Chaos
Hușman Carla-Gabriela
1211B

Povestea jocului:

Lumea reală e un loc înspăimântător. Dacă nu sunteți convinși, întrebați-o asta pe Nyx, o fată obișnuită, a cărei viață s-a schimbat radical după ce Moonagon și-a trimis supușii să atace planeta noastră. Înainte ca familia ei să fie capturată, aceștia au apucat să-i spună secretul ei, că este singura svetocha rămasă și din cauza lor are loc această invazie. Noile vești au copleșit-o, dar aceasta va lupta cu toate forțele ei pentru a-și dezvolta abilitățile și cel mai important, să găsească portalul pentru a ajunge pe planeta Chaos.

Prezentare joc:

Jocul va fi de tipul Side-Scrolling. Personajul principal este Nyx, care va trebui să depășească obstacole și să se lupte cu selenienii în diferite orașe ale lumii în căutarea portalului potrivit spre planeta Chaos. Sunt zeci de portaluri, care o vor purta dintr-un oraș în altul, dar numai unul este cel potrivit. Aceasta va strânge resurse pentru a putea rezista drumului, dar va debloca și noi abilități.

Reguli joc:

Jocul implică deplasarea personajului printr-un oraș și învingerea inamicilor. Aceasta se poate mișca stânga-dreapta (pentru a putea avansa), sus (pentru se putea urca pe obstacole, sări peste ele, pentru a se feri de puterile aruncate de către inamici). Nyx este învinsă dacă va fi capturată/ucisă de către selenieni, acestea întâmplându-se doar dacă va fi lovită de către puterile inamicilor sau dacă îi va atinge. Odată cu acestea, jocul va reveni în starea inițială. Înfrângerea inamicilor va putea fi posibilă doar dacă Nyx va reuși să-i ocolească sau dacă îi va distruge cu puterile ei. Cu fiecare nivel îndeplinit aceasta va trece într-un alt oraș al lumii. Începând cu cel de-al doilea nivel, jucătorul va putea alege două abilități dintre cele deblocate.

Personajele jocului:

- **Nyx** este protagonista. Ea este o persoană obișnuită care într-o secundă a aflat cine este de fapt. Capacitatea ei inițială este nulă, dar cu învingerea mai multor inamici, ea se va autocunoaște.

- **Selenienii** sunt soldații trimiși de Moonagon pentru a cuceri planeta și pentru a o captura pe singura svetocha în viață. Sunt periculoși când sunt atinși, de aceea vor trebui să fie ocoliți. Aceștia sunt Klorofyll, Jägare și Krabba. Singurul dintre cei trei care are abilități, dar care este și cel mai periculos, este Jägare. El va arunca către Nyx cu foc.



Tabla de joc:

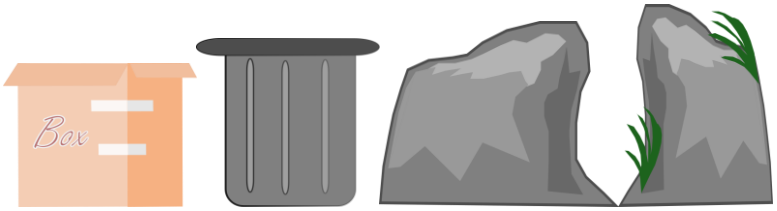
- Tabla

- Acțiunea celor două nivele se va desfășura de-a lungul a două nivele. Primul oraș este Seul, iar cel de-al doilea este Londra. Nyx va trebui să se miște de-a lungul imaginii, evitând obstacolele prezentate mai jos. Cele trei imagini au fost construite în inkscape.

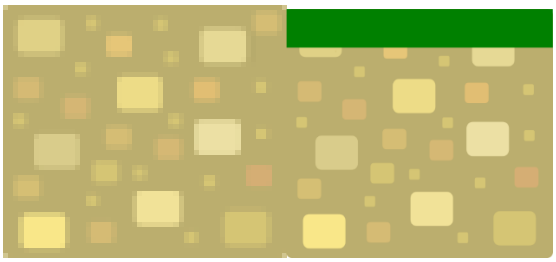
- Componente pasive

- Bomba - va provoca pierderea jocului la atingere acesteia;

- Cutiile, pietrele și coșurile de gunoi, vor fi obstacole peste care trebuie să sară sau pe care se poate urca. Sunt folositoare pentru a putea urca pe ziduri și a împiedica atacul inamicilor să o atingă.

