Titlu proiect (Super Java Battle)
Autor (Spătaru Alexandru)
Grupa (1209A)

Povestea jocului:

Jocul este situat pe insula misterioasă și îndepărtată sub numele de Java, locul natal al personajului nostru principal, SuperJava. SuperJava a lucrat și s-a străduit din răsputeri pentru a se dezvolta cat de bine a putut pentru a deveni limbajul de programare preferat al omenirii. SuperJava este o prințesă care este într-o luptă continuă pentru a deveni stăpânul limbajelor de programare. Însă în drumul sau stau bătrânii SuperPython, SuperC++ și boss-ul SuperC care încearcă să îi împiedice parcursul.

Prezentare joc:

Jocul constă în trei nivele, fiecare cu piedicile sale diferite și obstacole. Nivelurile sunt împărțite în două lumi cu diferite teme, primul nivel fiind o pădure unde se afla SuperPython, nivelul doi având loc într-un deșert unde se află SuperC++, iar în nivelul final, amplasat in “Lumea de jos”, se află SuperC.

Reguli joc:

Jocul implică parcursul logic al prințesei SuperJava în care jucătorul trebuie să învingă obstacolele de pe drum și să ajungă la boss-ul din acel nivel. După ce este învins boss-ul, jucătorul se va deplasa către o ușă care va deschide drumul pentru următorul nivel. Jucătorul poate ridica anumite puteri care îl vor face mai puternic sau colecta monezi pentru a câștiga vieți în plus. Jucătorul este ucis daca pierde viața în lupta sa către boss. În momentul drumul său spre ieșire, dacă jucatorul este atins de boss, atunci va muri.

Personajele jocului:

- **SuperJava:** este protagonista jocului care este controlat de jucător. Ea lupta pentru supremația sa în fața inamicilor.

- **SuperPython:** boss-ul primului nivel. Rolul acestuia este de a o bloca pe prințesă în ascensiunea sa.

- **SuperC:** boss-ul celui de-al doilea nivel. El este mai puternic decât SuperPython si va fi o provocare pentru jucător.

- **SuperC++:** Boss-ul final, este cel mai puternic dintre toți si va fi cel mai mare obstacol pe care jucătorul va trebui sa îl învingă pentru a termina jocul.

Obstacole si puteri:

În timp ce jucătorul își parcurge drumul prin fiecare nivel, va trebui să treacă prin obstacole precum gropi, pietre sau lacuri adânci.

De-a lungul drumului, jucătorul poate sa colecteze puteri speciale care ii vor oferi mai multă forță sau viteză, precum și monezi pentru a câștiga vieți în plus. Jucătorul poate folosi și un atac special pentru a învinge inamicii mai repede și să își facă drum prin obstacole mai ușor.

Scopul jocului:

Scopul jocului este de a ajuta SuperJava să învingă toți inamicii și să devină stăpânul absolut al limbajelor de programare. Pentru a face acest lucru, jucătorul va trebui să parcurgă toate cele trei nivele și să învingă fiecare boss. Jucătorul va trebui

sa fie atent și să folosească puterile și atacurile sale în mod strategic pentru a învinge inamicii și a ajunge la sfârșitul jocului cu cat mai multe vieți.

Mod de joc:

Jocul este un joc tip platforma, în care jucătorul controlează personajul principal, SuperJava, prin intermediul tastelor de săgeți pentru a se deplasa în stânga, dreapta, sus sau jos.

Jucătorul poate sări peste obstacole și poate ataca inamicii prin apăsarea tastei Space.

Jucătorul poate colecta puteri speciale și monezi pentru a câștiga vieți în plus și a deveni mai puternic în lupta împotriva inamicilor.

În cazul în care jucătorul pierde o viață, acesta poate sa reîncarce nivelul și să încerce din nou.

Tabela de joc va conține:

- Fundalul nivelului: tableta de joc va afișa fundalul specific fiecărui nivel (pădure, deșert, lumea de jos, etc.) care va oferi o atmosferă distinctă și va spori experiența jucătorului.
- SuperJava: personajul principal al jocului, care va fi plasat pe tabla de joc în centrul ecranului. Jucătorul va controla SuperJava și va trebui să o ghideze prin nivel, evitând obstacolele și înfruntând inamicii.
- Inamicii: SuperPython, SuperC++ și SuperC, fiecare dintre aceștia fiind boss-ul propriu-zis pentru fiecare nivel. Inamicii vor fi plasați pe tabla de joc în poziții strategice și vor ataca personajul jucătorului cu arme sau abilități speciale.
- Obiecte colectabile: monezile și puterile speciale vor fi plasate pe tabla de joc și jucătorul va trebui să le colecteze pentru a avansa prin nivel și a-și îmbunătăți performanța.
- Obstacole: tabla de joc va conține obstacole precum gropi, stânci sau alte obstacole care vor încetini progresul jucătorului sau îi vor reduce viața.

