, archived submission of your work. Urls to the cloud will not be accepted.

Missed Work
Since the course requires the submission of an assignment or project milestone approximately every two weeks, if you miss an assignment, you should work on the next assignment rather than trying to submit missed work. This will ensure that you don't get out of synch with the course. However, If you encounter extenuating circumstances that prevent you from submitting an assignment, read more about academic consideration forms and long term accommodation.

Academic Integrity and code reuse

Students are expected to uphold the values of academic Integrity, which include fairness, honesty, trust, and responsibility. Examples of actions that compromise these values include but are not limited to plagiarism, accessing unauthorized sites for assignments or tests, and unauthorized collaboration on assignments or exams.

You are free to make use of art assets found online provided that their license allows you to freely use the assets and you credit the source. Code fragments that are not of your own authorship are allowed under the following conditions: 1. The code should not be implementing the main tasks required for an assignment, but rather serve for adding additional features to the project; 2. Provide credit to the original author of the code and make sure that you understand what the code is doing.

If you are unsure about the expectations regarding academic integrity (how to use and cite references, how much collaboration with lab- or classmates is appropriate), ask your instructor. Sharing assignment specifications or posting them online (to sites like Chegg, CourseHero, OneClass, etc.) is considered academic misconduct. You are never permitted to post, share, or upload course materials without explicit permission from your instructor.

Misconduct in scholarly activity will not be tolerated and will result in consequences as outlined in Carleton University's Academic Integrity Policy. A list of standard sanctions in the Faculty of Science can be found here.

Additional details about this process can be found on the Faculty of Science Academic Integrity website.

Students are expected to familiarize themselves with and abide by Carleton University's Academic Integrity Policy.

Use of tools based on artificial intelligence

All of the assessed activities in this course except for the project were designed to be completed by an individual working alone. The use of artificial intelligence tools such as ChatGPT, Copilot, etc., is allowed in programming assignments. However, if you use any of these tools, you have to disclose this in a radme.txt file or report when submitting the activity, just as you would do when reusing code from other sources. Explain what portions of the assignment were created solely by you and what portions were created with the aid of the tool. The use of any of these tools without disclosure will be considered academic misconduct. An exception to this rule is made for automated grammar and punctuation checking tools (such as Grammarly). In addition, you are fully responsible for your code, whether you produced it with AI tools or not. Do not fully trust the output of AI tools and double check any generated code, as AI tools are known to produce incorrect outputs, such as code that does not compile or produces incorrect output.

The assignments consist of programming tasks. If any of the assignments that you submit does not run, it will receive a mark of zero. Consequently, after you upload your submission to Brightspace, you should re-download it immediately and ensure that the project can be executed.

You are expected to demonstrate good programming practices at all times and your code may be penalized if it is poorly written. You are also expected to do the necessary preparatory work (i.e., devising an algorithm) before you start coding.

School of Computer Science Laptop Requirement (only applies to on-campus courses)

Every student that has been enrolled in a 1000-level (i.e., first year) course offered by the School of Computer Science after the 2020/2021 school year is required to have a laptop. This includes COMP1001, COMP1005, and COMP1006. For more information, please visit https://carleton.ca/scs/scs-laptop-requirement/ and then review the requirements at https://carleton.ca/scs/scs-laptop-requirement/laptop-specs/.

Undergraduate Academic Advisors

The Undergraduate Advisors for the School of Computer Science are available in Room 5302HP; or by email at scs.ug.advisor@cunet.carleton.ca. The undergraduate advisors can assist with information about prerequisites and preclusions, course substitutions/equivalencies, understanding your academic audit and the remaining requirements for graduation. The undergraduate advisors will also refer students to appropriate resources such as the Science Student Success Centre, Learning Support Services and Writing Tutorial Services.

SCS Computer Laboratory

Students taking a COMP course can access the SCS computer labs. The lab schedule and location can be found at: https://carleton.ca/scs/tech-support/computer-laboratories/. All SCS computer lab and technical support information can be found at: https://carleton.ca/scs/tech-support/. Technical support staff may be contacted in-person or virtually, see this page for details: https://carleton.ca/scs/tech-support/contact-it-support/.

Mental Health and Wellness

Visit the Carleton Wellness Website for resources.

Academic Accommodations and Regulations

Academic Accommodation

Carleton is committed to providing academic accessibility for all individuals. You may need special arrangements to meet your academic obligations during the term. The accommodation request processes are outlined on the Academic Accommodations website (https://students.carleton.ca/course-outline/).

Student Rights & Responsibilities

Students are expected to act responsibly and engage respectfully with other students and members of the Carleton and the broader community. See the 7 Rights and Responsibilities Policy for details regarding the expectations of non-academic behaviour of students. Those who participate with another student in the commission of an infraction of this Policy will also be held liable for their actions.

Student Concerns

If you have any concerns regarding this course, your first point of contact is your instructor. Please email the instructor or visit them during their student hours, and they will do their best to address your concerns. If they cannot resolve the issue, the next point of contact is the School of Computer Science at studentconcerns@scs.carleton.ca. If the concern remains unresolved, the final point of contact is the Office of the Dean of Science at ODScience@carleton.ca. Please follow this order of contact. Note that you can also bring your concerns to Ombuds services.