Peer-Review 1: UML

<Valerio Xie>, <Zhitao Xu>, <Ziheng Ye> <Marco Zheng>

Gruppo <GC 23>

Valutazione del diagramma UML delle classi del gruppo <GC 13>.

Lati positivi

- La struttura del UML e' completa e coerente e presenta una serie di classi che coprono tutti le componenti che compongono il gioco; nel complesso l'attenzione al dettaglio e' ottima.
- L'idea di utilizzare un codice identificativo univoco per le Tiles e' un idea ottima, che risulta comoda per gestire il problema delle 3 differenti tipologie che ogni tessera puo' assumere.
- La presenza di una grande varieta' di metodi che faciliteranno un'implementazione fedele e compiuta di tutte le possibili azioni che i giocatori compiono nel gioco.
- Il fatto che il nickname sia unico e accessibile solamente in lettura e' una ottima idea che limita la possibilita' di commettere errori nella duplicazione del nickname dei giocatori e impedisce la nascita di bug indesiderati.

Lati negativi

- Un dettaglio che abbiamo notato e che potrebbe permettere di migliorare il progetto e' di fare utilizzo dei Design Pattern, Strategy nella selezione delle CommonGoals, invece di usare una enumerazione, per l'aggiunta di estendibilita' che questo Pattern dona; ovviamente nello sviluppo di questo gioco si assume che non ci sara' il bisogno di aggiungere ulteriori CommonGoals da quelle che ci vengono gia' fornite, ma vista l'opportunita' di imparare a utilizzare un framework cosi' comune e fatto a pennello per questa situazione, ci e' sembrato il caso di consigliarne l'implementazione.
- Nella classe ItemTile una serie di metodi che servono al settare e resettare gli attributi di adiacenza che sono booleani;
- Il nostro gruppo ha deciso per l'implementazione di un metodo a cui dato come parametro una tessera restituisce tutte le tessere adiacenti ad esso, che si e' rivelato molto comodo e piu' intuitivo, pertanto consiglieremmo una simile implementazione.
- La scelta di aggiungere il tipo VOID ai types, nell'Enumerazione e' stata una scelta che e' stata presa dal nostro grupppo all'inizio della progettazione; abbiamo fatto una scelta contro questa opzione seguendo il consiglio del tutor durante un laboratorio, e perche' avrebbe una creato una separazione che portava a piu' ordine e separazione mentale;
- Un'altra differenza e' stata la scelta di utilizzare un doppio array, mentre il nostro gruppo ha deciso di andare per un implementazione piu' estendibile utilizzando un ArrayList di

ArrayList, i cui metodi disponibili non sono strettamente necessari, ma portano a una maggiore facilita' nella realizzazione di certe operazioni che ci vengono fornite grazie alla libreria presente default della struttura dati.

- Il nostro gruppo ha scelto di aggiungere un concetto chiamato Square che definisce il concetto di "posizione" all'interno della LivingRoom e della BookShelf, aggiungendo un ulteriore strato di complessita', che rende i metodi piu' lunghi e verbosi nel codice, ma che permette un tradeoff in termini di ordine e pulizia nel codice;

Confronto tra le architetture

- I maggiori punti di discrepanza tra le due architetture sono la presenza di idee e dettagli minuti che rendono piu' funzionale il nostro codice, sicuramente verra' aggiunto il codice identificativo che rende unica ogni tessera, che risolve il problema posto durante laboratorio, di come gestire le 3 possibili immagini, che ogni tipo puo' assumere e la conseguente problematica di come gestire la View client-side in modo che fosse coerente per tutti i giocatori.
- La caratteristica di rendere il nickname giocatore unico e accessibile solamente in lettura e' una fantastica idea che verra' aggiunta al nostro progetto perche' coerente e sensata, portando a un decente guadagno in termini di utilita'.