Anexa 1

Titlu proiect (Spelunky)
Autor(Tudusciuc Cristian-Rafael)
Grupa(1210A)

Povestea jocului:

De-a lungul istoriei au existat multe evenimente majore iar cea mai memorabila este goana dupa aur ce a impins multi aventurieri in pesteri adanci si periculoase. Dintre toti acestia se remarca Jake, dar de ce? Precum marele Indiana Jones a descoperit comori de neinchipuit tot asa s-a nimerit ca Jake sa se gaseasca in cea mai bogata pesterea din lume ce se spune ca ar fi magica, oricine poate sa intre dar nu orcine reuseste sa iasa. Va reusi Jake sa scape impreuna cu bogate comori sau va sfarsi ca toti ceilalti?

Prezentare joc:

Campanii pentru un singur jucător în care jucătorul trebuie să învingă creaturile ostile pentru a avansa in adancimile Pesterii spre singura iesire ce trece la nivelul următor.

Reguli joc:

Jocul implica o dinamica intre elementele de agilitate si cele de gandire strategica. Jucatorul are la dispozitie biciul si 5 bombe consumabile cu care trebuie sa gaseasca sau sa croiasca drumul catre iesire. Inamicii sunt imprastiati peste tot si pot fi distrusi prin atacuri cu biciul si explozii sau prin saritura pe acesta. Exploziile pot distruge elemente de teren din joc si vor scadea din viata orcarei creaturi din jur inclusiv a jucatorului.

Pentru a isi creste numarul de vieti, jucatorul poate cauta si salva “domnisoara in pericol” ce se gaseste in fiecare nivel. Tot de-a lungul nivelelor exista tepuse ce pot omori jucatorul daca cade pe ele.

Jocul se sfarseste prin victorie cand jucatorul reuseste sa treaca 4 nivele sau prin infrangere cand jucatorul ramane fara vieti.

Personajele jocului:

- **Jake** este protagonistul și jucătorul-personaj. El este de obicei descris ca fiind tipul eroic, dar vesel, salvând deseori domnitele in nevoie. El are capacitatea de a

manui biciul si de a arunca bombe.

- **Sarpele** are o fire agreesiva si isi iubeste spatiul personal, este in stare sa muste pe oricine se apropie de el.
- **Liliacul** pretuieste somul iar apropierea de acesta il va face sa se indrepte direct spre sursa de zgomot.

Tabla de joc

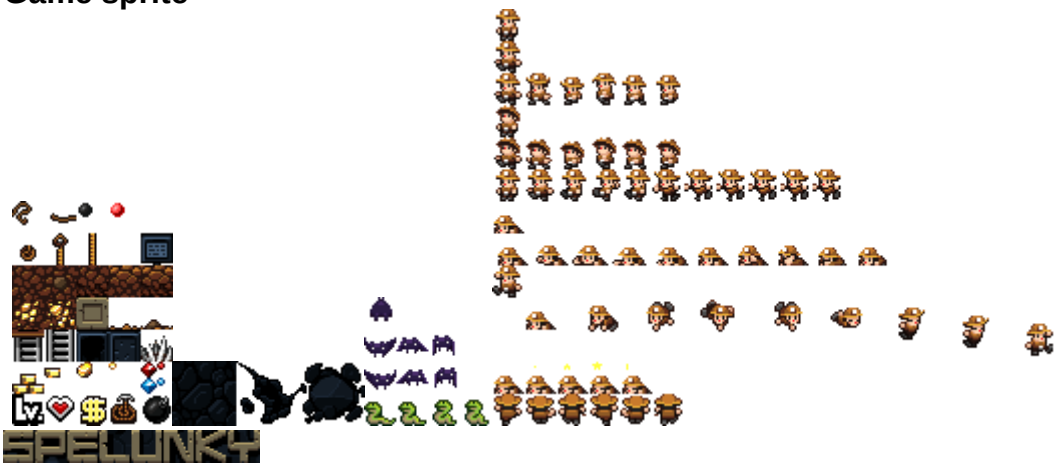
- Componente pasive: pamantul, peretii: ce nu pot fi distrusi, iesirea: usa prin care se trece la urmatorul nivel
- Componente active (Liliecii, serpii etc) si proprietatile acestora
- Comonente pasive: tepusii: pot ucide jucatorul, scarile: se poate urca pe ele
- Structura tablei de joc generata semi aleatoriu, avand sectiuni predefinite ce sunt alaturate aleatoriu pentru a asigura un drum continuu de la intrare pana la iesire