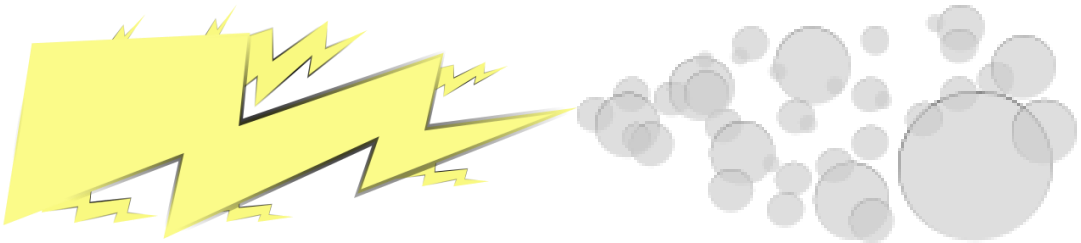


- Zidurile - Vor fi amplasate atât pe sol, cât și prin aer. Nyx se va putea urca pe ele ca să colecteze obiectele generatoare de puncte sau să evite inamicii.



- Componente active

- Abilitățile lui Nyx - vor fi aruncate către inamici la apăsarea unei taste și se vor deplasa până îi va atinge. Acestea pot fi aruncate oricând, dar nu vor distruge obstacolele, având efect doar asupra inamicilor. Blixt se numește prima ei abilitate. La activare, se vor elibera fulgere, care sunt dăunătoare pentru selenieni deoarece pe planeta lor, ei nu au astfel de fenomene. Cea de-a doua poartă numele de Bubblar și emite bule, care seamănă cu luna pentru a-i induce în eroare pe inamici.



- Abilitatea lui Jägare.

- Obiecte de colectat:

- fructe / bani / susținere - obiecte de care protagonista are nevoie pentru a ajunge la final sănătoasă.

- Blixt/Bubblar - abilitățile pe care care le va colecta pe parcursul nivelului 1.

- Panouri pentru a marca începutul și sfârșitul jocului. Ajungerea la portal va însemna câștigarea nivelului.

Mecanica jocului

-Navigarea se face cu UP și DOWN
-Selecțiile se fac ENTER și BACKSPACE

-Nyx se poate mișca în cadru astfel:
-Stânga - săgeată stânga;
-Dreapta - săgeată dreapta;
-Sus/Săritură - săgeată sus.

-Selenienii - se vor mișca stânga-dreapta.

-Eliberarea puterilor lui Nyx:
-tasta A - prima putere selectată;
-tasta S - a doua putere selectată.

-Puncte:
-fructe +100p;
-abilități +1000p;
-bani +150p;
-pierdere joc -500p;
-învingere inamici +500p.

-Interacțiuni :
-Nyx nu poate trece prin/pe lânga obstacole, ele trebuiesc evitate prin săritură;
-Jocul se va opri dacă ea va atinge bombele (determină rănirea ei) sau inamicii (determină capturarea) sau dacă va fi atinsă de puterea lui Jägare (determină rănirea ei).

Game sprite

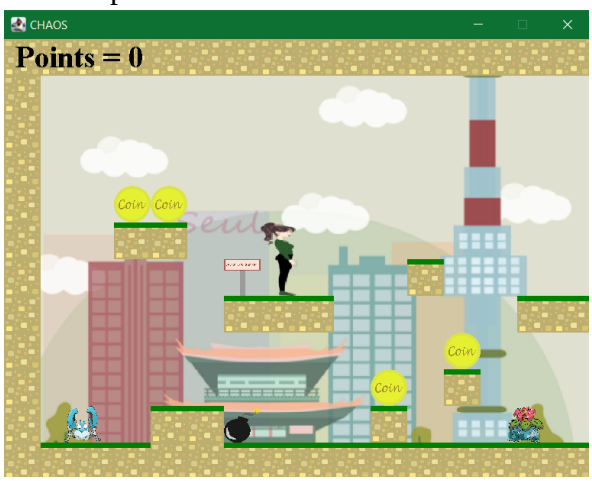
-Nyx are 3 stări - stat pe loc, alergare, săritură. Posibile atât spre dreapta cât și spre stânga.