Schița interfeței cu utilizatorul pentru joc va arăta astfel:

1. Meniul principal:
   - Buton de pornire a jocului
   - Buton de setări (setări de sunet, grafice, tastatură)
   - Buton de ieșire din joc

2. Ecranul de joc:
   - Bara de viață a personajului principal, SuperJava
   - Indicatorul monezilor colectate
   - Indicatorul vieților rămase
   - Tasta de pauză

3. Ecranul de joc în timpul jocului:
    - Imaginea de fundal a nivelului curent (pădure, deșert, lumea de jos..)
    - Inamicii și obstacolele pe care jucătorul trebuie să le evite
    - Monezile și puterile speciale pe care jucătorul trebuie să le colecteze
    - Indicatorul vieților și al puterilor speciale pe care jucătorul le-a colectat
    -

Sprite-uri utilizate în joc:

SuperJava:

Rândul 1: animație pentru stând pe loc
Rândul 2: animație pentru alergat
Rândul 3: animație pentru săritură
Rândul 4: animație pentru cădere
Rândul 5: animație pentru momentul când ajung pe loc drept
Rândul 6: animație pentru lovitură
Rândul 7: animație pentru atac

Rândul 8: animație pentru atac în aer (1)
Rândul 9: animație pentru atac în aer (2)

SuperC:

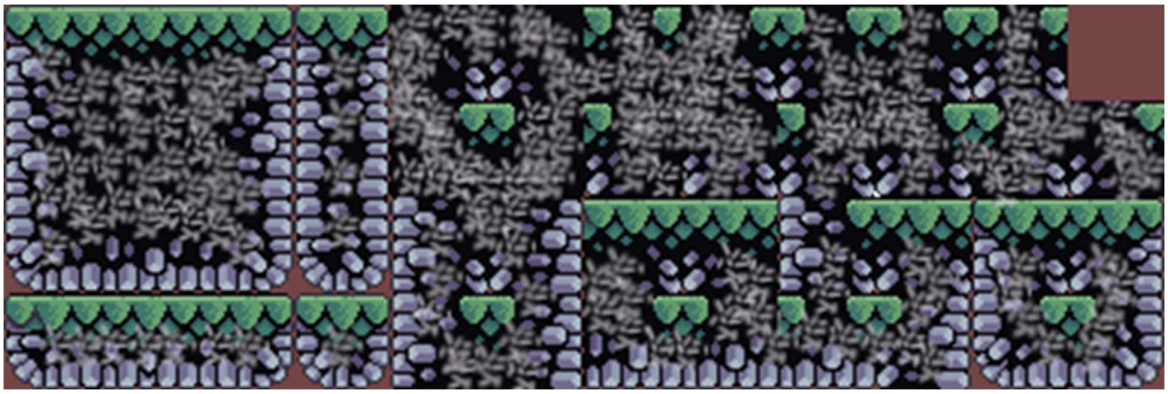
Rândul 1: animație pentru mers
Rândul 2: animație pentru alergat
Rândul 3: animație pentru atac
Rândul 4: animație pentru momentul când primește damage
Rândul 5: animație pentru momentul când moare

SuperC++:

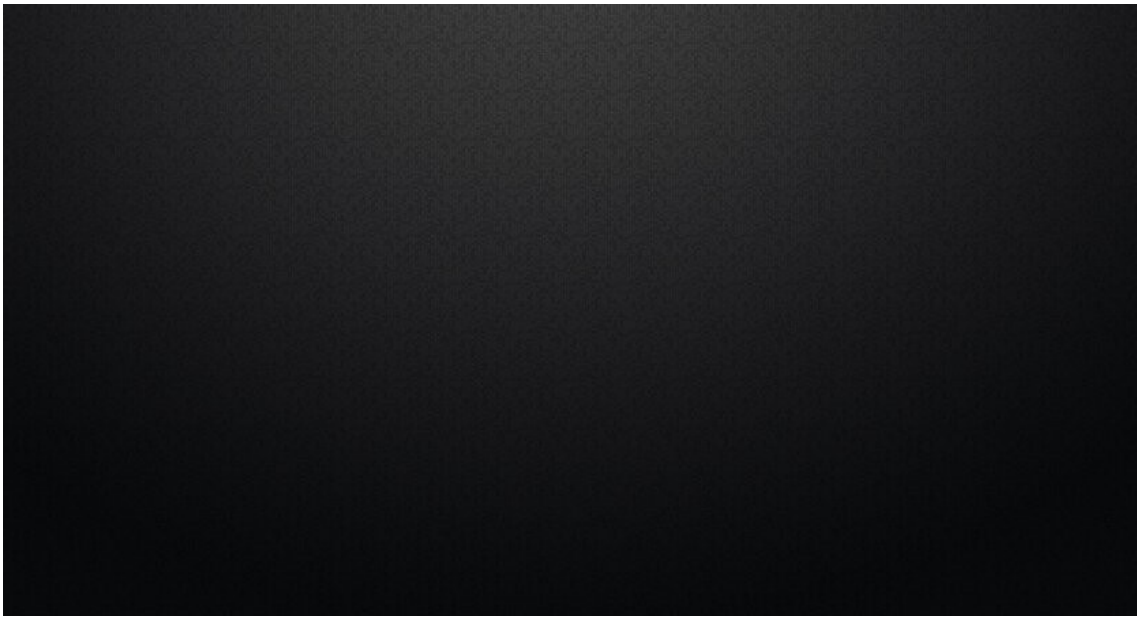
Rândul 1: animație pentru mers

Rândul 2: animație pentru alergat
Rândul 3: animație pentru atac
Rândul 4: animație pentru momentul când primește damage
Rândul 5: animație pentru momentul când moare

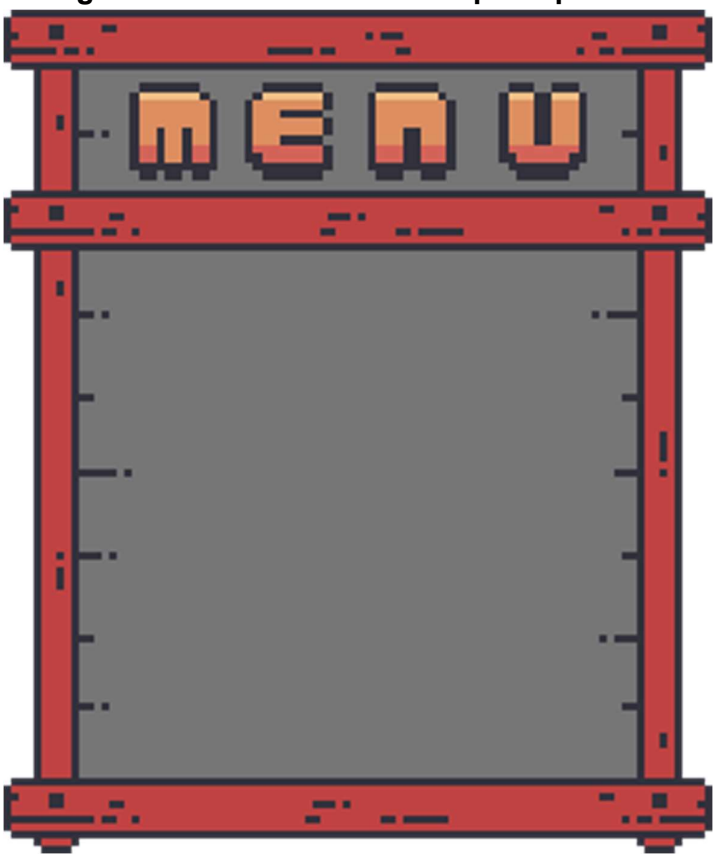
Tile-urile pentru hartă:



Background meniul principal:



Background butoane din meniul principal:



Background butoane din meniul de opțiuni:

Meniu pauză:

Pentru level terminat:

Atunci când mor:

Când jocul este finalizat:

Tun:

Bila tunului:

Butoane:

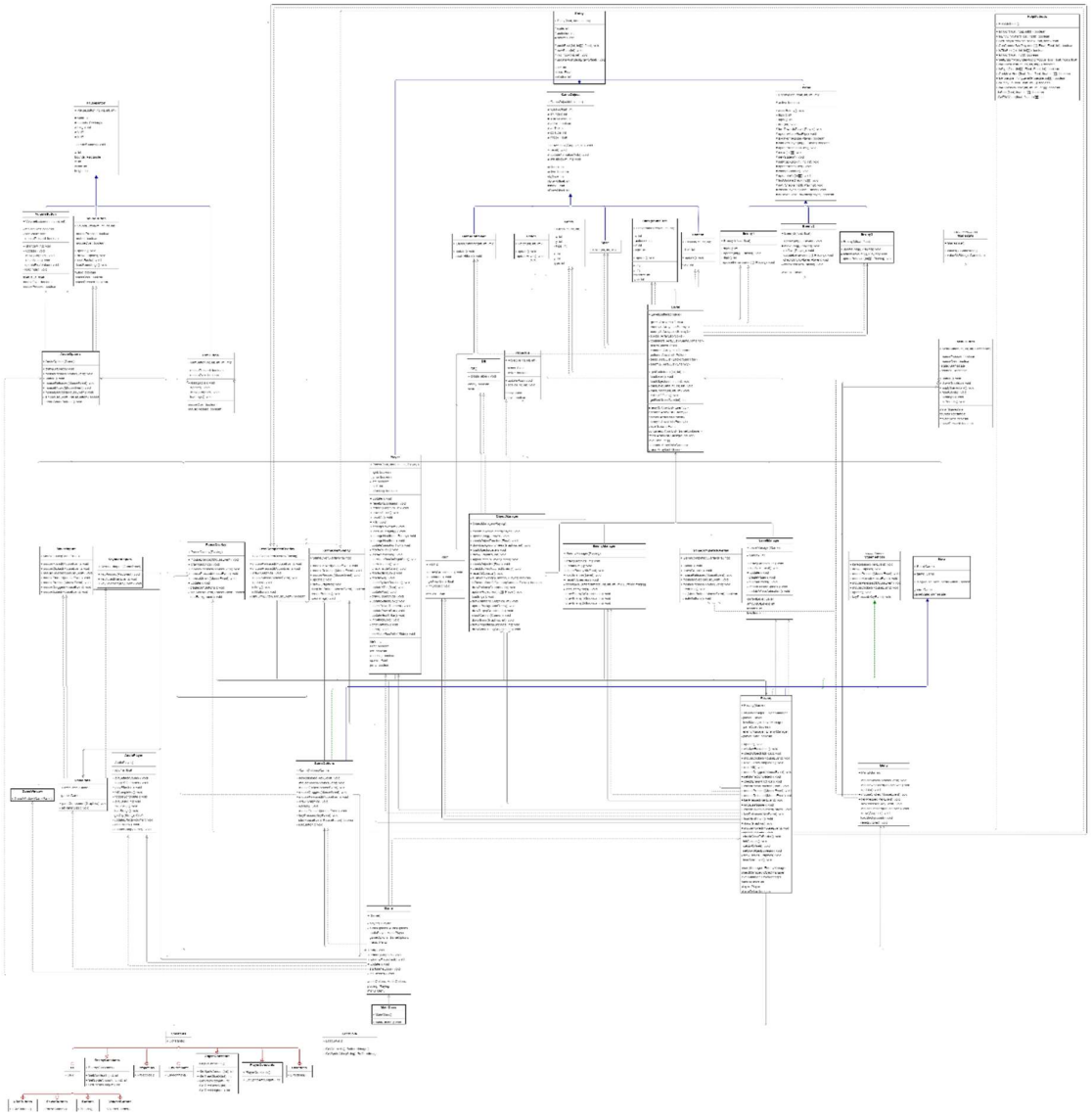
Diverse obiecte:

Poțiuni:

Țepi:

Copaci:

Diagrama UML:



Descrierea fiecărei clase:

- Clasa **AudioPlayer** este responsabilă de gestionarea redării sunetelor și efectelor audio într-un joc. Aceasta utilizează obiecte Clip pentru a încărca și reda coloana sonoră a jocului, precum și efectele sonore asociate acțiunilor din joc. Clasa permite încărcarea și redarea diferitelor piese de muzică și efecte sonore, ajustarea volumului general, controlul mutării sunetelor și efectelor, precum și comutarea între piesele de muzică corespunzătoare diferitelor nivele de joc. De asemenea, clasa oferă metode pentru redarea sunetelor de atac și semnalizarea finalizării unui nivel prin intermediul efectului sonor corespunzător.
- Clasa **DB** este responsabilă de gestionarea conexiunii și interacțiunii cu o bază de date SQLite în cadrul aplicației. Aceasta oferă metode pentru a verifica dacă tabela GAME_INFO este goală, a crea tabela SCORE în baza de date și a insera o înregistrare cu valorile implicite, a obține scorul din tabela SCORE, precum și a actualiza scorul în tabela respectivă. Clasa utilizează obiecte Connection și Statement pentru a stabili conexiunea cu baza de date și a executa instrucțiunile SQL corespunzătoare.
- Clasa **Rain** reprezintă un efect vizual de ploaie într-un joc. Aceasta utilizează o imagine a unei picături de ploaie și generează și actualizează pozițiile picăturilor pe ecran. Metodele clasei permit inițializarea picăturilor de ploaie, actualizarea lor în funcție de o viteză specificată și desenarea lor pe ecran. Prin intermediul acestei clase, este posibilă adăugarea unui efect de ploaie realist într-un joc.
- Clasa **Enemy** este o clasă abstractă care servește ca bază pentru diferite tipuri de inamici într-un joc. Această clasă conține funcții pentru gestionarea mișcării, atacului și stării inamicilor. Ea include funcții pentru actualizarea dreptunghiului de atac al inamicului, verificarea coliziunilor și efectuarea acțiunilor specifice în funcție de starea inamicului. Clasa conține, de asemenea, funcții pentru gestionarea animațiilor inamicilor și pentru determinarea direcției de mers. Există și funcții pentru verificarea interacțiunii cu jucătorul și pentru gestionarea sănătății inamicilor. Clasa Enemy este o clasă de bază flexibilă și extensibilă, care poate fi extinsă pentru a crea diferite tipuri de inamici cu comportamente și abilități specifice.
- Clasa **Enemy1** este o clasă care extinde clasa Enemy și reprezintă un tip specific de inamic într-un joc. Această clasă conține funcții pentru actualizarea comportamentului inamicului, gestionarea stărilor și acțiunilor specifice acestui tip de inamic. Aceasta include și funcții pentru inițializarea hitbox-ului și a zonei de atac a inamicului, precum și funcții pentru fliparea și scalarea graficii inamicului în funcție de direcția sa de mers.
- Clasa **Enemy2** reprezintă o implementare specifică a unui inamic de tipul 2 într-un joc. Această clasă extinde clasa Enemy și conține metode și logica necesară pentru comportamentul inamicului de tip 2. Inamicul poate fi în stări precum IDLE (inactiv), RUNNING (se deplasează), ATTACK (atacă) și HIT (lovit). Clasa gestionează actualizarea comportamentului inamicului în funcție de starea sa curentă, interacțiunea cu jucătorul și mediul înconjurător. De exemplu, inamicul poate detecta jucătorul, îi poate provoca daune și poate efectua acțiuni specifice precum rularea în timpul atacului. De asemenea, clasa conține funcții pentru setarea direcției de deplasare a inamicului și verificarea coliziunilor cu alte obiecte din joc.

