Introduction

Welcome to cs195n (officially known as CSCI 1950N), 2D Game Engines.

This document provides you with a lot of important information about cs195n, so it is essential for you to read and understand it.

You may also want to refer to the course syllabus, the calendar, and the list of assignments, all of which are available at the course website (http://cs.brown.edu/courses/cs195n/).

Announcements made during the semester will be sent to the class e-mail list, as well as posted on the home page of the course website.

In order to ensure that every student has a positive experience in this course and gets the attention and assistance he or she needs from the course staff, enrollment has been capped at 40 students. If enrollment reaches this limit and additional students still wish to take the course, the staff will decide which students will be admitted, with preference given to seniors.

Course Goals

In this course you will learn techniques needed to create 2D game engines, including vector and raster graphics, animation, simple AI, collision detection, physics, raycasting, and the entity I/O design pattern. You will create a 2D game engine over the course of the semester, adding a few features to it each week. At the same time, you will also create a series of games using your engine that demonstrate the use of the features you add. Near the end of the semester, you will design and implement a final project that uses your game engine to create a finished, entertaining game.

Course Structure

Lectures will be held once a week for a period of 2.5 hours. The last 30 minutes to 1 hour of the lecture will be used for playtesting, a process in which all students will get to play and provide feedback on other students games. In addition to the lectures, a 15-minute meeting with one of the teaching assistants will be used as design check. Students will need to explain how they plan on solving the major concepts of the week at a high conceptual level. Homework and other out-of-class work is estimated at around 15 hours per week, including the final project.

While background knowledge of video games is helpful it is not required to do well in this course. We welcome students from all backgrounds.
Course Staff

**Professor**  Office  Email (@cs.brown.edu)
Barbara Meier  CIT 401  bjm

**Teaching Fellow**  Login
Alexandra Papoutsaki  alexpap

**Head TA**  Login
Hassan Sufi  hsufi

**UTAs**  Login
Liam Callanan  lcallana
Peter Kirschner  pkirschn
Ben Sheeran  bsheeran

Prerequisites

The official prerequisite for this course is that you have completed a CS intro sequence (CS 15/16, CS 17/18, or CS 19), or have Prof. Meier’s permission. More specifically, the most important skills for you to have in order to do well in this course are:

- **Comfort with Java.** You should be able to design, program, and debug efficiently in Java, and read and understand Java documentation.

- **Object-oriented design.** You should be comfortable designing a system of cooperating objects to represent an abstraction or solve a problem.

In addition, experience with the following will be helpful and will make your life easier in this class:

- **Large-scale projects.** It is strongly recommended to have some experience designing, implementing, and debugging non-trivial (over 2000-line) code systems, such as the multi-person projects in CS 32. Although CS 32 is not a prerequisite, having completed it will make this class much easier.

- **Vector arithmetic.** This class depends heavily on high school-level vector math (adding, subtracting, dot products, normalizing, etc.), so it helps to have a working knowledge. If you are not comfortable with vectors, though, don’t let this prevent you from taking the class – the TAs can explain everything you need to know about vectors.

- **Physics.** Some knowledge of high school-level physics, such as momentum, velocity, and forces, will help with the projects involving physics simulation.

- **Version control systems.** As many people will do the final project in a group, experience with version control systems will help synchronize your code between group members.

Note that it is not required to have a good understanding of Java’s Swing UI toolkit, even though this class involves creating graphical applications in Java. You will be provided with support code that abstracts away many of the complications of using Swing to draw a window on the screen.
Class Meetings

The class will meet on Wednesdays from 3-5:20pm in CIT 316. If the time or location changes, we will make an announcement at the preceding class and send a message to the class e-mail list.

Approximately the first 30-90 minutes of this large time block will be used for a lecture on that week’s game engine topics. After that, class will move down to the Sunlab, where the remainder of the class period will be used to playtest the assignment that is due that week. Playtesting is required in order for your project to be considered complete on time, so class attendance will effectively be mandatory.

