EGP-410

Assignment 6 - Final Assignment

- Basic Game Play –

**Due to:** Tuesday, Midnight of December 2rd 2014

**Presentation:** Wednesday, class of December 3th 2014

This assignment goal is to create a Pac-Man clone game using everything you have learned so far.

In this assignment you are required to use your previous assignment as the starting point and enhance it to be used as a dungeon type game.

The assignment will be carried out in pairs – choose between your last submitted assignments, copy it to the new assignment folder on SVN and set it as the initial submission point for this.

Avoid copying any temporary MSDev files into the SVN (such as debug, obj, sdf, etc…)

The assignment will have presentation of the game in front of the entire class at the first hour of the class, and selecting the best game!

**Game Specifics:**

* The game will contain the player character and AI entities (the hideous **Grue**) within a multi-level map (as per our previous exercise) by using the **hierarchical maps and path finding** from the previous assignment
* The goal is to gain as higher score as possible by collecting coins without being eaten by the **Grues**.
* The player character will be able to eat **Grues** after eating ‘**all mighty candy’** item that will be scattered within the maps sparsely at strategic points and can **re-generate after 60 seconds.**
* Other bonus items and features are welcomed to keep the game.
* The game should contain user score
* The game should have help menu
* **Game specifics:**
  + **Coins** – will populate the map and will appear every several tiles as points for the player. These points should be placed using a random distribution algorithm with control over percentage.
  + The **all mighty candy** – will allow the player to destroy the enemies within 10 seconds after eating it. The player looks needs to reflect it (change color, start blinking at the last 3 seconds)
  + The **Grues** roaming around within the dungeon:
    - These hideous creatures will switch between patrol and wander behaviors until spotting the player (line of sight within distance) at that point they will chase or flee according to the player’s state (did he eat the **all** **mighty candy** lately).
    - Looks should to reflect their state: chase, wander/patrol, evade, other… - you should be fluent at explaining and demonstrating that.
    - Velocities should be **slightly** slower than the player’s.
    - Should spawn 10 seconds after being eaten by the player at their initial spawn point (different for each)

**Assignment outcome (what you need to provide):**

1. Editor - enhanced code and executable
   1. The editor will enable maps creations and linking
   2. The editor will enable spawning points for AI and the player
   3. The editor will enable **all mighty candy** points
2. The game itself
3. **Help** menu for both applications (pressing F1 or H).

**Guidelines**

1. All requirements for the editor, maps generation and content placement should follow the ones from the last exercise (and be enhanced).
2. You are to create several multi maps chained between each other with transition points that can be used by both AI and the player
3. Implement **State Machine** and use this technique to create behavior and control over the AI and the player’s character.
4. Connections exist between two neighbors cells sharing an edge – a diagonal connection is not allowed and all moves of both player and AI should not use diagonal moves.
5. The enhanced A\* algorithm should be used for this exercise (works with linked maps)
6. Allow ‘debug’ mode that shows the entities’ state and their path planning.
7. Enable easy control over entities and player speed and other properties.
8. Allow ‘invincible’ key that will prevent player from dying without changing any other rule of the game.
9. ‘ESC’ quits the program

**Grading**

Each student will be required to present his code changes and talk about his contribution to the overall.

The grading scheme will be as follows:

* 10 points – coding standards, using messaging system, correct OOP design
* 20 points – editor implementation for content creation
* 20 points – state machines implementation and usage
* 30 points – overall game implementation, game manager, and functioning
* 20 points – visualization and polish