Relazione Progetto Game Programming

Membri:
Mattia Baracetti 147742
Nicola Davico 148659
Riccardo Marin 147513
Davide Mirabelli 148184

Composto da 5 scene, 3 narrative e due di gioco.

L'idea attorno alla quale è stato costruito il gioco è quella di un insieme di mini-giochi legati tra di loro da una semplice linea narrativa. La strutturazione in scene alterna scene di narrazione a parti di gioco vero e proprio, andando ad aumentare le componenti di interattività nel corso del tempo.
Il gioco si apre con una scena di dialogo tra il personaggio ed una voce non determinata al fine di dare una semplice introduzione narrativa.
Il primo gioco è un **gioco del 15**: le tessere vengono generate come elementi singoli all'interno di una griglia 16x16, con i numeri in ordine da 1 a 15 ed una casella vuota e poi mescolati dalla funzione shuffle(); è stato scelto di separare le funzioni per semplificare la fase di test del videogioco, permettendoci di velocizzare la fase di gioco. Il movimento delle tessere può essere effettuato sia con i comandi WASD che attraverso l'uso delle frecce presenti sul display. Ad ogni attivazione del movimento, una volta svolte tutte le operazioni necessarie, viene richiamata la funzione **isRisolto()** che va a verificare se le tessere siano o meno posizionate nell'ordine corretto. In caso affermativo, la scena termina e, dopo un'ulteriore scena di dialogo, inizia il secondo gioco
Il giocatore si trova in un **dungeon** composta da varie stanze attraversabili, con al loro interno vari nemici. L'idea implementativa principale è stata quella di sfruttare la gestione della mappa da tiled unita ad un ambiente fisico a gravità 0: abbiamo deciso di utilizzare questa gestione per avere una migliore integrazione tra gli elementi in movimento e quelli statici (ad esempio, il muro appare come sopra il personaggio se quest'ultimo è nel lato basso di una stanza). Grazie alla gravità 0 ci è stato possibile utilizzare gli impulsi sui corpi fisici per muoverli, che combinato al bounce=1 permette di mantenere il movimento costantemente.
Un workaround adottato per mantenere tutti gli oggetti all'interno della mappa è stato quello di usare 5 oggetti per creare il personaggio: uno con il corpo fisico chiamato **idle** e quattro con le animazioni per i quattro movimenti (attivabili anche qua con WASD e frecce su tastiera e frecce su schermo); abbiamo dovuto procedere in questo modo a causa delle attuali limitazioni di tiled, siccome un elemento può al momento contenere solo un'animazione.
Il 'teletrasporto' nella stanza segreta avviene con un meccanismo a tempo legato alla collisione con la scala, permettendo così di concludere la collisione e avere una meccanica di gioco di attesa che si aggira attorno al secondo.

Ogni chest contiene una casella di testo: viene attivata all'urto con la chest e portata in fadeOut quando il personaggio si allontana. La gestione del testo è unica, basata sul nome della chest, con solo una serie di if annidati che modificano la *quote*.
Il meccanismo di attacco del boss è quello più interessante tra i vari nemici: vi è infatti un listener che verifica se il personaggio è nel raggio di 50px dal nemico. In quel caso gli viene applicata, dopo due secondi, una velocità lineare che lo fa muovere verso il personaggio. Il delay applicato permette al giocatore di riuscire a muoversi ed evitarlo, garantendo comunque un movimento abbastanza fluido per il nemico. Vi è poi la possibilità di terminare la fase di inseguimento allontanandosi dal boss, che dopo un attimo di attesa tornerà alla posizione iniziale (al centro della stanza).
In generale ogni gruppo di nemici è delimitato all'interno di stanza unica utilizzando dei muri invisibili che disattivano l'evento collisione nel caso urtino con idle, mentre mantengono il loro corpo fisico nel caso l'urto avvenga con un nemico.
Per uscire dal dungeon viene valutata se la chiave, 'sbloccata' in una chest, è visibile.
Abbiamo scelto di usare questo metodo per la semplicità di utilizzo: al contatto con la cassa designata, viene cambiato il valore alla proprietà di key **key.isVisible,** con l'immagine già caricata nella creazione della scena.

Abbiamo riscontrato alcune problematiche che non siamo riusciti a risolvere per ragioni di tempistica, la più evidente, è l'immagine facente parte dell'oggetto "hearts", che nel caso in cui si prenda la vita bonus presente nella stanza in fondo a sinistra, essa (l'icona) verrà coperta dalla mappa stessa, risultando non presente sullo schermo (non è in primo piano).