Global Requirements

Spring 2018

Assignment Handins

These requirements must be met for every assignment you hand in, including the non-final weeks of multi-week projects. They represent our basic expectations for work done in this class, and if you do not meet them we will not give you credit for the assignment.

- Your top-level handin directory must contain a Qt project file (.pro).
- You must hand in a file called README explaining anything the TAs need to know about your code. At the minimum, it must contain:
  - A copy of the rubric for the assignment, which can be found at `/course/cs195u/rubrics/<asgn>/grade.txt`. This will be used for the following two items.
  - How to verify each engine requirement. This will vary depending on the requirement, but can usually be satisfied by pointing to source files in your engine package. Please give information for each requirement in the rubric.
  - How to verify each game requirement. Again, this will vary depending on the requirement, but can usually be satisfied by either pointing to source files in your game package or indicating how someone might find evidence of this requirement while playing the game. Please give information for each requirement in the rubric.
  - Documentation of any known bugs, or indication that there are none.
  - The approximate number of hours it took you to complete the assignment. This will help us gauge the difficulty of the assignments for future iterations of the class.
- You must hand in a file called INSTRUCTIONS file containing instructions for how to play your game.
- Engine and game code must be both logically and functionally separated. It should be possible to create an entirely new and different game by reusing just your engine code, and this code should exist in a different subdirectory than your game code.
- Engine code must be well-designed. Although we do not grade code style and comments, we do expect that you give careful thought to the design of your engine so that it can be reused from week to week without major refactoring. We will not give credit for an assignment that violates basic design principles or seems excessively "hacky." Examples include making every field of your objects public or making your entire engine a single object.
- Game code must be reasonably designed. Since your game code will not be reused as often or for as long as your engine code, it is not as important to make it extensible and generic. Nevertheless, we still reserve the right to refuse credit for a game that is excessively poorly designed, such as executing all of your game code inside a single method.
- Your code must work on department machines.

- Your code must not crash under any circumstances.
- Your game must work under the following resolutions: the standard 800x600 window set up by the support code and fullscreen on the standard two-monitor computers in the SunLab. Minimally, this means that your camera transformations change based on the new aspect ratio. However, we recommend that any other game UI is drawn based on the window size as well.
- Your game should generally run at 30 updates per second (UPS) or above on any department machine. It must never go below 20 UPS unless otherwise specified in the assignment handout.
- You must have attended a design check. If you do not attend a design check, you will automatically use your standard retry for the week on the first handin.
- You must complete playtesting for your handin to be considered on-time. If you do not complete playtesting for an assignment, you will automatically use your standard retry for the next assignment.

Design Checks

Design checks will be held Thursdays through Saturdays every week. You will sign up for a design check using cs195u_signup ¡project¿ in the terminal. As mentioned above, attending a design check is mandatory. If you do not attend a design check or forget to sign up for one you will **lose your standard retry for that week.** The weekly checkpoints for each project will contain a design check section containing the questions you will be asked. Feel free to answer questions with verbal descriptions, diagrams, or even code if you start coding before your design check. Though these will not be as formal as other courses' design checks, we will have the following expectations:

- You must have thought about the project and your class structure.
- You should either understand the major concepts or algorithms necessary to complete that week's project, or have questions that will help clarify these concepts and algorithms.
- You should be able to explain how you will structure your code to solve the problem, or have questions that will help clarify how to structure your code.
- Your design check should take no longer than 20 minutes.

We encourage you to think of design checks as your own personal hours slot. Though we will ask some general questions to ensure you understand the week's material, design check slots are an excellent time to ask any other questions you have for the week's assignment.

Playtesting

After each week's lecture there will be a playtesting session for the assignment due that week, unless it is the first checkpoint of a project. Even in weeks that are not the final week of a project, you should still have a working program that other students can run and interact with. As mentioned above, completing playtesting is a mandatory part of each assignment. Each student will be assigned to a small group for playtesting. The basic requirements for playtesting are:

- Each person in the group must playtest each other person in the group.
- You must fill out a feedback form for each game that you playtest.
- If your handin is not ready for playtesting (i.e. is not working by class time), you will not get playtested. However, you must still playtest the other members of your group.
- If your project does not meet all the requirements but works to some extent, you should hand it in anyway so that the features you have finished can get playtested.