Guza’s Journey

Guriuc Vlad-Ionut
1207B

Proiectare context si descriere

Acțiunea jocului este fixata pe un taram indepartat si mistic intr-o perioada medievala in care oamenii, de la mic la mare, erau invatati sa stapaneasca, daca erau inzestrate, farmece, ritualuri si artele magiei pentru a face lumea un loc mai bun si pentru a combate spiritele întunecate. In acest taram peste care domnește cuvantul magiei, o gasim pe Guza. Guza este o tanara vrăjitoare extrem de talentata, desteapta si mai ales curajoasa, iar cel mai bun prieten al ei si totodată mana sa dreapta este Mushu, care se trage dintr-un soi rar de dragoni pitici. Într-un moment de neatentie Mushu a fost rapit de catre Zarus, Cavalerul Întunericului, pentru a-l tine drept trofeu de vanatoare. Acesta l-a dus in Gradinile Intunecate unde il tine captiv, pazit de animale vrajite, spirite si bestii. Guza este dispusa sa faca orice e nevoie pentru a-l invinge pe Zarus si pentru a-l salva pe Mushu si intra in Gradinile Intunecate, incumentandu-se intr-o aventura fara precedent. Jocul incepe dupa ce Guza intra in gradini, iar aceasta trebuie sa adune nestemate pentru a-si mari viata sau viteza si sa invinga inamicii pentru a creste scorul necesar trecerii la urmatorul nivel. La ultimul nivel va trebui sa il infrunte pe Zarus si sa-l salveze pe Mushu.

Proiectare sistem

Jocul este creat in stil PixelArt, cu vedere de tip three quarters asemanator Legend of Zelda, cu camera centrata in mod permanent pe eroina. Eroina se afla in permanenta pe mijlocul ecranului iar mapa este generata in jurul ei in functie de tastele apasate de utilizator. In acest joc, utilizatorul se poate bucura de conceptul “Fog of War”, avand un cerc de lumina care se intuneca spre marginile ecranului, ce are centrul situat in pozitia curenta a eroului.

Controale in timpul jocului:

W-deplasare eroina in sus;

A-deplasare eroina la stanga;

S-deplasare eroina in jos;

D-deplasare eroina in dreapta;

SPACE-atac de tip aruncare proiectil flacara ce se deplaseaza pe directia curenta a eroului(pot fi generate mai multe flacari concomitent pe harta);

Controale pentru meniu principal:

W,S- navigare intre optiunile disponibile sus-jos(New Game, Load Game, Quit;

ENTER-accesare optiune curenta;

OBS: Aceste comenzi sunt valabile si pentru ecranul aparut odata ce jocul este pierdut sau castigat:

-Game Over cu “Retry”, unde jocul reincepe de la ultima salvare, “Quit”, care ne trimite in meniul principal;

-Game Won cu “Restart”, care ne trimite in meniul principal si optiunea “Quit” care inchide jocul.

Regulile jocului

Odata ce incepe jocul, eroina trebuie sa acumuleze un anumit scor pentru a putea trece de la nivelul 1 la 2 si de la 2 la 3, iar pentru a termina cu succes nivelul 3 eroina trebuie sa il invinga pe Zarus ca sa poata obtine o cheie cu care va deschide usa dupa care se afla Mushu. Odata ce il atinge, dragonul este salvat si jocul castigat, dupa care se va afisa ecranul pentru victorie. Scorul se obtine atunci cand eroina:

-invinge un inamic(animale,spirite,Zarus);

-colecteaza un obiect care ii va modifica instant viteza de deplasare sau numarul de vieti(diamante,rubine,smaralde,sabii,mancare);

-colecteaza o cheie;

-deschide o usa.

Proiectarea nivelurilor

Rularea codului jocului deschide meniul principal

Exista 3 nivele si toate sunt generate in urma incarcarii a 3 matrici de dimensiuni 50x50 si valori(asociate cu anumite tile-uri) diferite incarcate din 3 fisiere de tip “.txt”. De la un nivel la altul, eroina isi pastreaza viteza si viata.

