PAOO - L02

Anexa 1

**Titlu proiect :Salvarea Wessexului**

**Autor :CIOBANU EDUARD-TARCIZIU**

**Grupa:1208A**

1 punct oficiu pentru aspect, respectare cerinte

**Povestea jocului:**(1 pct): Jocul se petrece in Wessex fosta regiune a Angliei situata la sud vestul tarii.Erau vremuri de pace pana cand o armata de mercenari pusi de regele regiunii rivale Mercia,pe numele lui Mark I, ordona rapirea fiicei regelui Edward,rege de Wessex.Pe cand fiica acestuia,pe numele ei Ecaterina,se indrepta spre locul ei de recreere de langa cetate,insotita de 5 ostasi,mercenarii i-au atacat si au rapit-o pe Ecaterina. Intentiile regelui Mark I,erau de a-l santaja pe regele Edward cu moartea fiicei sale,astfel acesta din urma sa renunte la regine si sa il proclameze pe Mark rege al Wessexului si al Merciei.Ajuta-l pe regele Edward sa cucereasca cetatile regiunii Mercia si sa ii salveze viata fiicei sale.

**Prezentare joc**:(1 pct):La inceputul jocului vor fi mai multe cetati apartinand, in functie de numar, regelui Edward si regelui Mark I.Cetatile vor avea fiecare cate un numar diferit de ostasi.Ele vor avea si drumuri intre care vor reprezenta cai de acces spre lupta.Pentru a ajunge la cetate regelui Mark este nevoie sa cuceresti toate cetatile acestuia si astfel sa o salvezi pe fiica regelui Edward.

**Reguli:** La inceputul jocului ficare cetate va avea un numar de ostasi stabilit.Pentru a trimite ostasi spre un castel trebuie sa dai click pe castelul respectiv iar un numar predefinit de ostasi vor pleca.Daca numarul de ostasi din cetatea pe care doresti sa o cuceresti este mai mic decat numarul de ostasi trimisi de tine atunci ostasii tai vor cuceri cetatea aceasta devenid cetatea ta.Pentru a mari numarul de ostasi din cetatea de baza va trebui sa tastezi click pe ceatea de baza astfel numarul de ostasi va

creste.La fiecare secunda numarul ostasilor din cetatile rivale va creste cu 2 unitati sau cu mai mult in functie de dificultatea nivelului.La un interval de timp automat un numar de ostasi va fi trimis spre cetatile tale.Daca numarul acestora nu este suficient de mare astfel incat sa cuceresaca cetatea atunci vor muri iar numarul de ostasi din cetatea ta va scadea in functie de numarul de ostasi trimisi,in cazul in care numarul ostasilor este mai mare decat ostasii tai cetatea respectiva va fi cucerita.Pentru ca termina nivelul este nevoie sa cuceresti toate castelele rivale.

**Personajele jocului**:(1 pct):

**-Robin-Hood** este protagonistul și jucătorul-personaj.Este comandantul garzii regelui Edward,este cel mai priceput arcas al regiunii de Wessex si este de neoprit.De la un colt la altul al Wessexului este cunoscut pt indemanarea sa si curajul de care a dat dovada pana acum

C:\Users\Eduard\Desktop\JOC_PAOO\ScheletProiectPAOO\res\textures\playerback1.png

**-Mercenarul** platit sa o rapeasca pe fiica regelui si sa apere regiune Merciei.Nu are suflet si nu cunoaste mila acesta fiind antrenat sa ucida si sa fie platit foarte bine pentru asta.Este de asemenea de stiut ca si acesta este un razboinic indemantic si cu sange rece