Mecanica jocului

- combinații și interacțiuni între multiplele elemente ale jocului precum jucator si inamici, inamici si uneltele jucatorului, explozii si entitati, unelte si teren(pot adauga elemente in teren sau sterge).
- jucatorul are la dispozitie mai multe taste si metode de a se deplasa prin nivel precum:
  -saritura variata in funtie de timpul apasarii tastei Z
  -posibilitatea de a se agata de colturi pentru a ajunge mai sus sau de a nu primi lovitura de la cazatura de la inaltime mare.
  -se pot arunca bombe cu viteze diferite sau se pot plasa jos
  -poate folosi un bici pentru a lovi inamici
  -jucatorul se poate ghemui pentru a se aseza pe un colt

- Liliacul urmareste jucatorul
- Sarpele se misca pana da de perete sau gol

Game sprite



Descriere fiecare nivel

Tutorialul are semne ce explica fiecare buton si ajuta jucatorul sa inteleaga mecanicile si modul in care se calculeaza scorul final

Nivelele din joc sunt aleatorii ceea ce forteaza jucatorul de a isi imbunatati aptitudinile de a parcurge nivelul, nu poate invata o organizare predefinita a nivelului, astfel jocul devie provocator, cu fiecare inaintare de nivel jucatorul poate pierde vieti, bombe sau funii astfel creste presiunea si limiteaza optiunile disponibile de parcurgere a nivelului

Descriere meniu

Organizeaza pe ecran 3 elemente: titlul jocului pe centru in partea de sus, tabela de scoruri in partea din dreapta si optiunile in partea din stanga.

Optiunile sunt: resume, Toggle music, reset highscore, intra in tutorial, termina parcurgerea curenta a jocului sau optiunea de a inchide jocul.

Bibliografie

https://www.javier8a.com/itc/bd1/articulo.pdf
https://github.com/RCTd/Spelunky
https://tinysubversions.com/spelunkyGen/