Descrierea fiecărui nivel

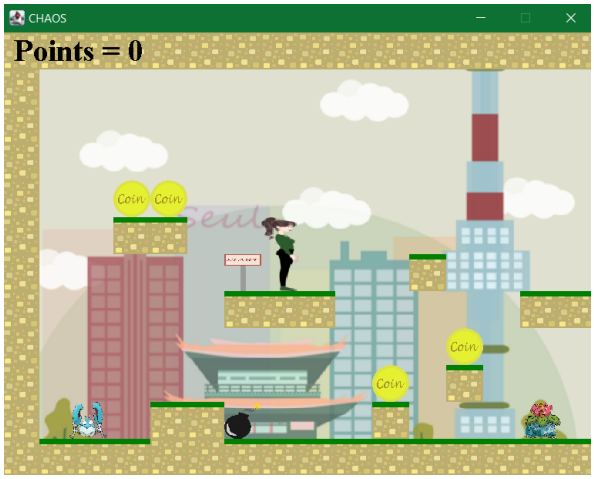
-Nivel 1

- Este un nivel de acomodare a jucătorului cu obiectele și cadrul. Acțiunea are loc în Seul. Pentru a avansa și pentru a ajunge la final, jucătorul trebuie să o deplaseze pe Nyx spre dreapta. Ea nu va avea puteri, astfel fiind nevoită doar să evite inamicii și bombele. Toate acestea au loc deoarece ea abia a aflat despre puterile ei și nu știe care sunt. Aceasta va colecta obiecte generatoare de puncte, dar și primele ei abilități. Sunt amplasate în aer, pe sol sau pe zidurile pe care se poate urca. Nivelul se va termina doar dacă va ajunge la portalul de la sfârșitul drumului. În caz contrar (atingere bombe sau inamici), ea va pierde și jocul se va întoarce în meniu, de unde poate reîncepe.

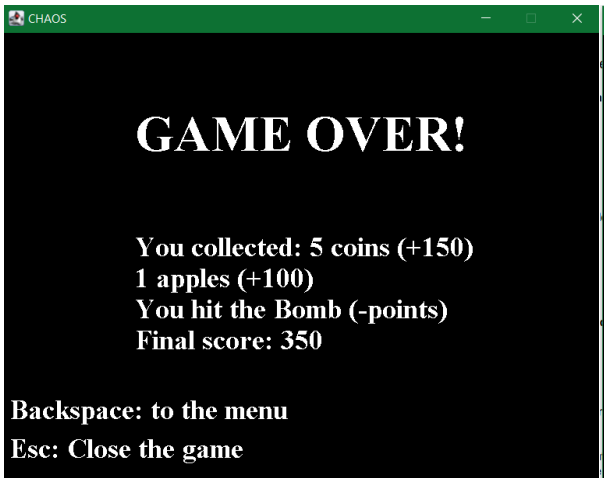
Ipostaze nivel:

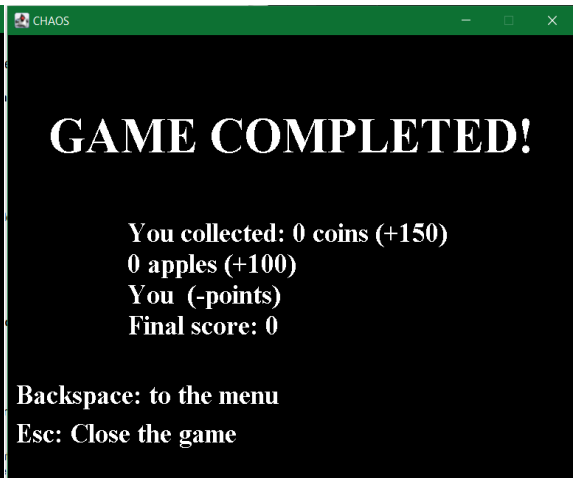


Începutul nivelului 1 fără a pierde înainte sau câștiga



Începutul nivelului 1 cu revenire prin apăsarea taste Backspace -> Meniu -> StartGame





-Nivel 2

-După pătrunderea în portal, aceasta ajunge în Londra, unde va putea să-și folosească puterile pentru a distruge inamicii mai ușor, nemaifiind nevoită doar să-i evite. Înainte de a începe, jucătorul va trebui să aleagă abilitățile lui Nyx. Jocul devine mai interesant deoarece vor exista mai multe interacțiuni între personaje. Acțiunile sunt aceleași ca la nivelul anterior, singura diferență fiind că vor apărea mai mulți inamici, obstacole noi, bombe mai multe, acestea crescând nivelul de dificultate deoarece vor fi mai greu de evitat. La final, dacă se va ajunge, se va întoarce în meniu.

Descriere meniu

Primul contact al playerului cu jocul este meniul unde jucătorul are mai multe opțiuni.