In primul nivel eroina este creata aproximativ in mijlocul hartii. Exista un mesaj in partea de sus a ecranului care indruma utilizatorul sa acumuleze un anumit scor pentru a putea avansa prin portile aflate in coltul din stanga-sus al hartii. Astfel, primul nivel tine de explorarea hartii ce are cateva franturi de labirint. Eroina trebuie sa gaseasca chei, sa descuie porti, sa invinga inamicii(maimute si gaini) si sa adune obiecte pentru a mari viata, viteza si a ajunge la pragul de scor dorit pentru a ajunge la nivelul 2.

(Locul in care este generat eroina in primul nivel)

(Portile prin care trebuie sa treaca odata ce atinge scorul cerut)

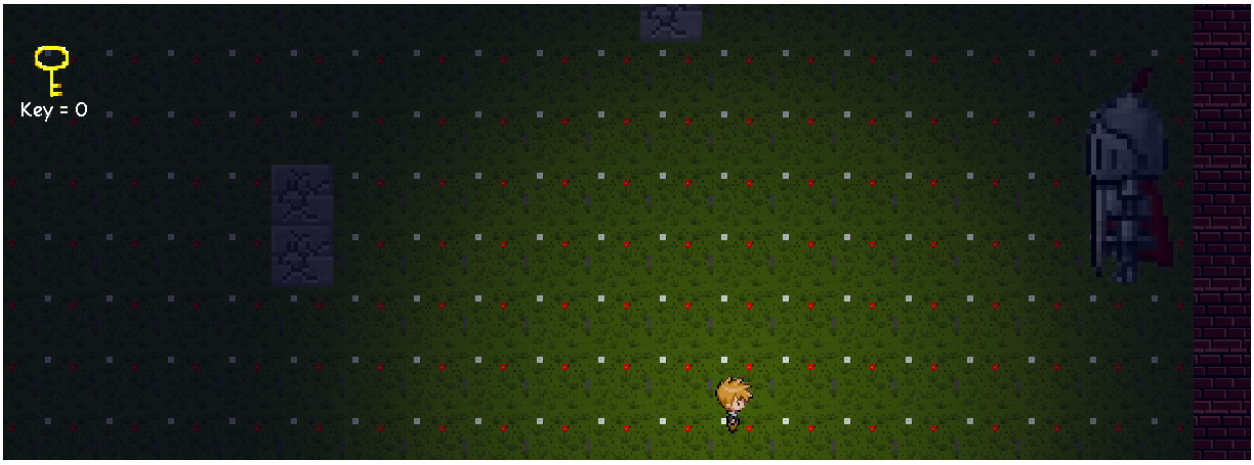
In al doilea nivel, eroina este de asemenea creata aproximativ in mijlocul hartii care de aceasta data este un intreg labirint. Astfel, exista un singur

traseu corect si harta este “presarata” cu camere (incuiate sau nu) ce au obiecte si inamici in interior. La fel ca la primul nivel, utilizatorul este indrumat de un mesaj cu scorul obligatoriu pentru a trece la nivelul 3.

(O parte din cel de-al doilea nivel)

In al treilea nivel, eroina este situata in coltul din stanga-sus al hartii si prima treime de harta pe care trebuie sa o strabata este de tip labirint, urmand sa intre intr-o incapere patrata ce ocupa restul hartii unde se afla Zarus.Spre deosebire de ceilalti inamici, Zarus are mult mai multa viata, este mai mare si mai rapid. Odata infrant, Zarus va lasa in urma o cheie cu care Guza va putea sa descuie incaperea in care se afla Mushu. Jocul este terminat in momentul in care Mushu este atins de Guza, dupa care se va afisa ecranul setat pentru castigarea jocului, pe care apare mesajul de felicitari si scorul acumulat, urmat de optiunile “Restart” si “Quit”.