Descrierea scheletului proiectului

Diagramă proiect

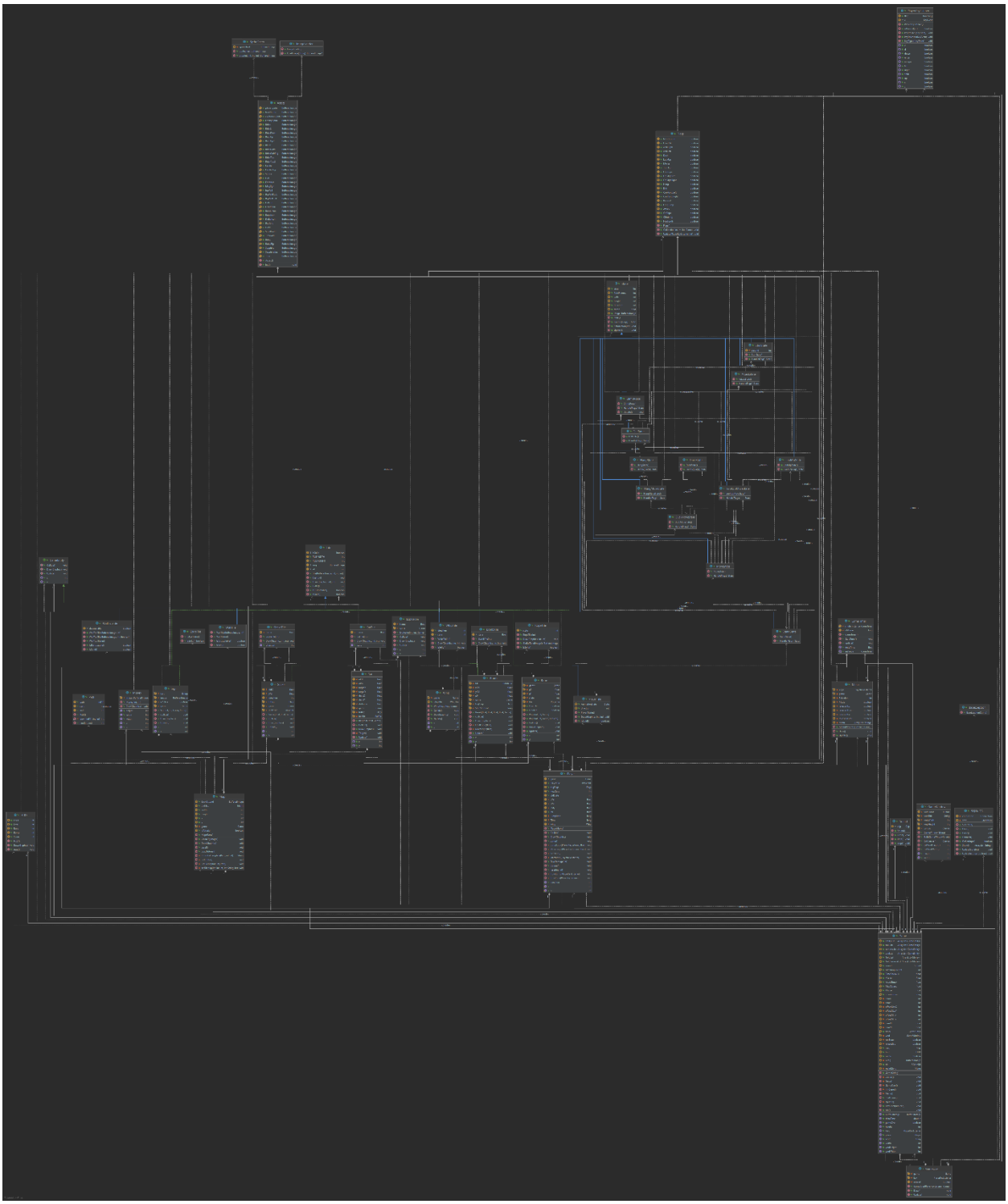


Figura 1. Diagrama UML de clase (https://github.com/RCTd/Spelunky/blob/master/Package%20GamePakage.png)

În figura de mai sus se poate observa diagrama de clase UML (sunt vizibile toate elementele intrucat diagrama era deja aglomerata). Fiecare clasa are rolul ei în cadrul proiectului și la rândul său instanțiază/moștenesc alte clase.

După cum s-a discutat în laboratorul anterior, trebuie să existe cel puțin o metoda
main în una din clase pentru a fi apelată la rularea programului (doar o metoda main poate fi rulata la un moment dat). Așadar, execuția programului începe din clasa Main, metoda main. În cele ce urmează sunt prezentate toate clasele, începând cu Main, precum și firul logic al acțiunilor.

Descriere clase proiect

Main

După cum se poate observa din diagrama prezentată mai sus, această clasă creează o instanță de tipul Game, iar mai apoi pornește fluxul activităților prin apelul metodei StartGame() din Game.

Game

Rolul acestei clase este de a face inițializările (de exemplu de a crea instanțe pentru jucători/inamici, pentru a inițializa harta, etc.), dar și de a menține interfața grafică la zi cu cea ce se întâmplă în joc. Această clasă implementează interfața Runnable pentru a avea comportamentul unui fir de execuție (thread). În momentul apelului metodei StartGame() se instanțiază un obiect de tip Thread pe baza instanței curente a clasei Game.

Orice obiect de tip Thread trebuie să implementeze metoda run() care este apelată atunci când firul de execuție este pornit (start()). Această metodă run() inițializează jocul prin crearea unei instanțe GameView, iar mai apoi controlează numărul de cadre pe secundă printr-o buclă while și "pregătește" noua scenă (Update()) pe care o va desena pe interfața grafică (Draw()).