1. Start - începerea jocului;

Meniul la prima deschidere a jocului

Meniul după ce s-a scris prima oara în DataBase

2. Story - prezentarea poveștii din spatele jocului;

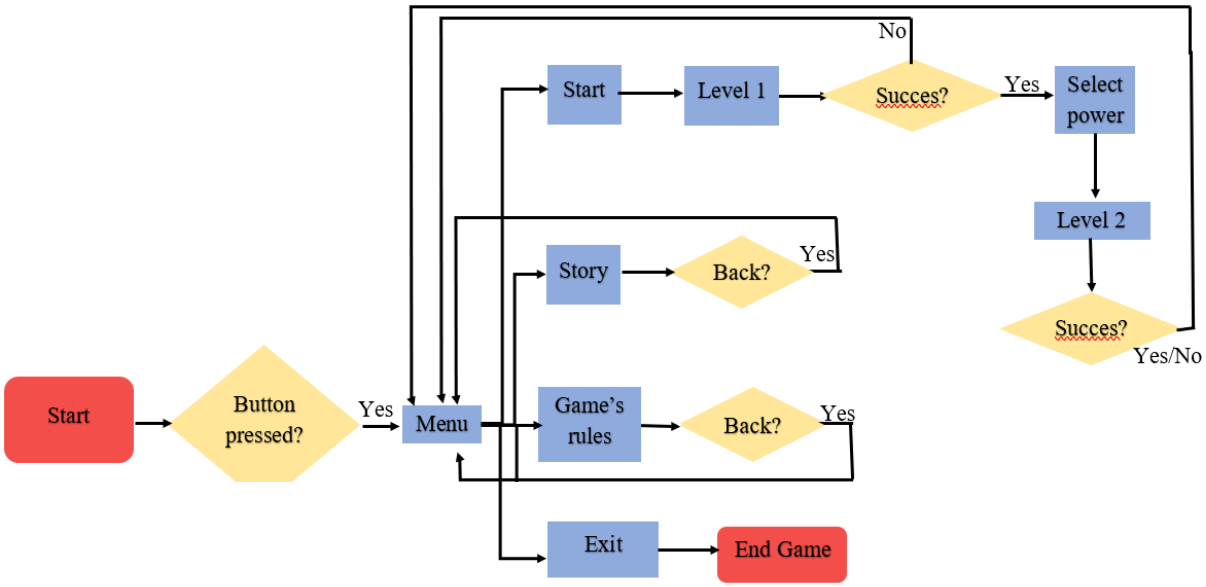
Apăsare backspace = întoarcere în meniu

3. Game's Rules - se va prezenta mecanica jocului (pe scurt);

Apăsare backspace = întoarcere în meniu

4. Exit - închiderea jocului.

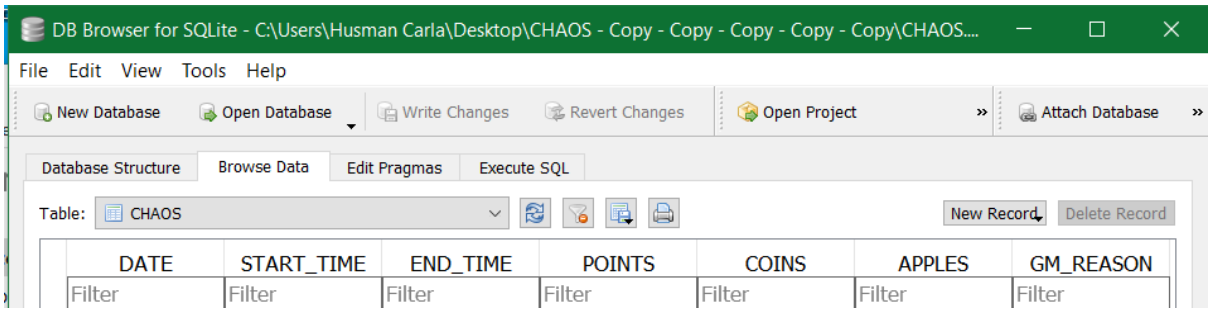
Diagrama meniului



Baza de date

Pașii parcuși:

1. Crearea unei baze de date din DB Browser for SQLite cu numele “Chaos” și tabelul următor:



DATE = data la care s-a conectat jucătorul
START_TIME = ora la care a început jocul (nivelul)
END_TIME = ora la care a terminat jocul (a pierdut)
POINTS = punctele acumulate pe parcursul jocului prin colectarea obiectelor
COINS = numărul de bănuți colectați
APPLES = numărul de mere colectate
GM_REASON = motivul pentru care s-a terminat perioada START_TIME – END_TIME (dacă a lovit bomba, dacă a lovit inamicii sau dacă a câștigat)