C:\Users\Eduard\Desktop\JOC_PAOO\ScheletProiectPAOO\res\textures\inamicback1.png

**Game sprite** (1 pct):



**Meniu:** ****

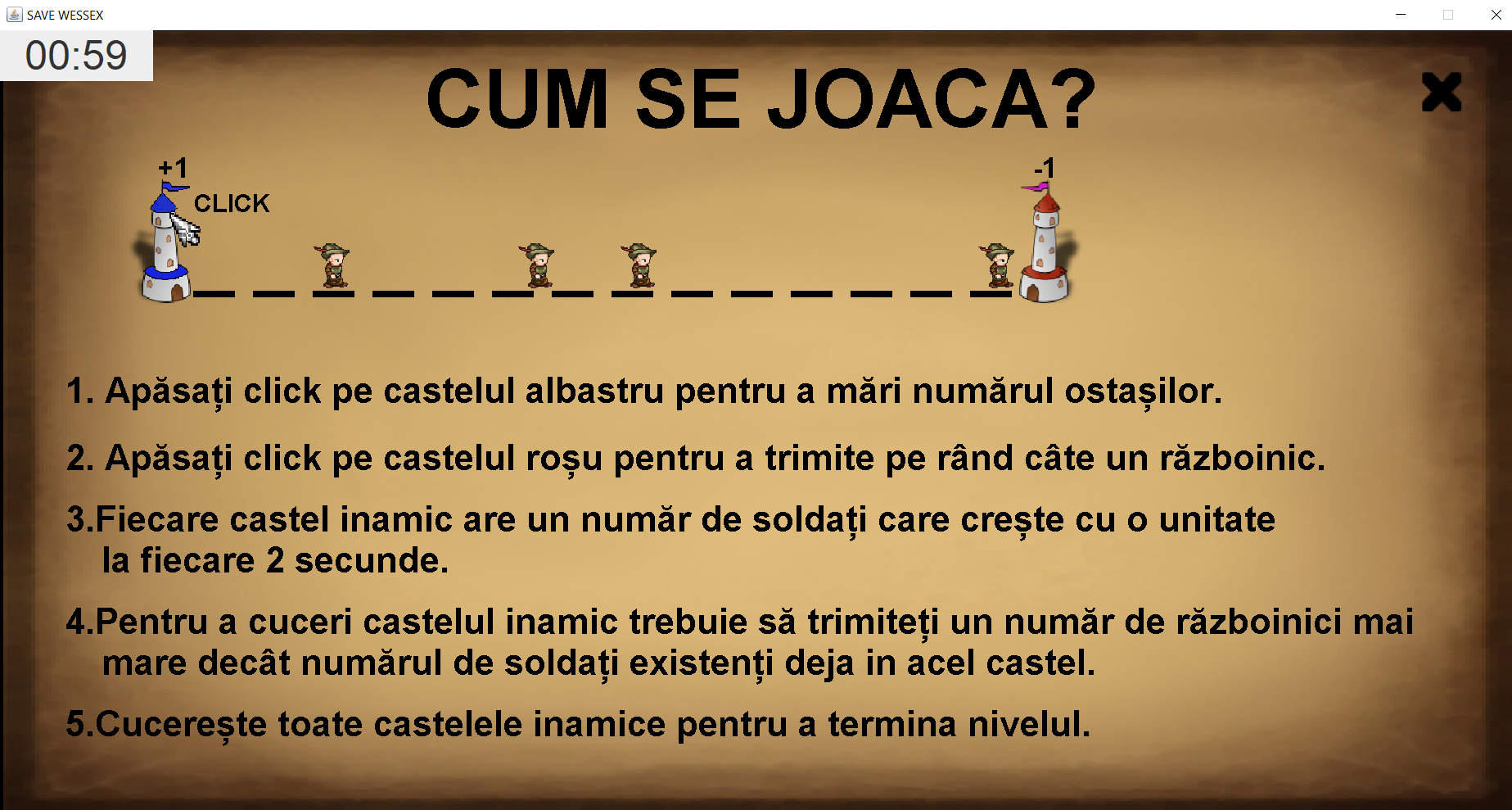
Harta nivel:

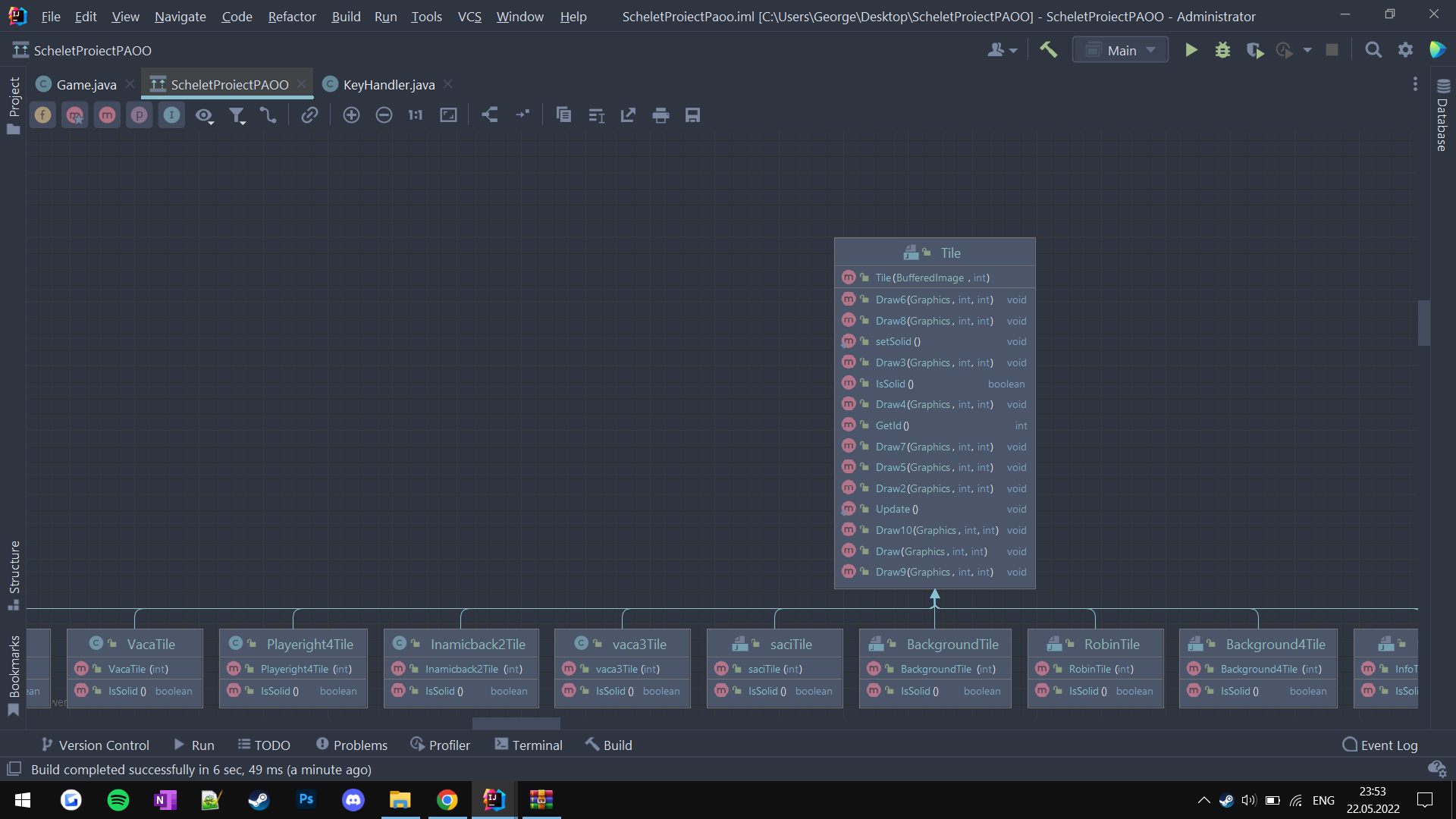
