Final Project

GAM1514

Description:

-For the final you will be building a tower defence game using the knowledge and skills you have learned throughout the semester.
-There must be a menu system that has the following menus: splash, main menu, settings, level select, pause menu, game over and level complete.
-See the menu system flow chart on the last page
-The settings menu must load/save its settings to either a json or xml file
-You must have at least 2 animated textures in your game, they don't have to be complex animations, but should be implemented in a meaningful way. (ie explosion)
- No fixed tower locations. This tied to the requirement for pathfinding.
- No fixed paths. Levels can and should have obstacles loaded into force some paths to the enemy AI, but there will not be a hard coded path for them to walk along.
-The player must have a *default* (that number is your choice) number of lives, and must be able to gain and lose lives.
-There must be a lose state, the game over menu must be presented when the player loses all their lives
-There must be a win state, the level complete menu must be presented when the player completes a level
-The on-screen heads up display must use a bitmap number atlas to display time, score and lives remaining.
-There must be effective use of random numbers used throughout to vary the gameplay.
-The enemy AI must effectively use the A* path-finding algorithm to seek and destroy the main player
- The enemy AI must react appropriately if the A* algorithm fails to find a path to the main player
-There must be at least 3 different levels, each loaded from a file, you can (and should) use the level editor that you created in assignment 3 to create these levels.
-To get maximum marks, you can keep track of the top 5 high scores, you should have a menu on the main page to display these high scores. The highscores should be saved to either a json or xml file.
-Memory management is incredibly important, make sure you are properly creating and deleting your objects.

-You may develop this assignment on either *windows* or *mac*, however it must compile for BOTH platforms.
- Remember the style guide - readable code matters.

Core gameplay features:
- Fast forward. Give the player the ability to speed the game up x2, then x5 then back to x1 again. update the button texture to clearly show the current speed. Watch out for collision detection failures at high speed.
- Persistent terrain damage. When an explosion goes off, the terrain at ground zero should be permanently damage (example: dark scorch mark). This may also included destructible terrain.

- Tower upgrades. Three aspects of towers can be upgraded - range, damage, firing speed. Each tower can have 5 upgrades applied to it using resources collected. Allow the player to apply the upgrades they want, and devise a way to communicate visually those upgrades to the player.

Note - you are not being marked on your artistic abilities. Programmer level art is acceptable.

Due date:

010 - December 13th (end of class)
020 - December 12th (end of class)
030 - December 13th (end of class)
040 - December 12th (end of class)

Submission:

Create a tag of your assignment on Github and I will download the assignment from Github via that tag

Comments:

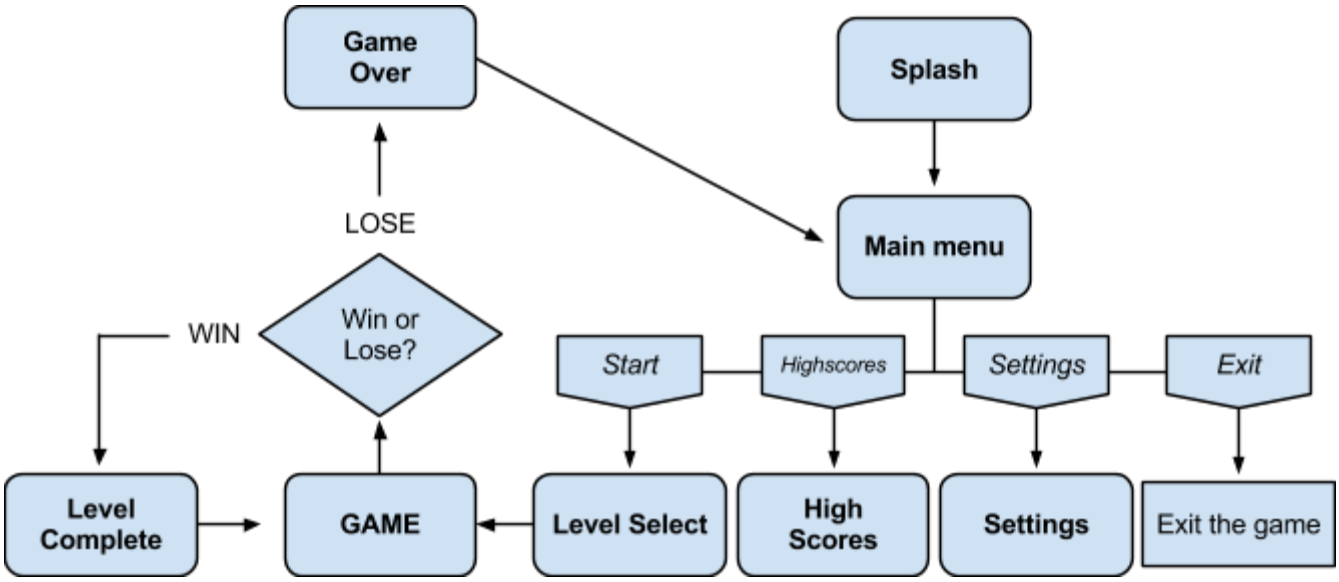
Grading:

Multi-platform: The game compiles on both Windows and OSX and doesn't crash	/ 5
Content: There are 3 unique levels to play, each loaded from a file	/ 30
Content: There are 2 or more animated textures	/ 10
Gameplay: Feature 1: Fast forward	/ 10
Gameplay: Feature 2: Persistent terrain damage	/ 15
Gameplay: Feature 3: Tower Upgrades	/ 20
GUI: All the required menus are complete and flows as expected	/ 30
GUI: The HUD displays the time, score and lives using a bitmap font atlas	/ 10
Logic & GUI: The settings menu saves and loads its settings from an JSON or XML file	/ 10
Logic & GUI: The game over menu is displayed when the player fails a level	/ 10
Logic & GUI: There is a level complete menu that is displayed when th player completes a level	/ 10
Logic: The player has a default number of lives that can be both gaine and lost	/ 5
Logic: The enemy AI effectively uses the A* pathfinding	/ 20
Logic: Random numbers are effectively used to vary the gameplay throughout the game	/ 10
Extra GUI: There is a high scores menu and the top 5 high scores are being saved/loaded from either a JSON or XML file	/ 25
Memory Management: Memory is properly allocated and released	/ 20
Coding style: The student followed the programming style guide outlined on Blackboard	/ 10
Total	**/ 250**

Weighting:

This assignment is worth 15% of your final mark

Menu Flow:


Game Over
Splash
LOSE
Main menu
WIN
Win or Lose?
Start
Highscores
Settings
Exit
Level Complete
GAME
Level Select
High Scores
Settings
Exit the game