Clasa Game contine liste de entitati si de prada. Prin parcurgerea lui enityList se pot accesa metodele Draw Update si Collide a fiecarei entitati din joc. Lista removeList ajuta la stergerea unei entitati avand in vedere ca in timpul parcurgerii listei de entitati nu putem sterge din lista parcursa un element al listei asa ca se adauga la o lista de stergere ca dupa terminarea parcurgerii sa se stearga elementele necesare.

De asemenea, Game se ocupa cu camera jocului ce urmareste jucatorul calculand translatiile necesare a imaginilor randate.

Metoda newLevel elbereaza listele pentruu a face loc urmatorului nivel si determina daca se va crea un tutorial sau un nivel, pe langa resetarea listelor se

reseteaza si jucatorul, offset-urile pentru camere si eventual se reseteaza HUD.

Metoda Update() actualizează starea jocului (de exemplu: modifica poziţia jucătorilor pe baza tastelor apăsate, schimbă poziţia inamicilor folosind chiar tehnici de inteligenţă artificială, crează diferite tile-uri (dale), etc. Dacă ne referim la un joc 2D care are harta pătrată, aceasta este împărţită în celule de aceeaşi dimensiune(ca o tablă de sah), iar aceste celule sunt tile-uri care pot avea texturi diferite, pot reprezenta diverse obiecte (de exemplu proiectilele unui tanc, acesta poate fi un tile ca şi restul elementelor de pe hartă dar care are fundalul transparent).

Metoda Draw() va desena pe interfaţa grafică modificările făcute de metoda Update(). Interfaţa grafică este un canvas, făcând o analogie cu realitatea, poate fi considerată o pânză pentru desen, pe care sunt desenate diverse obiecte.

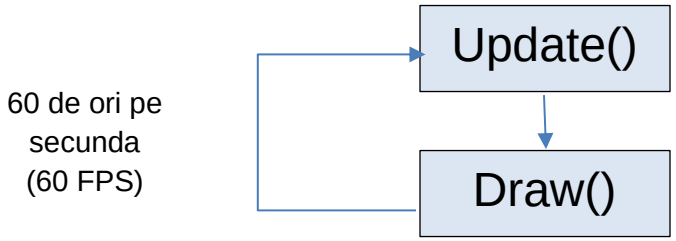


Figura 2. Bucla metodei run() din clasa Game

GameWindow

După cum discutam mai sus, metoda run() a clasei Game crează o instanţă a clasei GameWindow. Această clasă este responsabilă cu reastra în care vor fi desenate obiectele pe un canvas (se poate observa ca avem o metoda GetCanvas care întoarce canvasul). De asemenea, avem un obiect JFrame care permite desenarea de butoane, controale, textbox-uri, etc, dar poate conţine şi un canvas în care pot fi desenate diverse obiecte folosind texturi. Dacă de exemplu dorim să avem un meniu pentru joc,
atunci sunt necesare mai multe obiecte JFrame, unul pentru meniu în care sunt desenate butoanele, etc şi unul pentru acţiunea jocului, iar în funcţie de interacţiunea utilizatorului acestea vor fi schimbate între ele.

Tile

Această clasă reţine informaţii despre tile-urile (dalele) din joc. În clasa Game există o instanţă a clasei Tile, deoarece clasa Tile conţine un vector cu obiecte tot de tipul Tile, aşadar în acest vector vor fi stocate toate „dalele/tile-urile„ din joc în aşa fel încat acest vector poate fi parcurs şi pentru fiecare element se apelează metoda Update() şi apoi Draw(). De asemenea, clasa Tile mai

reţine şi câte o instanţă pentru fiecare sub-tip de
tile (GrassTile, MontainTile, TreeTile, SoilTile şi WaterTile). În funcţie de hartă, acel vector de tile-uri va conţine tile-uri de acest tip.