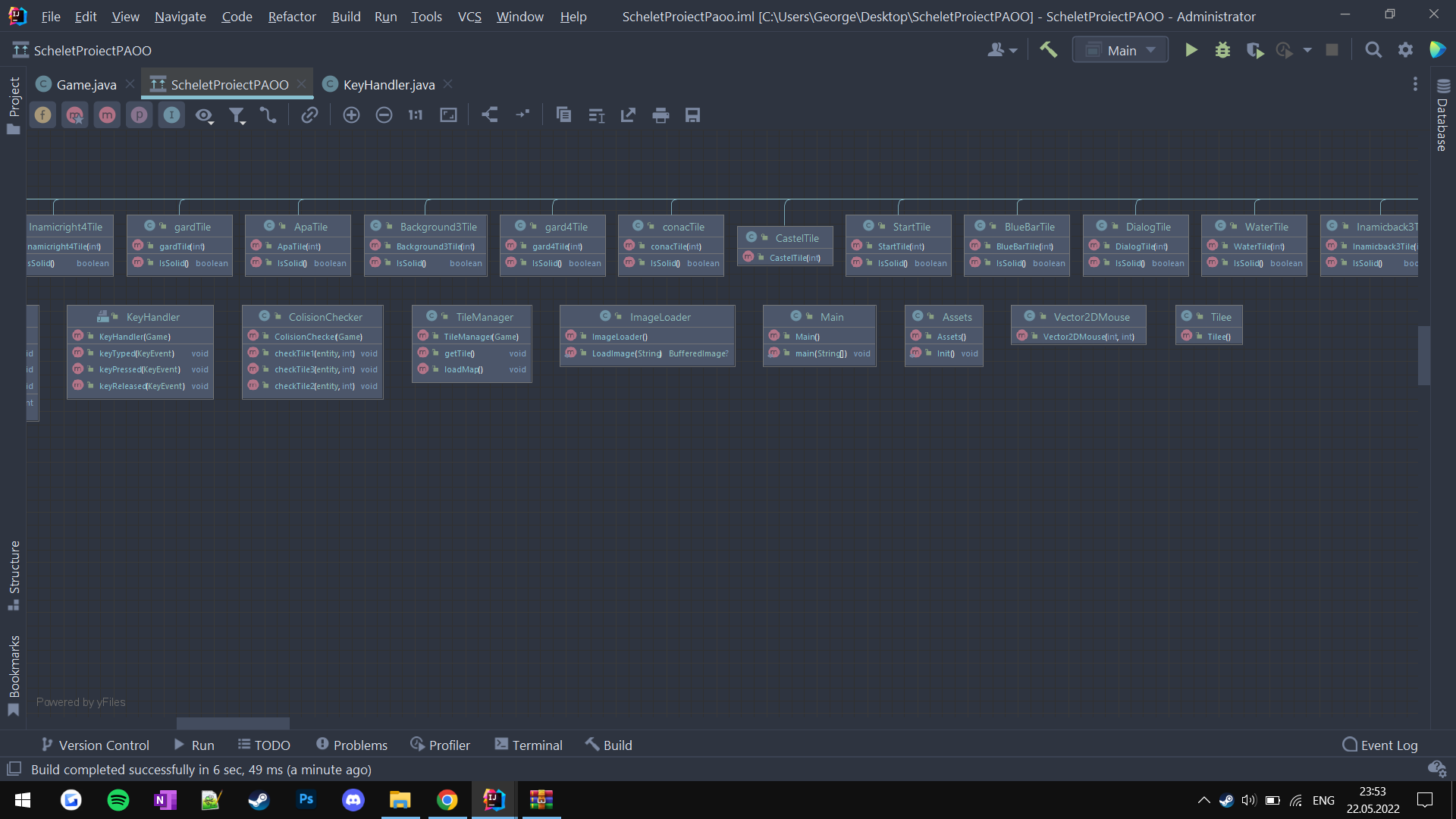
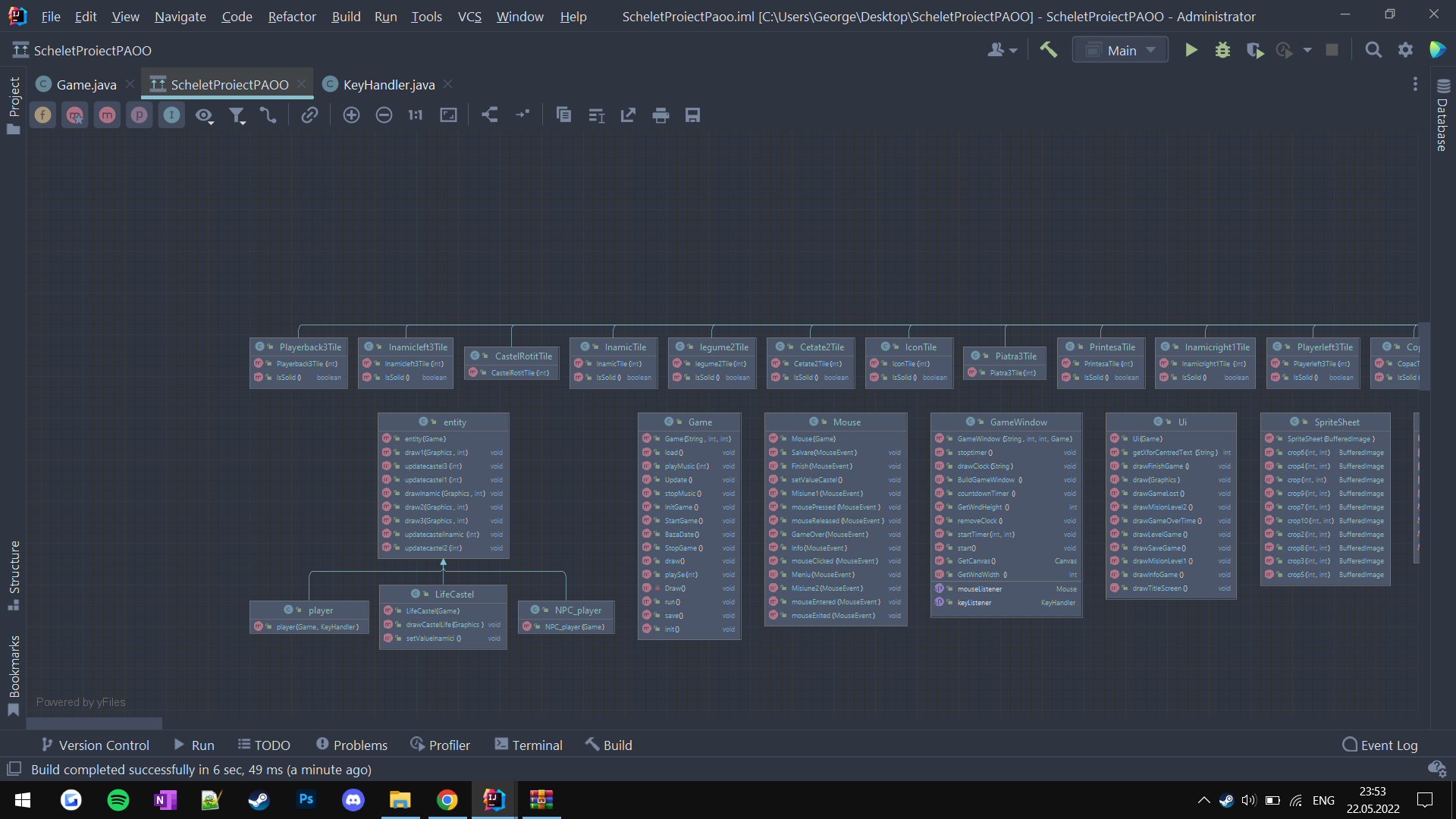