2. Am creat clasa DataBase în care am creat4 metode, specifice lucrulului cu baza de date:

```
public void openDataBase() {
    try {
        //conectare la baza de date
        Class.forName("org.sqlite.JDBC");
        c = DriverManager.getConnection( url: "jdbc:sqlite:CHAOS.db");
        c.setAutoCommit(true);
        stmt = c.createStatement();
    } catch (Exception e) {
        System.out.println("Nu s-a deschid baza de date.");
        System.err.println(e.getClass().getName() + ": " + e.getMessage());
        System.exit( status: 0);
    }
}
```

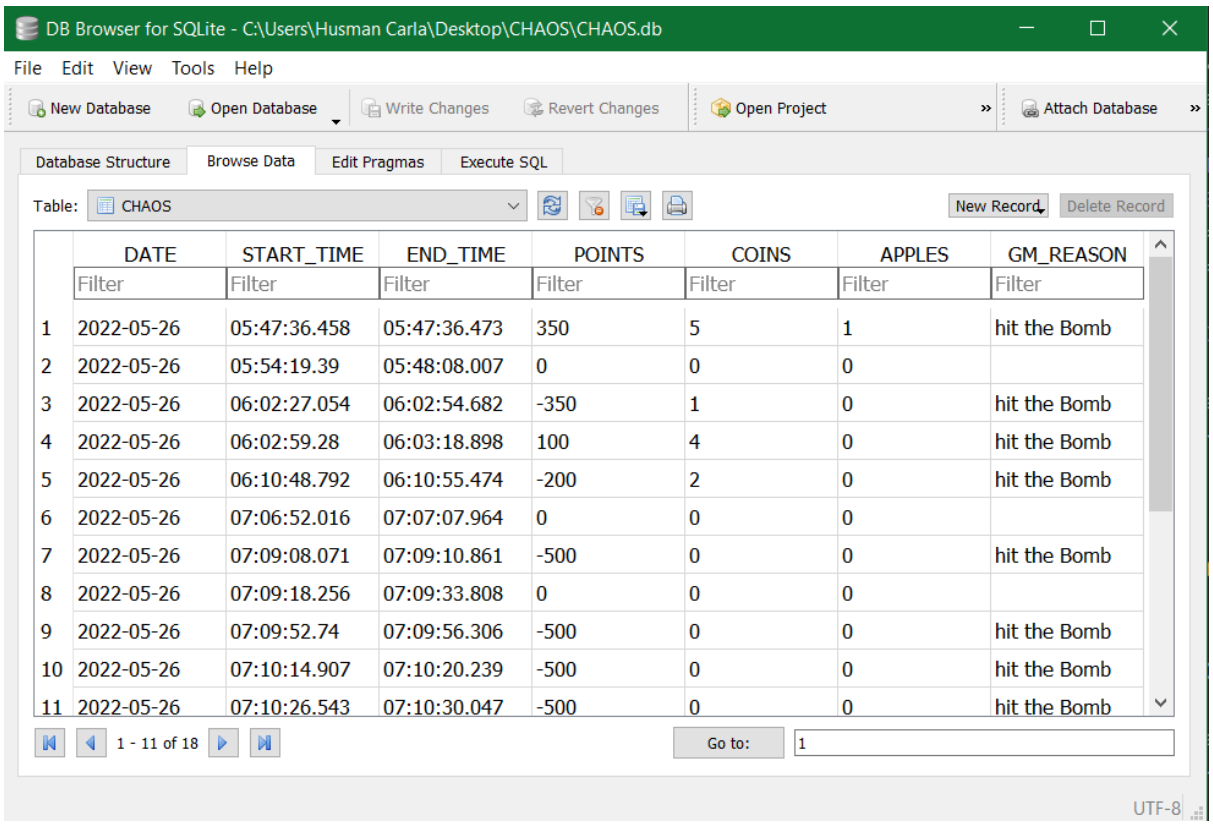
```
public void writeInDataBase() {
    try {
        sql = "INSERT INTO CHAOS (DATE,START_TIME,END_TIME,POINTS,COINS,APPLES,GM_REASON) " +
                "VALUES ('" + (DateTimeFormatter.ISO_LOCAL_DATE).format(time1) +
                "','" + (DateTimeFormatter.ISO_LOCAL_TIME).format(time1) + "','"
                + (DateTimeFormatter.ISO_LOCAL_TIME).format(time2) + "',"
                + gp.player.points + ","
                + gp.player.coin + ","
                + gp.player.apple + ",'"
                + gp.player.reason + "');";
        stmt.executeUpdate(sql);

    } catch (Exception e) {
        System.out.println("Nu s-a putut adauga o informatie in baza de date.");
        System.err.println(e.getClass().getName() + ": " + e.getMessage());
    }
}
```

```
public void writeFromDataBase() {
    try {
        rs = stmt.executeQuery( sql: "SELECT * FROM CHAOS;");
        while (rs.next()) {
            DATE = rs.getString( columnLabel: "DATE");
            START_TIME = rs.getString( columnLabel: "START_TIME");
            END_TIME = rs.getString( columnLabel: "END_TIME");
            POINTS = rs.getInt( columnLabel: "POINTS");
            APPLES = rs.getInt( columnLabel: "APPLES");
            COINS = rs.getInt( columnLabel: "COINS");
            GM_REASON = rs.getString( columnLabel: "GM_REASON");
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