BatTile,SnakeTile,BombTile,RopeTile,WhipTile,PlayerTile,EdgeTile, WallTile,WallTopTile

Toate aceste clase extind clasa Tile, preluând comportamentul acelei clase sau putând să-l suprascrie folosind @Override. Constructorul acestora primeşte ca parametru un ID şi apelează constructorul clasei de bază, adică al clasei Tile, care primeşte ca parametru un membru static al clasei Assets.

Assets

Conţine câte un membru static de tip BufferedImage pentru fiecare tile/dală, chiar şi pentru jucători/inamici. In aceşti membri sunt stocate imaginile, adică texturile acestora care vor fi desenate prin apelarea metodelor Draw() din fiecare clasa tile apelate din metoda Draw() din clasa Game. Pentru a putea instanţa aceste obiecte statice trebuie să
decupăm multe dintre texturi din acea imagine. Pentru a face asta se foloseşte o instanţă a clasei SpriteSheet în metoda Init(). Metoda folosită din SpriteSheet este crop(x, y).

SpriteSheet

Constructorul acestei clase primeşte imaginea, iar clasa conţine 2 membri constanţi (final in Java) pentru înălţimea, respectiv lăţimea texturilor (trebuie sa aibă toate aceleaşi dimensiuni). Metoda crop(x, y) primeste doi parametrii, x specifică pe ce coloană se află textura pe care dorim să o decupăm în imaginea cu toate texturile, iar y specifică linia. Pentru a afla efectiv poziţia în imagine se determină pixelii de unde încep texturile de interes prin înmulţirea liniei/coloanei cu înălţimea, respectiv lăţimea texturilor. Astfel se obţine colţul din stânga sus al texturii, iar pentru a afla celelalte colţuri se folosesc înălţimea, respectiv lăţimea.

ImageLoader

Înainte de a extrage fiecare textură, imaginea cu toate texturile trebuie să fie citită din memorie, iar pentru asta se foloseşte clasa ImageLoader cu metoda statică LoadImage(path) care primeşte ca parametru calea către imagine în memoria calculatorului.