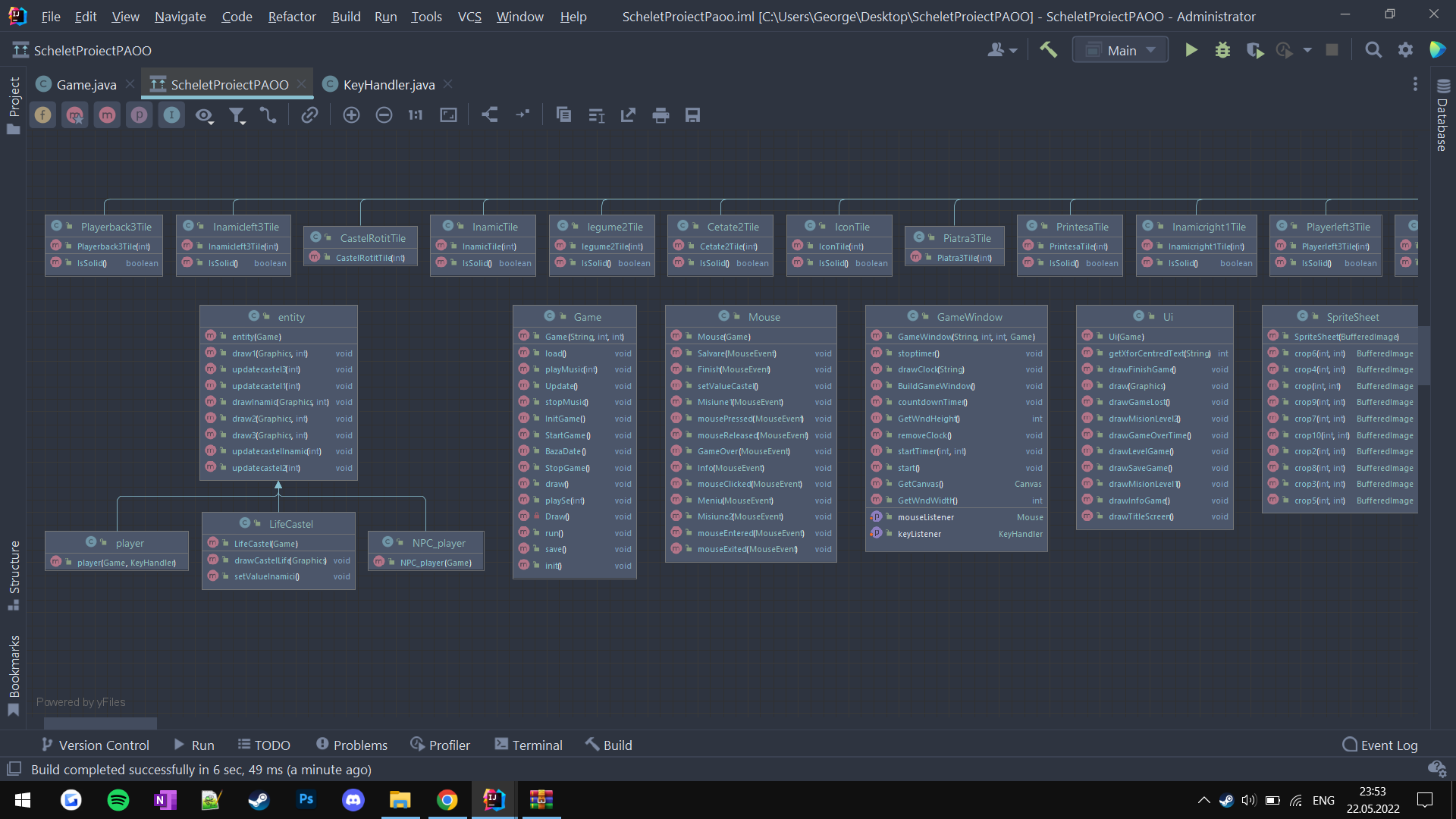