Guriuc Vlad-Ionut

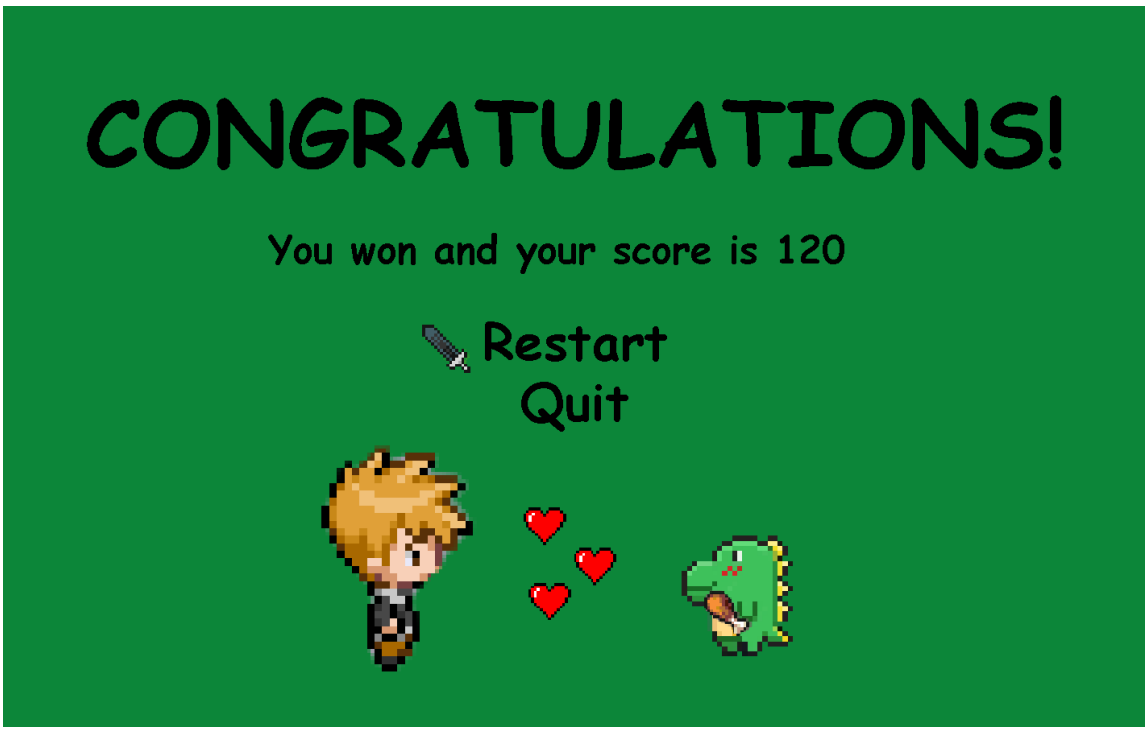


(Zarus in cel de-al treilea nivel)

(Incaperea in care se afla Mushu)

Guriuc Vlad-Ionut

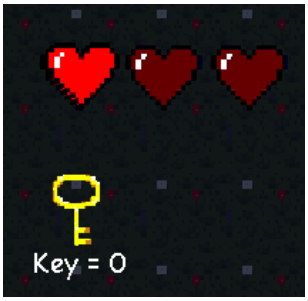
(Ecranul pentru joc pierdut)



(Ecranul pentru joc castigat)

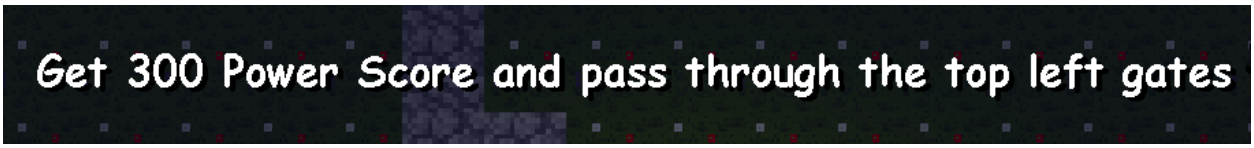
Proiectarea interfetei cu utilizatorul

Interfata cu utilizatorul este proiectata astfel incat sa fie clar delimitata clar de restul elementelor din joc, intr-o maniera simplista si sugestiva.



