Exercise 6: Freestyle II

Introduction

For this assignment, you will implement another game with the similar constraints to exercise 5. Again, this is an exercise in starting from a blank slate. However, you must make a game in a different genre than your entry for exercise 5, and you cannot reuse any of your own code between the two exercises.

See exercise 5 for general advice on possible genres, etc.

Requirements

You can make any kind of game you like provided that (this is roughly what the grading rubric will be):

- Objects are on the screen
  - There must be at least three kinds of objects on the screen
  - At least two kinds of objects must move
- Controls
  - The player must be able to directly control at least one object on screen. This could be using a joystick to pilot a car in a racing game, or using a button to control paddles in a pinball game.
- Object interactions
  - At least three kinds of objects on screen must be able to interact with one another.
  - Possible interactions include, but are by no means limited to:
    - An object chasing another object
    - Objects colliding
    - One object landing on another
    - One object applying a readily identifiable force to another (e.g. gravitational attraction of a spaceship to a star)
- Sound
  - Events that are significant in the game (in the sense that it's important that the player know they happened) must be marked by sound cues (the playing of a sound).
  - This only applies to instantaneous events, like collisions. You don't have to figure out how to make sounds out of on-going game state such as the player's health (although it's fine if they do).
  - For example, this events would need sound cues:
    - Scoring
    - Colliding with the player, if the player has an avatar
    - Firing of weapons
  - Examples of events that are not required to have sound cues (although it's fine if they do):
    - Events involving objects the player never interacts with
    - Steering in a racing game – the game doesn't need to generate a sound just because the player changed direction.
- Goal and progress:

    - The player must have a goal they understand
    - The player must have a way of evaluating whether they're making progress toward that goal
    - The player must have a way of knowing when they've achieved the goal
- Playability
    - The game's instructions must be sufficient to allow the reviewers to play it and understand if it's working
    - The player shouldn't be able to get into "stuck states" such as flying off screen and not being able to figure out how to get back

You must include instructions for your game. You are responsible for insuring that the reviewers understand the intended behavior of your game well enough that (a) they can play it and (b) they can determine whether you've successfully implemented the requirements listed above.

Academic dishonesty and intellectual property rights

You may use any sprites and sounds you like for this assignment provided that:

- You can legally use and distribute them (don't pirate them)
- You include file named CREDITS.txt that documents who their original authors were, or failing that, what web site you got them from.

You may also use any code you like from other sources, provided that:

- You credit it in CREDITS.txt
- It is not used to implement any of the items listed above under "Requirements". That is, if you want to use someone's cool particle system for some effect, that's fine. But if we were to disable that code, the resulting game should still satisfy all the assignment's requirements.

What to turn in

Upload a ZIP file to canvas containing:

- An instruction file called INSTRUCTIONS. If it's in some format other than Word or TXT, include a PDF version of it. The file should make very clear
    - What all the objects on screen are
    - What their behavior is supposed to be
    - What the player's controls are
    - How the player scores
    - How the game ends, including win/lose conditions, if appropriate
- Your CREDITS.txt file, if you used assets you obtained on-line
- Your Assets, Packages (if any), and ProjectSettings directories