Indecent Exposure Project Proposal

Team Source Force

May 2, 2016

Purpose of the proposed system

This game will provide our users with their daily dose of the absurd. The premise of the game plays off the idea of being trapped in a nightmare where you're completely naked. Our implementation aims to reflect this mood of playfulness in its 8-bit aesthetic, which also helps to keep the nudity as a narrative device rather than just a shock factor.

Game play starts with the user entering their dream world and realizing they are completely without clothing. The goal is to avoid being seen by peers and colleagues while searching for items of your clothing in a multilevel map of a high school. We plan on building three levels of the game, which correspond to the three floors of the school. Your character will follow a fairly linear path and use objects such as desks to hide behind.

Stakeholders

Our game is aimed at anyone who's ever had a bad dream about being naked in public. We believe our audience will be young adults who are Internet savvy, and are looking for light entertainment. The goal is for the game to be worth the time spent playing it.

System context

Our game will be made available as a cross-platform application for PC users. We won't be implementing it for use with any other consoles. We plan on using the SDL game development library for our game engine. This will mean using SDL2 and SDL2 image libraries in our code. Our program will be written in C++.

Behavioral Requirements

Here are a few examples of standard use cases that would be typical for an average game play.

The user starts the game: The game displays a menu with three save slots. If a slot is empty, it has a button to start a new game. If a slot has a save in it, it has a button to load it, and a button to delete it.

The user starts a new game: The game shows a short cinematic bit showing the character getting in bed and falling asleep, and then the user is put into the first level within the characters dream bubble.

The user loads a game: The user is put at the last checkpoint they reached before saving the game previously.

The user is playing the game: The system displays an environment from a top-down/isometric perspective. The user's character can move around the screen using the arrow keys or WSAD. If the character is seen by an NPC, the game restarts at the nearest checkpoint. The character must navigate the map and collect all items to win.

The user pauses the game: When the user hits escape, the game pauses and a menu appears. The only options are to quit the game, or unpause. (There is no save option, as the game auto-saves at checkpoints.)

The user passes a "checkpoint": The game saves the user's location and stats.

Value

We aim to create a piece of software that will provide a few laughs, as well as some moderately challenging game play. We feel that the relatability of this idea will make the game successful, as unplanned public exposure is a fear that can be understood at all ages. Although this game may sound trivial, there is consumer demand for the genre. Simple, comedic games are fairly popular (Shower With Your Dad Simulator 2015, Surgeon Simulator 2013, Goat Simulator, Who's Your Daddy, etc.).

Design Concept

Architectural Design Sketch

Our game will be an object-oriented program. Our main class will initialize our biggest classes. This will include our SDL_setup class that will handle all events, like I/O, and the rendering of the game's window. Our main class will also contain a function named GameLoop() that will run every frame. The gameloop will update an Environment object, that will contain all aspects of what is displayed on the screen. We will use sprite objects to manipulate images on the screen, and some sort of collision object will be a part of our Sprite class.

Main Dependencies

Before beginning to design our game we will need a C++ program that compile with SDL libraries without errors. This requires us to have SDL installed and set up properly. Then we can begin building the tools for our game, namely sprites and the controls to our gameplay. After that we will design a level to test with, and then we can work on a function-based AI. When we have working NPCs that can keep the game competitive we will design a few levels required to beat the game.

Assessment of work/risk

Due to our experience with SDL, I believe there is little risk to having a program set up to use it properly. We also have some experience creating a sprite class, so creating sprites with proper control shouldn't be too risky either. Our biggest challenges most likely lie with designing working collision detection and NPCs that work how we would like them to. The next challenge will be creating levels that are both challenging and enjoyable.