Nivelul de viata si numarul de chei colectate afisate in stanga-sus

Acestea se modifica de fiecare data cand eroina intra in contact cu un inamic sau un obiect, iar numarul cheilor se afiseaza in functie de valoarea aflata in baza de date. Eroina incepe cu 2 puncte de viata(o inima intreaga) si are o limita maxima stabilita de 10 puncte. Inamicii nu vor lasa dupa moarte de fiecare data un obiect ce ii va regenera viata eroinei, ceea ce poate ingreuna sau usura jocul in functie de sanse. Acest fapt face ca fiecare sesiune de joc sa fie unica in felul ei.



Mesaj pentru indrumarea utilizatorului la fiecare nivel afisat in partea de sus

Mesajul este afisat in permanenta pe parcursul nivelului si este diferit pentru fiecare nivel, avand din ce in ce mai putine detalii.

Mesaj nivel 1:”Get 250 score and pass through the top-left gates”.

Mesaj nivel 2:”Get 550 score to pass through the gates”.

Mesaj nivel 3:”Defeat Zarus and save Mushu!”.

Scor curent afisat in dreapta-sus in timpul jocului
Scorul se modifica atunci cand eroina interactioneaza cu orice obiect sau entitate pe care o invinge si este salvat si afisat din baza de date.

Proiectare continut

Toate caracterele, obiectele si tile-urile au la baza incarcarea unor imagini de tip “.png” fara fundal. Hartile sunt alcatuite din 3 fisiere “.txt” ce contin fiecare cate o matrice de dimensiuni 50x50, unde fiecare element reprezinta una din imaginile dedicate pentru tile-uri.

Obiectele sunt incarcate pe harta cu ajutorul unei clase numite AssetSetter. Fiecare obiect este creat cu ajutorul unei clase proprii si in AssetSetter le concepem, precizand tipul de obiect dorit, harta pe care sa fie generat si coordonatele dorite. Acestea ofera o serie de avantaje sau dezavantaje, spre exemplu:

 -mareste cu o unitate viteza eroinei;

 -mareste scorul cu 100, dar scade un nivel de viata si doua nivele din viteza eroinei;

 -mareste viata cu o unitate si creste scorul cu 5.

Entitatile au atribuite o serie de sprite-uri ce sunt incarcate succesiv in baza unor contoare si a directiei corespunzatoare(sus, jos, stanga, dreapta), in functie de actiunea savarsita(animatie de mers, animatie de moarte). Dupa ce sunt infrante, entitatile vor disparea de pe harta, mai putin in cazul eroinei. Fiecare entitate are un nivel de viata, vizibila in cazul unor inamici odata ce sunt atacati si o viteza de deplasare(ex. Green cu 20 de puncte de viata, dar cu 2 de viteza). Toti inamicii se deplaseaza pe harta in baza unei functii cu un contor ce se randomizeaza si schimba in functie de 2 sau 4 intervale directia de deplasare a entitatii(ex. Gaina se deplaseaza stanga-dreapta, Green in toate cele 4 directii). Odata ce sunt atacati, inamicii se vor indrepta spre directia din care au fost loviti, si cand vor efectua

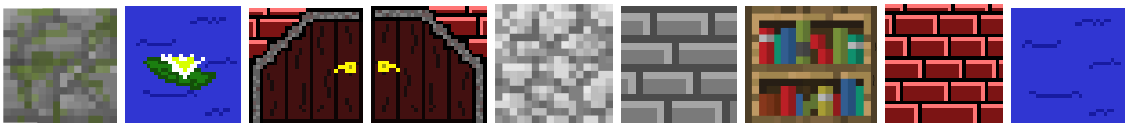
coliziunea cu eroina, ambii membrii vor avea cate un punct de viata scazut si vor intra intr-un stadiu de invincibilitate pentru o secunda in care nu vor putea muri. Daca inamicul, eroina primeste 10 puncte de scor si in baza unei functii de randomizare va avea 25%,50% sau 100% sa primeasca un obiect de la entitatea omorata(ex. Dupa ce un inamic de tip Ghost moare, exista 50% sanse sa lase in urma sa un smarald).