```java
public void closeDataBase() {
    try {
        rs.close();
        stmt.close();
        c.close();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

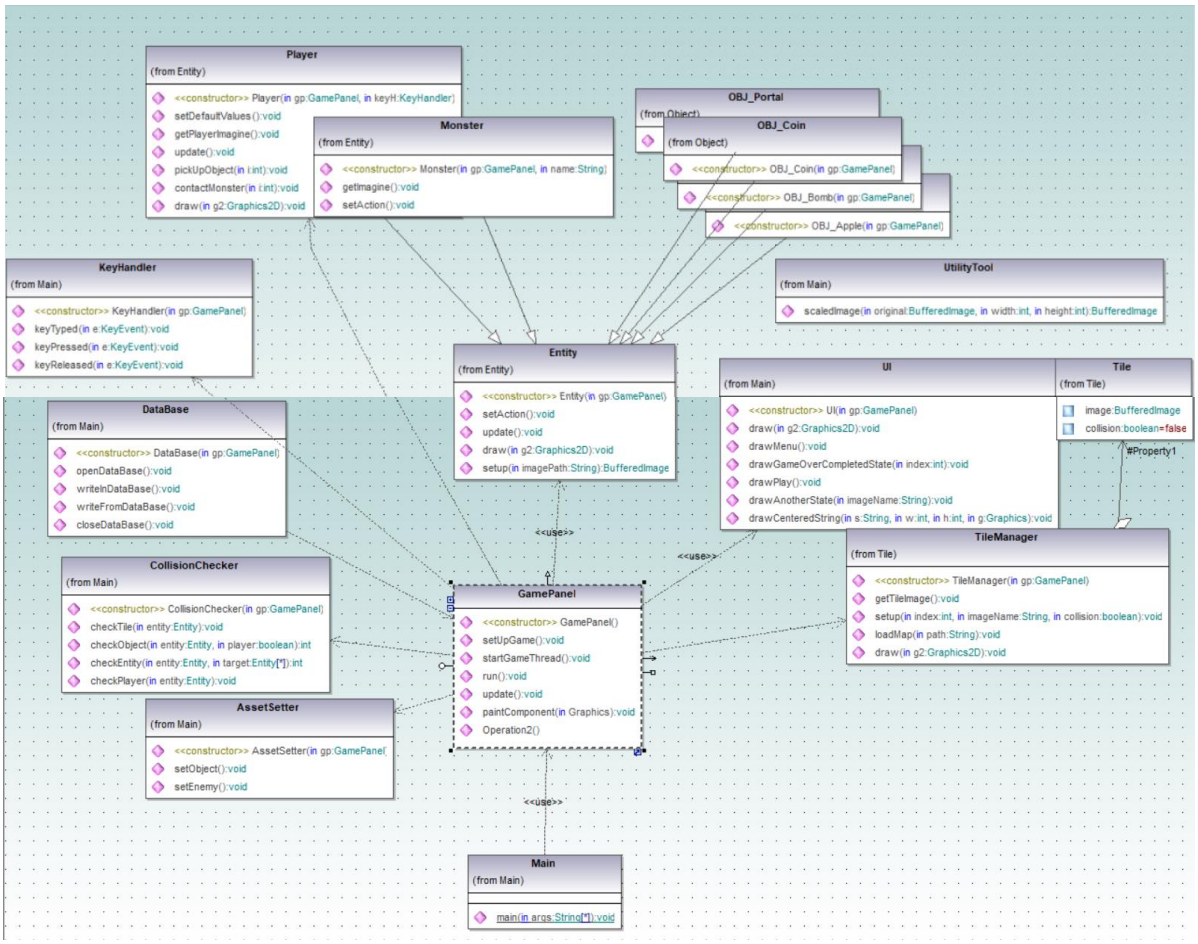
3. Am apelat respectivele funcții pe parcursul jocului pentru a putea efectua operațiile dorite. După încheierea programului am verificat baza de date creată