Tabela de joc:



Cum se joaca?





**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()).

**Metoda Update()** actualizează starea jocului (de exemplu: modifica poziția jucătorilor pe baza tastelor apăsate, schimbă poziția inamicilor folosind chiar tehnici de inteligență artificială, crează diferite tile-uri (dale), etc. Dacă ne referim la un joc 2D care are harta pătrată, aceasta este împărțită în celule de aceeași dimensiune(ca o tablă de sha), 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 și restul elementelor de pe hartă dar care are fundalul transparent).

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

**GameWindow**

După cum discutam mai sus, metoda run() a clasei Game crează o instanță a clasei GameWindow. Această clasă este responsabilă cu fereastra 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 conține și 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 și unul pentru acțiunea jocului, iar în funcție de interacțiunea utilizatorului acestea vor fi schimbate între ele.

In aceasta clasa am adaugat functia de startTimer() care implementeaza timpul jocului,intervalul de timp permis pentru fiecare nivel.

**Assets**

Conține câte un membru static de tip BufferedImage pentru fiecare tile/dală, chiar și pentru jucători/inamici. In acești 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. În acest exemplu este doar o imagine care conține toate texturile folosite, așadar pentru a putea instanța acest obiecte statice trebuie să decupăm fiecare textura din acea imagine. Pentru a face asta se folosește o instanță a clasei SpriteSheet în metoda Init(). Metoda folosită din SpriteSheet este crop(x, y).

**SpriteSheet**

Constructorul acestei clase primește imaginea, iar clasa conține 2 membri constanți (final in Java) pentru înălțimea, respectiv lățimea texturilor (trebuie sa aibă toate aceleași 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 poziția în imagine se determină pixelii de unde încep texturile de interes prin înmulțirea liniei/coloanei cu înălțimea, respectiv lățimea texturilor. Astfel se obține colțul din stânga sus al texturii, iar pentru a afla celelalte colțuri se folosesc înălțimea, respectiv lățimea.

**ImageLoader**

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

**Entity**

In aceasta clasa am initializat pozitiile jucatorilor si inamicilor,vitezele lor si directiile.Am implementat functii de update() si de draw(),care se apeleaza in functiile Draw() si respectiv Update() din clasa GAME(),pentru a modifica pozitiile jucatorilor si directiile lor.

**LifeCastel()**

Aici am implementat functiile de desenare ale castelelor si a numarul soldatilor care se actualizeaza ori in functie de numarul click-urilor ori in functie de cati soldati ajung la castelele inamice.

De asemenea am implementat mesajul de inceput al Printesei blocata in cetate,respectiv deblocarea cetatii si eliberarea printesei.

**Npc\_player()**

De aceasta clasa ma folosesc pentru a crea obiecte de tipul **entity()-**aceasta clasa extinde clasa **entity()**

**assetSetter()**

In aceasta clasa instantiez obiecte de diferite tipuri,le initializez cu pozitii, directii si viteza.

**ColisionChecker()**

In aceasta clasa verific coliziunile cu ajutorul unei matrici

**UI()**

In aceasta clasa implementez functii de desenare al gameState-ului in functie de joc:

**-**desenare meniu joc

**-**desenare mapa level

**-**desenare misiuni level

**-**desenare meniu de GameOver