Money

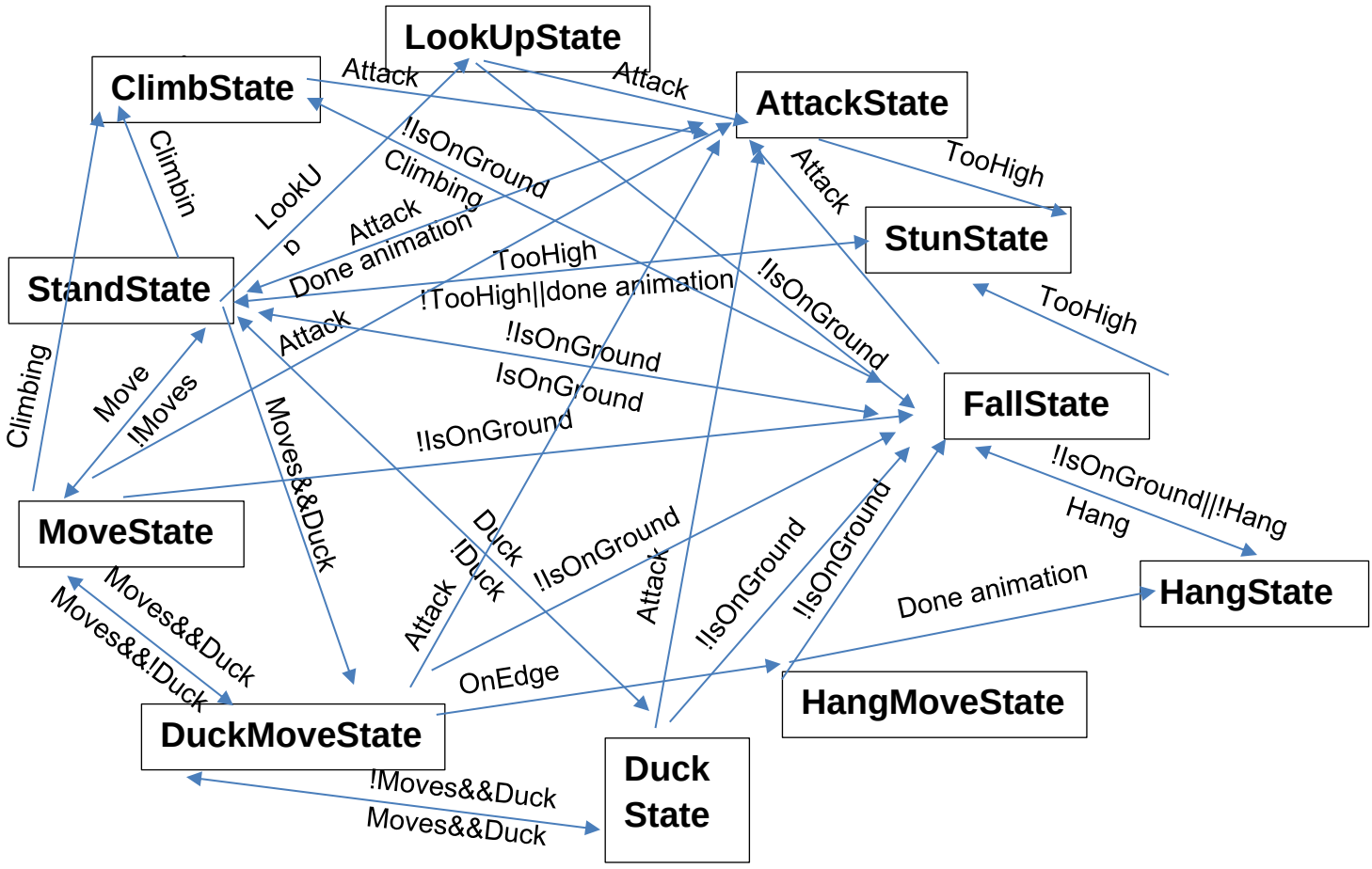
Clasa retine o pereche de coordonate x,y si isi atribuie aleatoriu o valoare asociata de o imagine a unui tip de valuta (Gold,Ruby, Sapphire,etc). Obiectul creat va fi adaugat intr-o lista pentru a usura adunarea de scor a jucatorului.

State

Pentru implementarea modelului stare a fost conceputa clasa abstracta State ce va fi implementata de starile jucatorului, prin aceasta se gestioneaza animatia jucatorului. In metoda Update() se actualizeaza frame si imaginea corespunzator decupand din playerSpriteSheet cadrul necesar. Metoda Handle() va fi suprascrisa de fiecare stare in parte pentru a determina conditiile de trecere in urmatoarea stare.

AttackState,ClimbState,DuckMoveState,DuckState,FallState, HangMoveState, HangState, LookUpState, MoveState, StandState, StunState

Precum este scris mai sus aceste clase gestiioneaza animatiile jucatorului trecand dintr-o stare in alta prin suprascrierea starii curente cu o noua instanta de stare. Graful de fluenta respecta urmatoarea schema:



Path

Scopul acestei clase este de a genera o matrice de 4x4 ce sa reprezinte drumul de la inceputul pana la sfarsitul nivelului.

Fiecare celula a matricii reprezinta tipul de camera ce va fi pus in joc:
-camerele de tipul 1 au iesiri in dreapta si in stanga
-camerele de tipul 2 au iesiri in dreapta, stanga si in jos
-camerele de tipul 3 au iesiri in dreapta,stanga si in sus

Map

Contine matricea nivelului si metodele de creare a tutorialului si de generare a nivelelor precum si Imaginea de fundal intrucat trebuie sa fie de dimensiunile nivelului.

Pentru generarea tutorialului se citeste dintr-un String caracter cu caracter iar in funtie de caracterul citit se plaseaza fie Teren(1), Scara(L), Platforma scarii(P) sau Iesirea(9). Marginile nivelului sunt create ca elemente nedestructibile.

Restul elementelor (inamici, semne si prada) sunt predefinite si adaugate in liste ale jocului.