	DATE	START_TIME	END_TIME	POINTS	COINS	APPLES	GM_REASON
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	2022-05-26	05:47:36.458	05:47:36.473	350	5	1	hit the Bomb
2	2022-05-26	05:54:19.39	05:48:08.007	0	0	0	
3	2022-05-26	06:02:27.054	06:02:54.682	-350	1	0	hit the Bomb
4	2022-05-26	06:02:59.28	06:03:18.898	100	4	0	hit the Bomb
5	2022-05-26	06:10:48.792	06:10:55.474	-200	2	0	hit the Bomb
6	2022-05-26	07:06:52.016	07:07:07.964	0	0	0	
7	2022-05-26	07:09:08.071	07:09:10.861	-500	0	0	hit the Bomb
8	2022-05-26	07:09:18.256	07:09:33.808	0	0	0	
9	2022-05-26	07:09:52.74	07:09:56.306	-500	0	0	hit the Bomb
10	2022-05-26	07:10:14.907	07:10:20.239	-500	0	0	hit the Bomb
11	2022-05-26	07:10:26.543	07:10:30.047	-500	0	0	hit the Bomb

1 - 11 of 18

Diagramă proiect



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 metodelor *setupGame()* și *startGameThread()* din GamePanel.

GamePanel

Rolul acestei clase este de a face inițializările (de exemplu de a crea instanțe pentru jucători/inamici, pentru a inițializa harta, pentru setarea gameStatetului inițial, pentru dimensiunile tileurilor, pentru a fixa dimensiunile ecranului etc.), dar și de a menține interfața grafică la zi cu ceea ce se întâmplă în joc. Lucrează ca un ecran pentru jocul nostru. Această clasă implementează interfața Runnable pentru a avea comportamentul unui fir de execuție (thread) și extinde clasa Jpanel. Totodată în această clasă se instanțiază toate celelalte clase și se lucrează direct de aici, se vor apela funcțiile specifice respectivelor instanțe.

În momentul apelului metodei *StartGameThread()* se instanțiază un obiect de tip Thread pe baza instanței curente a clasei GamePanel. 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, 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 (*paintComponent(Graphics g)* care are rol de pensulă).

Metoda *update()* actualizează starea jocului (de exemplu: modifica poziția jucătorilor pe baza tastelor apăsate, schimbă poziția inamicilor, deseneaza diferitele backgrounduri pentru gameStateuri). Metoda *paintComponent(Graphics g)* va desena pe interfața grafică modificările făcute de metoda update().

Bucla metodei run() din clasa GamePanel

```
// 1 UPDATE
update();
// 2 DRAW cu noua informatie
repaint();//call paintComponent
/*Game loop
<- - -  - - - - -
|               |
update          |
|               |  60 FPS
repaint         |
|               |
- - - - - - - - ->
 */
```

KeyHandler

Implementează interfața KeyListener pentru a putea lucra cu tastatura. În această clase se vor petrece toate evenimentele ce țin de apasarea tastelor (UP, DOWN, LEFT, RIGHT, ENTER, BACKSPACE) și actualizeaza gameStateurile din GamePanel (are un membru de tip GamePanel), dar permite și navigarea prin meniu, selectarea opțiunii dorite, întoarcerea, etc. Toate acestea fiind realizate de metoda *KeyPressed(KeyEvent e)*.

UtilityTool

Această clasă are o singură metodă *BufferedImage scaledImage(BufferedImage original, int width, int height)* în care se va returna noua imagine cu noile dimensiuni.

Tile

Această clasă reține informații despre tile-urile din joc. Conține un membru de tip BufferedImage în care se va stoca un tile, dar și un flag pentru coliziune deoarece nu toate tileurile necesită să oprească personajul.