 -Ghost—10 puncte de viata, 3 de viteza, deplasare in cele 4 directii si 50% sanse sa lase un smarald dupa moarte.

Guza dispune de 2 puncte de viata, 3 de viteza si proiectile de tip flacara inca de la inceput.

Sprite-uri folosite

Tile-uri:



Obiecte:

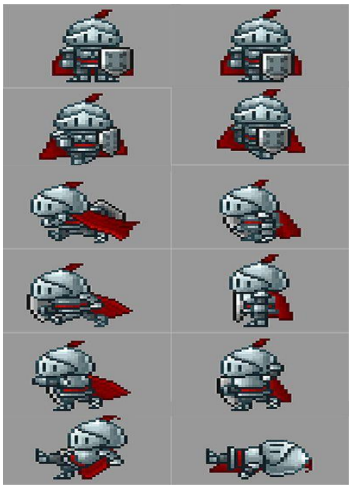
Guriuc Vlad-Ionut

Eroina Guza:

Inamici:

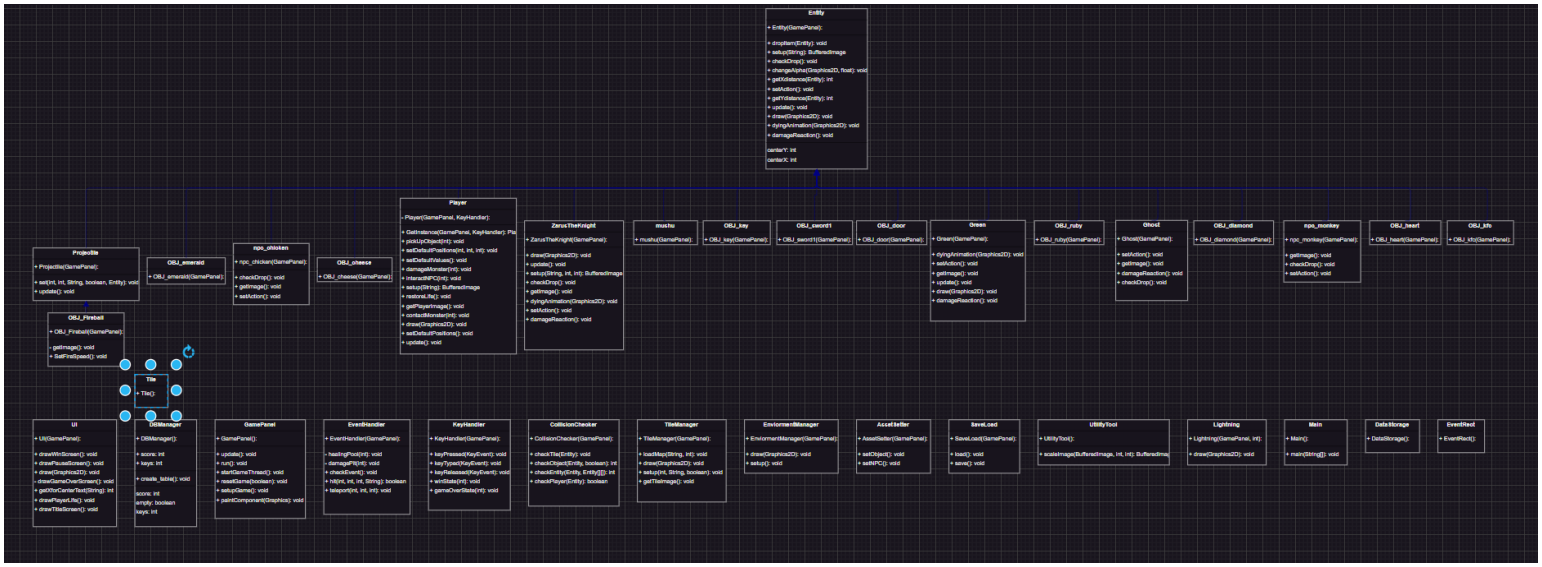
Mushu:

