CSCI 321, 2D Pygame Specifications

Geoffrey Matthews

April 3, 2015

1. Game programming must be your own work. You may consult with each other on ideas, artwork, *etc.*, and use each other as game testers, but you may not consult on source code.

2. You can use resources from the internet, but all games must contain substantial original work, and credit must be given for resources created by others. You can copy an existing game design, but the majority of the programming must be your own. You can use modules and code of others, but it must be acknowledged both in the code and in the documentation, and the bulk of the code must be your own. The artwork and sounds may be original or not, but credit must be given for all work that is not your own.

3. All games must be accompanied by a Game Manual. The game manual must be typeset nicely, and must include:

   - Title page. Name for the game, your name(s), student number(s), class number, quarter, instructors name.
   - Back story for the game (if any). If you have different kinds of NPCs (non-player characters), give each of them their story.
   - User's guide. This includes how to play, what the objective of the game is, how scoring is decided, different play modes, *etc.* Most important: how to locate the in-game help screen (usually F1), and how to quit the game (usually ESC). Another common pattern is to have ESC bring up a menu, and two of the items on the menu are always Quit and Help.
   - Module documentation. A brief summary of each module in your code, what functions and classes are found there, which modules they import. Also, a brief overview of how the code fits together.
   - Cheats. I dont have time to solve all your puzzles, or acquire the skill to defeat your game. A good game should take hours to finish, but I want to see everything you did in a few minutes. Please provide walkthroughs for your puzzles, and genuine cheats (keys to press to get to the next level without earning it, an immortality mode, unlimited ammo, the BFG, *etc.*), so I can see your whole game in just a few minutes.
   - Acknowledgement of credit for any artwork, sounds, or code that is not your own. Failure to acknowledge such is plagiarism and grounds for academic disciplinary action. Even if the stuff you use is public domain, it is always nice to credit the place you got it. Whenever you grab something off the internet, just save the URL with the file and list all these in your acknowledgements.
   - Autobiographical info on the programmer. There is a wide skill range in this class in programming, and in game knowledge. Games will be judged individually, and based on what the student brings to the game. Some students may focus on the artwork, some on the game physics, some on the puzzles, some on trying out a completely novel game idea, *etc.* Needless to say, grading will be very subjective. However, I will attempt to discern how much effort went into the project. To that end, it is not inappropriate here to give some autobiographical information in the Game Manual about your history, what you found challenging, what was easy, how this differs from games youve done in the past, *etc.*

4. All games must have in-game documentation. Usually this takes the form of a help screen, traditionally accessed by pressing F1 or by choosing a Help menu item. The help screen should give much of what is in the users manual, but more succinctly.

5. Grading criteria. Notwithstanding the range of games I expect to see, here are some of the things I look for in evaluating a game (based on John Lairds criteria).

   (a) Is the game functional?

   (b) Manual and in-game documentation.
   - Is the documentation clear?
   - Is it entertaining and inviting?
   - Is there a back story to the game?
   - Does the game have in game instructions on keys and controls?

   (c) Non-trivial implementation.
   - Sound effects
   - Music
   - Background sound
   - Different difficulty levels
   - Multiple weapons with different behavior
   - Multiple opponents with different behavior
   - Interesting mix/match between weapons and opponents
   - Multiple levels
   - Tutorial level
   - Complex interactions between player and enemies
   - Complex properties of game pieces (health, shields, ...)
   - Two player game mode
   - Graphics unusual and engaging
   - Physics complex
   - Universe bigger than screen
   - Deep story
   - Network support

   (d) Bugs/Design Flaws.
   - Game crashes
   - Game locks up
   - Long load time
   - Bad controls
   - No ability to stay in game after one turn
   - Only one life
   - Difficult to tell if youre making progress
   - Collision detection problems (walking through walls, getting stuck in walls, *etc.*)

   (e) Game Play.
   - Impossible to win
   - Too easy to win
   - Just a move and shoot game
   - Rate of feedback and achievement not good
   - Only single level of goals
   - Not fun
   - Good mechanics for gameplay
   - Originality of game
   - Goals are well integrated and are appropriately rewarding
   - Feedback appropriately informs players of what is important
   - Dynamic difficulty adjustment
   - Judge gets hooked on playing the game
   - Judge shouts, screams, or laughs out loud while playing

6. Bottom line:
   Does this game show evidence of a substantial amount of work and learning on the part of the student?