TileManager

În clasa GamePanel există o instanță a clasei TileManager. Conține un vector de Tile, în care se va stoca fiecare tile, cât și o matrice de numere întregi în care se va încărca mapa

nivelului, fiecărui index corespunzându-i un index din Tile[]. În metoda *getTileImage()* se va încărca fiecare imagine prin apelarea metodei *setup(int index, String imageName, boolean collision)*. A doua metodă a fost implementată cu scopul de a simplifica încărcarea imaginilor și a îmbunătăți lizibilitatea codului. Aceasta clasă conține o instanță a clasei UtilityTool.

Metoda *loadMap()* încarcă mapa dintr-un fișier text, iar *draw(Graphics g2)* desenează fiecare tile în funcție de poziția personajului, camera mișcându-se odată cu el. S-a ales această implementare pentru a crește performața codului.

Entity

Este clasa părinte pentru toate entitățile și conține informații care vor fi folosite la player, enemies, OBJ, cum ar fi: viteza, dreptiunghiul pentru coliziune, coliziunea, directia, imaginile pentru Sprites, coordonate, etc. Conține o clasă *setAction()* care nu face nimic, dar este implementată în clasele derivate, deci suprascrisă, așa că apelarea acestei metode prin Entity, va apela de fapt metoda corespunzătoare clasei care o implementează.

Metoda draw desenează pe ecran, iar update verifică coliziunile și actualizează coordonatele în funcție de deplasarea entityului (ex: Enemy)

Player

Extinde clasa Entity. În *update()* se va verifica dacă tasta apasată este UP, DOWN, LEFT, RIGHT, atunci direcția se va actualiza și se vor verifica coliziunile cu tiles, cu obiectele și cu inamicii. Dacă se atinge de obiecte, atunci acestea vor dispărea de pe ecran și punctele vor fi incrementate, iar dacă atinge un inamic, jocul se va termina. Prin apelarea funcțiilor respective din CollisionChecker, se va returna un index, index care reprezinta

Monster

Entinde clasa Entity și conține constructorul în care se pasează un GamePanel, se setează coliziunea, numele fisierului, mărimea dreptunghiului pentru coliziune și se încarcă respectivul obiect Aceștia vor fi adăugați prin intermediului lui *AssetSetter.setEnemy()*. Metoda *setAction()* îl mișcă pe ecran.

OBJ_Apple / Bomb / Coin / Portal

Toate clasele extind clasa Entity și conțin doar constructorul în care se pasează un GamePanel, se setează coliziunea, numele fisierului, mărimea dreptunghiului pentru coliziune și se încarcă respectivul obiect. Acestea vor fi adăugate prin intermediului lui *AssetSetter.setObject()*.

CollisionChecker

Este instanțiată în GamePanel și conține un membru GamePanel. În această clasă se vor detecta coliziunile dintre personaj și tiles, personaj și obiect, personaj și inamic, inamic și inamic, inamic și tiles. Metoda abordată este cea a încadrării într-un dreptunghi.

AssetSetter

Această clasă încarcă fiecare obiect și inamic prin cele două metode în obj[] (membrii de Entity ai clasei GamePanel).

UI

Desenează informațiile necesare jocului când se află în starea de meniu (și stările lui), gameOver, gameFinished. Aici se afișează pe ecranul jocului ultima conectare a jocului și se încarcă informațiile în data de baze.

Șabloane de proiectare

Singleton

Bibliografie:

https://inkscape.org/ (accesat la 04.02.2022)

https://www.vecteezy.com/vector-art/273697-flat-modern-london-s-landmarks-vector-illustration (accesat la 05.02.2022)

https://mixkit.co/free-sound-effects/game/?page=2 (accesat la 06.02.2022)

https://www.freepik.com/free-vector/flat-geometric-background_13859520.htm?query=color%20block (accesat la 06.02.2022)

https://www.deviantart.com/andersonaas107/art/003-venusaur-PNG-5-759452001(accesat la 17.02.2022)

https://www.deviantart.com/andersonaas107/art/003-venusaur-PNG-2-759450039 (accesat la 17.02.2022)

https://www.deviantart.com/pokevectors/art/HD-Gengar-06-Excited-768165804 (accesat la 17.02.2022) https://www.deviantart.com/pokevectors/art/HD-Gengar-02-Side-768166236 (accesat la 17.02.2022)

https://www.deviantart.com/lekadema/art/fire-vector-287636879 (accesat la 17.02.2022)

https://www.deviantart.com/lahirien/art/Moon-Splash-s-Cutie-Mark-Request-5091767 17 (accesat la 17.02.2022)

https://www.deviantart.com/conceptshinies/art/Shiny-Krabby-836626300 (accesat la 17.02.2022)