Zarus:



Guriuc Vlad-Ionut

Diagrama UML si explicarea claselor



Clasa Main

Sunt apelate functiile necesare pentru crearea ferestrei in care se va desfasura jocul, functia create_table() din clasa DBManager pentru baza de date.

Clasa GamePanel

In aceasta clasa este scalata rezolutia ferestrei de joc,declarate dimensiunea in pixeli a tile-urilor, dimensiunea hartilor(50x50), vectorul de entitati, de obiecte si lista de proiectile si singleton-ul pentru Player.

Functia setupGame() plaseaza pe harta entitatile, obiectele, atribuie gameState-ului in prima instanta conceptul de titleState la deschiderea jocului si apeleaza functia setup destinata pentru aparitia Fog Of War.

Functia resetGame(boolean restart) se apeleaza atunci cand utilizatorul pierde jocul si apasa Retry sau cand incepe o sesiune noua apasand “New Game” in titleState.

Functiile startGameThread(), run() si update() asigura continuitatea jocului prin executarea succesiva a task-urilor, thread-urilor si functiilor de update care implementeaza notiunea de timp, procesarea si incarcarea la momentul potrivit al sprite-urilor pentru entitati, obiecte si tile-uri.

Functia paintComponent modifica vectorii si listele de entitati si obiecte ori de cate ori este generata sau eliminata o entitate de pe harta pentru a fi desenata sau stearsa.

Clasa EventHandler

Folosindu-ne de clasa EventRect care implmenteaza notiunea de dreptunghi, putem genera o serie de evenimente ce se vor intampla la coordonatele (definite in EventRect si declarate in EventHandler) dreptunghiului dorit. Una din cele mai importante functii din aceasta clasa este cea de teleportare cu ajutorul careia eroina trece la urmatorul nivel in proximitatea portilor daca a acumulat suficiente puncte.

Clasa KeyHandler

In aceasta clasa regasim doar variabile de tip bool iar cu ajutorul functiilor KeyReleased si KeyPressed am implementat navigarea in meniuri sau in joc cu ajutorul tastelor. gameOverState si winState sunt ecranele ce apar la castigarea sau pierderea jocului si sunt implementate in aceasta clasa si apelate in functie de state-ul jocului.

Clasa UI

Aceasta este clasa in care am implemetat interfata cu utilizatorul si dispune de o serie de functii draw pentru desenarea ecranelor pentru winState, gameOverState, titleScreenState si playState. Primele trei mentionate reprezinta o serie de sprite-uri aflate si in joc, pe un fundal colorat, texte potrivite contextului si un indicator pentru partea interactiva ce arata optiunea curenta selectata, sub forma unui sprite de sabie.

In timpul jocului, utilizatorul dispune multumita functiilor drawPlayerLife() si draw(), de afisarea nivelului curent de puncte de viata, chei, si scor. De asemenea, utilizatorul este indrumat la fiecare nivel de un mesaj scurt, afisat in marginea superioara a ferestrei jocului.

Clasa CollisionChecker

Aceasta clasa contine toate functiile necesare jocului pentru a avea coliziuni intre toate elementele. Functiile sunt checkTile, checkPlayer, checkEntity, checkObject si implementeaza notiunea de coliziune cu ajutorul unor dreptunghiuri de diferite dimensiuni. Tile-urile(48x48 pixeli) au dreptunghiul pentru dimensiune la fel de mare, iar pentru celelalte elemente, dreptunghiurile lor sunt sincronizate in functie de aspectul fiecaruia. Cu alte

cuvinte, daca dreptunghiul unei entitati intalneste dreptunghiul unui tile ce are coliziune, viteza viteza entitatii va fi redusa la zero cat timp entitatea incearca sa se deplaseze in directia acelui tile.