Projects and Grading

The course consists of five programming projects: four regular projects, and a student-directed final project. Each project is divided into one or more weekly checkpoints. There are no homeworks or exams. Projects/checkpoints will be assigned each Wednesday after the lecture, and will be due the following Tuesday at midnight (11:59 pm). You will hand in assignments by running the `cs195n_handin` script from your project’s root directory on a CS department computer.

There will be two sets of project-specific requirements for each checkpoint – engine requirements and game requirements – as well as a set of global requirements that must be fulfilled for every handin. At each checkpoint, if your handin meets the checkpoint’s requirements, it will receive a “complete.” If you do not hand in an assignment on time or your handin does not meet the checkpoint’s requirements, it will receive an “incomplete.”

If a handin receives an “incomplete,” you have one week to “retry” the assignment and submit another handin that meets the checkpoint’s requirements, but only if you attended your design check! If and when you retry, the handin will be regraded, and the new grade will replace the old grade. (This means that you do not lose credit for turning in an assignment late if it is complete on the retry handin.)

Furthermore, you have two “extra retries,” each of which allows you to retry a previous retry. This means if your “retry” handin still receives an incomplete, you can use an “extra retry” to fix it and submit it again for grading. You may also hand in your project two weeks late and state that you want to use your standard and extra retry at the same time.

Note, however, that these projects are cumulative. The engine features that you implement in one week will generally also be necessary in order to complete the next week’s project. Since each project depends on the previous one, you will still need to complete them in order, and once you are late on one project you run the risk of staying behind schedule on every subsequent project. Do not let this happen! This grading system can be a little confusing at first, so don’t hesitate to email the TAs if you have any questions.

Projects are not assigned points-based grades, and all projects have the same weight. Your letter grade in the course is simply determined by the number of complete vs no-credit projects you have by the end of the course:

However, regardless of the number of completes or no-credits, you must have a complete engine that satisfies all engine requirements from every checkpoint throughout the semester.
Incompletes in the class (grades of INC) will not be given except in extenuating circumstances authorized by a dean or a note from Health Services. If you know in advance that you will be requesting an incomplete grade, please talk to the Head TA as soon as possible.

Since grades in this class are not determined by points or letter grades for assignments, there will be no grading curve for the distribution of final grades. All final grades will be determined using the table above.

All projects will be graded by the HTA or the UTAs. If you have a problem with the grade you received for an assignment, you should first talk to the TA who graded that assignment. If you are still unhappy, you can contact the Head TA; if the Head TA is unable to resolve the problem, contact the professor or the Teaching Fellow.

Final Project

The last project in this course will be an open-ended final project, for which you are allowed and encouraged to work in a group with other students. Unlike the other programming projects, you will determine the requirements for this assignment. You will be able to pick from a list of engine features to implement, and you will write your own game requirements based on the kind of game you want to create.

While some of the other projects will focus more on implementing engine features than gameplay, the goal of the final project is to use your engine to create a fun and exciting game. It’s important to put some thought into what kind of game you want to create and how you will make it enjoyable to play, so start thinking about the final project as early as possible.

You should start coming up with ideas for your final project, at the latest, by the time you are starting M (the last predefined project). An idea for your project will be due at the same time as the first weekly checkpoint for M, and a more detailed design proposal, including your list of requirements, will be due at the same time as the final handin of M. If you want to work in a group, this will also be the time to form one; some class time will be dedicated to helping people finalize project groups the week before the design proposal is due. Although it is possible to do the final project on your own, everyone is strongly encouraged to form at least a two-person group so that you will have the resources to make a more interesting game.

Design Checks

There will be a mandatory design check for every checkpoint, held on the Fridays and Saturdays preceding its due date in the Moon Lab (CIT 227). Each design check will last 15 minutes. You will sign up for a design check using cs195n_signup <project> in the terminal.