Generarea unui nivel se face folosind metoda RoomGetTemplate ce primeste un Path pentru a selecta aleatoriu un string ce la fel ca la generarea tutorialului determina unde va fi plasat teren, scari,iesirei, tepuse, lucru de care se ocupa metoda SetTileMap ce primeste un Path si String-ul ce reprezinta template-ul camerei, fiecare caracter poate insemna:
-0:Gol, unde se verifica dalele de deasupra si de dedesupt pentru a modifica sau a adauga elemente de estetica
-1:100% sansa sa fie Teren
-2:50% sansa sa fie Teren
-L:Scara
-P:Platforma scarii
-7:30%sansa sa fie Tepuse
-9:Intrarea daca camera se afla in partea de sus a nivelului sau Iesirea daca se afla in partea de jos

GameTimer

Un Singleton ce este folosit drept cronometru pentru a calcula distantele parcurse de entitati si perioada de la ultimul cadru

Flags

Permite manipularea unor variabile de tip boolean ce semnaleaza diverse coliziuni sau actiuni catre entitati pentru a putea fi tratate.

HUD

Contine cate un counter pentru numarul de vieti, numarul nivelului curent, numarul de bombe si funii ramanse si valoarea scorului curent

Desenarea textului se face cu un font personalizat la locatii absolute pe ecran alaturi de iconite corespunzatoare seemnificatiei fiecarei valori.

GameOver

Introduce in baza de date scorul la care s-a ajuns si afiseaza un mesaj de sfarsit alaturi de scorul curent. Ofera posibilitatea de a iesi din joc.

Menu

Organizeaza pe ecran 3 elemente: titlul jocului pe centru in partea de sus, tabela de scoruri in partea din dreapta si optiunile in partea din stanga.

Pentru calcularea pozitiilor se introduc optiunile intr-o lista si se determina lungimea listei si latimea(cel mai lung sir), evidentierea optiunii selecate se face printr-o variabila ce poate fi modificata prin sageti si selectata prin tasta enter.

Optiunile sunt: resume, Toggle music, reset highscore, intra in tutorial, termina parcurgerea curenta a jocului sau optiunea de a inchide jocul.

Bat

Are doua stari: de asteptare si de urmarire. Liliacul asteapta pana jucatorul este sub liliac la o distanta de macar 100px dupa care urmareste playerul fara a trece prin teren, in caz de coliziune cu terenul liliacul se intoarce la pozitia precedenta de pe axa cu coliziunea, in caz de coliziune cu una dintre uneltele jucatorului liliacul moare. Pentru coliziunea cu jucatorului daca jucatorul este deasupra liliacului atunci liliacul moare altfel liliacul loveste jucatorul.

Snake

Se misca in stanga sau dreapta pana de de perete sau de gol si are posibilitatea de a cadea in jos daca ii dispare terenul de sub picioare, coliziunea cu jucatorul si uneltele acestuia este la fel ca la liliac.

Bomb

Bomba ricoseaza cand loveste un perete si isi injumatateste din viteza pe y cand ajunge la pamant, ea creaza o explozie dupa 2 secunde. Jucatorul o poate arunca in sus, drept sau doar sa o plaseze in funtie de ce viteze initiale transmite construcorului

Explosion

Trece prin animatia sa iar la final distruge tot ce gaseste in jur.

Rope

Are starea de urcare si starea de coborare. Poate urca si cobora pana la o inaltime maxima de 8 patrate. La urcare daca se loveste de teren se opreste si continua prin coborare, la coborare lasa in urma funie care se comporta la fel ca scara iar daca se loveste de teren dispare.

Whip

Apare doar in cadrul atacului si este in spatele jucatorului la inceputul atacului dar in fata lui la sfarsit.

Sound

Gestioneaza partea audio a jocului prin incarcarea coloanei sonore si metodele de loop stop si close.

SQLiteDB

Gestioneaza partea de baze de date prin metodele Connect, Close, UpdateSettings, UpdateScore,GetSettings si Show ce returneaza o lista cu scorurile in ordine descrescatoare.