Clasa AssetSetter

Aceasta clasa are scopul de a seta coordonatele tuturor obiectelor si inamicilor aflati pe oricare din cele 3 harti.

Clasa EnviromnentManager si Lightning

Aceste clase implementeaza notiunea de Fog Of War, iar cu ajutorul a doi vectori se creeaza efectul intunericului. Unul din vectori se foloseste pentru a crea cercuri din ce in ce mai mari, care au acelasi centru(coordonate in pozitia curenta a eroinei) si cu celalalt asignam nuanta din ce in ce mai intunecata dintre intervalele cercurilor.

Clasa DBManager

In aceasta regasim functia pentru crearea unui tabel in sql cu conexiunile necesare pentru JDBC si actualizarea valorilor in baza de date intocmind si apeland functiile getScore(), setScore(int score), getKey() si setKey(int key).

Clasele DataStorage si SaveLoad

In DataStorage declaram toate variabilele necesare pentru a intocmi un set de variabile temporare pe care le salvam sau incarcam in joc, folosind functiile save() si load(). In DataStorage folosim aceste elemente pentru a salva si incarca pozitia eroinei de la ultimul checkpoint, nivelul de viata, viteza si eliminarea sau adaugarea pe harta a unor obiecte in functie de circumstante. Functia save() e folosita de fiecare data in corpul functiei teleport() pentru realiza notiunea de checkpoint la fiecare nivel, astfel jucatorul va trebui sa rejoace numai nivelul. Pentru a rejuca nivelul, avem nevoie si de functia load() care este apelata atunci cand reincepem un nivel resetat sau cand accesam optiunea “Load Game” din meniu iar aceasta are rolul de a interschimba elementele temporar salvate cu cele curente.

Clasele Tile si TileManager

In Tile regasim variabile declarate folositoare deosebirii tile-urilor iar TileManager are rolul de a incarca hartile dorite si a desena toate tile-urile bazate pe corespondenta fiecaruia cu cate un numar predefinit in fiserele hartilor.

Clasa Entity

Este cea mai ampla clasa intrucat toate clasele obiectelor si ale personajelor o mostenesc. Aici sunt create functiile de desenare a caracterelor in functie de butoanele destinate miscarii apasate pe tastatura, regasim functia de setup(String imagePath) care aici are rolul de a incarca si scala fiecare imagine iar parametrul sau trebuie sa fie adresa pozei dintr-un fisier. In functia draw(Graphics2d) avem scris si algoritmul pentru implementarea barii de viata a inamicilor. Doua functii semnificative pentru Inamici sunt dyingAnimation(Graphics2d) si dropItem(Entity) (folosita cu ajutorul checkDrop()). Aceste 2 functii sunt implementate diferit pentru aproape fiecare inamic. In afara de functiile ce implica miscare, procesul este similar si la obiecte, iar in clasa player exista un switch(bazat pe denumirea obiectului) creat cu ajutorul caruia eroina primeste diferite atribute pozitive sau negative.

Clasa Player

In aceasta clasa sunt prezente toate functiile necesare pentru ca eroina sa fie creata, procesata, incarcata si gata de interactiunea cu nivelul prin functii precum setDefaultValues(), restoreLife(), draw(), pickUpObject() sau contactMonster().

Bibliografie

Thinking in Java-Bruce Eckel
Head First Java, 2nd Edition-Kathy Sierra & Bert Bates
Curs Practic de Java-Cristian Frasinaru
Clean Code-Robert C. Martin
Effective Java-Joshua Bloch
Java: A Beginner's Guide, Eighth Edition-Herbert Schildt
Pentru creare sprite-uri: https://www.piskelapp.com/