Since projects are not graded with points, design checks do not represent a percentage of your project grade. Instead, they must be completed to receive a standard retry for the checkpoint. If
you do not complete the design check for a specific checkpoint, you will have to use an extra retry to retry that checkpoint.

The design checks in this course are fairly informal. In order to receive credit for a design check, you will need to answer a few questions regarding the assignment’s content. It important that you have thought about the project, understand the concepts that it depends on, and have a plan for solving the major problems that it involves. Questions for each design check will be posted with the assignments, and the TA’s will expect you to answer these questions at a conceptual level. While it is not necessary to have prepared a design diagram or have any code written, feel free to bring either to your design checks to help answer the questions.

Playtesting

Playtesting is an important part of this class. In addition to creating a game engine, you are also creating playable video games, and you will want to make sure that other people can play them successfully. Even in the checkpoint weeks when you are not required to have a finished game, your partially-completed game should still be a functional demonstration. For the week the final version of a project is due, playtesting should be a fun way to find out what kind of games your peers have created, and receive valuable feedback on the game you created.

Furthermore, since projects do not receive numeric grades, playtesting will be an important opportunity to gauge how well your project works and detect non-obvious bugs before they become a problem. At playtesting you are encouraged to try to “break” the games you playtest, and report any bugs or odd behavior you find.

As mentioned previously, playtesting will occur during class time after lecture. Students will be randomly split into roughly equal-sized groups, each of which will have a supervising TA, and each student will playtest every other student’s game in the group. You will need to fill out a playtesting form for each game you playtest, so that the person you’re playtesting has a written record of your feedback.

Collaboration Policy

In order to make sure that each student in cs195n is graded as fairly and individually as possible, the course staff have written a collaboration policy by which we expect all students to abide. Please read this policy carefully, as it may differ from collaboration policies in other CS classes you have taken. The policy isn’t too long, and we have tried to make it easy to read.

cs195n involves some challenging software design problems for which there is often no single “right” solution. Thus, it is helpful and encouraged to discuss the projects with your peers and help each other find more creative solutions to these problems. However, the work you hand in should be entirely your own and represent your own understanding of the concepts taught in this class. For these reasons, the collaboration policy generally allows you to talk about the projects and your code’s high-level design with other cs195n students, but not to share code or help other students with debugging.

As with other CS classes that prohibit sharing code between students, you are responsible for ensuring that the permissions on your source code directories do not allow other students to view
them. Ask a consultant or a TA for help with permissions if you are unsure of how to use them.

**TA Hours**

TA hours will be held Sundays, Mondays, and Tuesdays in the Moon Lab (CIT 227). Once the TAs work out their own class schedules, the exact hours will be posted on the course web page. You can go to TA hours to ask questions about the concepts and algorithms presented in class, get advice on the design of your engine, and ask for help in solving particularly difficult bugs.

TAs are here to help you, but remember, TAs are students too. Please don’t ask them questions outside of official TA hours. This includes talking to them in person or electronically while they are at home or in the lab.

If you need to contact the TAs outside of TA hours, e-mail the alias cs195ntas@cs.brown.edu. You should generally use this alias instead of sending e-mail to TAs individually, as most questions can be answered by any TA and you are more likely to get a timely response by e-mailing the alias.

If TA hours are rescheduled or canceled for any reason, there will be an announcement to the class e-mail list. If you feel you can’t possibly make the scheduled TA hours (especially after a reschedule), get in touch with the head TA.

**Inclusivity Statement**

Creating an inclusive educational environment that embraces diversity is a matter of utmost importance. We want to ensure that all students feel welcome and capable of excellency inside and outside class despite of differences in race, nationality, gender identity, sexual orientation, religion, age, physical or cognitive abilities, economic background, military experience, political ideology, and many other dimensions.

**Special Accommodations**

Please let know the staff any accommodation request that is tied to anything mentioned above in a timely manner preferably at the beginning of the semester. You can reach us at the end of the class, during office hours, or through email.