Introduction

Welcome to CSCI1972: Topics in 3D Game Engine Development!

You may also want to refer to the calendar, and the list of assignments, all of which are available at the course website (cs.brown.edu/courses/cs1972).

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 this course you will learn techniques needed to create 3D game engines, including game engine design, procedural generation, player representation, collision detection and response, GPUs, and OpenGL, in addition to potential other topics like game networking, spatial subdivision, and more. You will independently develop two 3D game engines (one for voxel-based games such as Minecraft, and one for geometric-based platformer games). Near the end of the semester, you will work in a group to design and build a 3D game that may incorporate features from the engines you’ve developed.

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 25 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 Staff

<table>
<thead>
<tr>
<th>Professor</th>
<th>Office</th>
<th>Email (@cs.brown.edu)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barbara Meier</td>
<td>CIT 401</td>
<td>bjm</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Head TA</th>
<th>Login</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ben Sheeran</td>
<td>bsheeran</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TAs</th>
<th>Login</th>
</tr>
</thead>
<tbody>
<tr>
<td>Miranda Chao</td>
<td>mc184</td>
</tr>
<tr>
<td>Ben Attal</td>
<td>battal</td>
</tr>
</tbody>
</table>

Prerequisites

There are two prerequisites for this course. The first is some form of software engineering. Any of CS123, CS195n, CS32, or CS33 will satisfy this requirement. The second prerequisite is graphics, which may be satisfied with either CS123 or by completing the winter assignment (available on the course website). You may also obtain the instructor’s permission to take the course.

The TA staff strongly recommends taking CS123 in preparation for this course. Students who do well in CS123 are well-prepared for CS1972, as they require largely the same set of skills. More
specifically, those skills are:

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

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

- **3D Graphics/OpenGL** You should understand concepts from 3D graphics such as scene graphs, local and global coordinates, and cameras. You should be able to use OpenGL calls to draw a scene.

- **Vector arithmetic**. This class depends heavily on high school-level vector math (adding, subtracting, dot products, normalizing, etc.), so it is important to have a working knowledge.

**Classes**

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. On most weeks, class will then move down to the Sunlab (after the lecture portion), 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, so class attendance will effectively be mandatory, and missed playtesting must be made up. If you will be unable to attend class on a particular day, playtesting must be done by **no later than 6PM** on that Wednesday. Failure to do so will result in knocking your grade down by one letter grade for the week.

**Projects and Grading**

The course consists of four programming projects: a 2-week warm-up project, 2 regular projects, and a student-directed final project. Each project is divided into two 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 (11:59:59 pm). You will hand in assignments by running the `cs1972_handin` script from your project’s root directory (typically where the Qt `.pro` file resides) 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. These will be further split into primary, secondary, and tertiary requirements. At each checkpoint, if your handin meets all requirements for the checkpoint, it will receive an equivalent of an A. If you only meet primary and secondary requirements, then you will receive a B for that assignment, and if you only meet primary requirements you will receive a C. If you don’t complete at least all of the primary requirements it will receive an “incomplete”, or NC.

If a handin receives an NC you can use one of 5 “retries”, which is effectively one week to “retry” the assignment and submit another handin that meets the checkpoint’s requirements. 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.)

If, after using the appropriate retires, a handin is still “incomplete”, that checkpoint will receive “no-credit.”

Note 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!

All projects have the same weight. This class is not curved. Your letter grade in the course is determined by the average number of A’s, B’s, C’s, or NC’s you have by the end of the course, where the grades measure to numbers as follows:

<table>
<thead>
<tr>
<th>Grade</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>3</td>
</tr>
<tr>
<td>B</td>
<td>2</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
</tr>
<tr>
<td>NC</td>
<td>0</td>
</tr>
</tbody>
</table>

However, regardless of your grades, you must have handed in a working version of all checkpoints by the end of the semester to pass the class.

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.

In addition, if you need any academic accommodations throughout the semester, please email the HTA and the professor at cs1972headtas@lists.cs.brown.edu to discuss your options. We want to make sure you prioritize your health and stability so let us know what we can do for you!

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.

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.

You should start coming up with ideas for your final project, at the latest, by the time you are halfway through Platformer (the last predefined project). An idea for your project will be due at the same time as the final handin for Platformer, and a more detailed design proposal, including your list of requirements, will be due after that. 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 encouraged to form at least a two-person group and at most a three-person group.
so that you will have the resources to make a more interesting game.

**Design Checks**

Before most checkpoints you will have to do a design check, in which you will answer high-level questions we give about the project of the week. These are intended to help you with your game engine design and to also help you get a head start on your project, and these are also mandatory. If you cannot make or miss a design check, then please contact us as soon as possible. Design checks that are missed without communication will knock your grade down for that week by one letter grade.

**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 project should still be a functional demonstration.

Furthermore, 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 on weeks of projects. Students will be randomly split into roughly equal-sized groups, 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. The feedback form is anonymous.

**Collaboration Policy**

In order to make sure that each student in CS1972 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.

CS1972 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 CS1972 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 two days of the week in CIT 203. 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. If you cannot make it to TA hours or encounter a problem outside of the times in which they are held, you can send your issues to the TA list at cs1972tas@lists.cs.brown.edu, but please try to come to hours as much as possible.

If you need to contact the TAs outside of TA hours, e-mail the alias cs1972tas@lists.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.