- Clasa **Enemy3** extinde clasa Enemy și definește comportamentul specific al inamicului de tip 3 într-un joc. Clasa conține funcții pentru actualizarea stării și animației inamicului, precum și pentru gestionarea comportamentului acestuia în diferite situații. Acestea includ verificarea interacțiunii cu mediul și cu jucătorul, deplasarea inamicului și efectuarea atacurilor. Clasa implementează și funcții specifice de atac pentru inamicul de tip 3.
- Clasa **EnemyManager** este responsabilă de gestionarea și actualizarea inamicilor dintr-un joc. Aceasta conține metode pentru încărcarea inamicilor, actualizarea stării lor, desenarea lor pe ecran, verificarea coliziunilor cu caseta de atac și resetarea lor. Clasa conține, de asemenea, metode auxiliare pentru încărcarea imaginilor inamicilor și obținerea unui tablou bidimensional de imagini dintr-un atlas. Scopul principal al clasei este să mențină și să gestioneze toți inamicii în cadrul jocului, facilitând interacțiunea și sincronizarea cu nivelurile și alte componente ale jocului.
- Clasa **Entity** este o clasă abstractă care servește drept clasă de bază pentru alte entități dintr-un joc. Aceasta conține variabile și metode care sunt comune tuturor entităților, cum ar fi poziția, dimensiunile, hitbox-ul, animația și starea entității. De asemenea, conține metode pentru gestionarea coliziunilor și a efectelor de deplasare. Clasa Entity nu poate fi instantiată direct, ci este destinată să fie extinsă de alte clase care reprezintă entități specifice în joc.
- Clasa **Player** este o entitate care reprezintă jucătorul într-un joc. Aceasta are funcționalități precum gestionarea animațiilor, detectarea coliziunilor, atacarea inamicilor și gestionarea stării de sănătate și putere a jucătorului. Clasa conține, de asemenea, metode pentru actualizarea poziției jucătorului, verificarea coliziunilor cu obiecte din joc și desenarea jucătorului și interfeței grafice asociate (bară de sănătate și bară de putere). Jucătorul poate să se miște în stânga și în dreapta, să sare și să atace. De asemenea, poate colecta obiecte (potion-uri) și poate fi afectat de inamici (reduce sănătatea).
- Clasa **GameOptions** este o clasă Java care reprezintă starea de opțiuni a unui joc. Aceasta extinde clasa State și implementează interfața Statemethods. Clasa conține funcționalități pentru încărcarea imaginilor de fundal și a butoanelor, actualizarea și desenarea stării opțiunilor jocului, gestionarea evenimentelor de mouse și tastatură.
- Clasa **Gamestate** este o enumerare care definește diferitele stări posibile ale jocului. Aceste stări includ "PLAYING" (jocul este în desfășurare), "MENU" (meniul jocului), "OPTIONS" (opțiunile jocului) și "QUIT" (ieșirea din joc). Variabila statică state este utilizată pentru a reprezenta starea curentă a jocului și este inițializată cu valoarea "MENU" (meniul jocului). Această variabilă poate fi accesată și modificată din alte clase pentru a controla fluxul jocului și pentru a naviga între diferitele stări.
- Clasa **Menu** este responsabilă de gestionarea stării meniului într-un joc. Ea se ocupă de încărcarea și afișarea fundalului și a butoanelor din meniu, precum și de detectarea și gestionarea evenimentelor de interacțiune cu utilizatorul, cum ar fi apăsarea unui buton de mouse sau tastă. Clasa oferă funcționalitatea necesară pentru actualizarea stării butoanelor și a meniului în ansamblu, în funcție de acțiunile utilizatorului. Astfel, clasa Menu permite utilizatorului să navigheze și să interacționeze cu diferitele opțiuni disponibile în meniu, oferind o interfață intuitivă și plăcută pentru experiența de joc.
- Clasa **Playing** reprezintă un manager de joc responsabil de gestionarea diferitelor aspecte ale jocului, precum desenarea norilor, controlul stării jocului (game over,

paused, level completed, game completed), manipularea evenimentelor de mouse și tastatură, manipularea stării jucătorului și a inamicilor, precum și gestionarea nivelurilor și obiectelor din joc.

- Clasa **State** este o clasă de bază pentru gestionarea stărilor jocului în cadrul aplicației. Aceasta oferă funcționalități generale precum verificarea interacțiunilor cu butoanele de meniu și setarea stării curente a jocului.
- Clasa **KeyboardInputs** este responsabilă pentru gestionarea intrărilor de la tastatură într-un joc. Implementează interfața KeyListener și are trei metode principale: keyTyped, keyPressed și keyReleased.
- Clasa **MouseInputs** gestionează intrările de la mouse într-un joc. Implementează interfețele MouseListener și MouseMotionListener și are mai multe metode care răspund la diferite evenimente legate de mouse, cum ar fi clicuri, tragerea mouse-ului sau mișcarea cursorului. Clasa este responsabilă pentru comutarea între diferitele stări ale jocului (MENU, PLAYING, OPTIONS) și apelarea metodelor corespunzătoare din starea curentă a jocului pentru a manipula evenimentele mouse-ului în consecință. Prin intermediul clasei MouseInputs, intrările de la mouse sunt direcționate către componentele corecte ale jocului și acțiunile corespunzătoare sunt luate în funcție de starea curentă a jocului.
- Clasa **Level** este responsabilă de gestionarea și încărcarea datelor pentru un nivel într-un joc. Aceasta utilizează o imagine pentru a defini aspectul nivelului și încarcă datele corespunzătoare, inclusiv terenul, entitățile, obiectele și alte caracteristici. Prin intermediul metodei loadLevel(), clasa parcurge fiecare pixel din imagine și extrage valorile componente Red, Green și Blue pentru a determina tipurile de teren, entități și obiecte prezente în nivel. De exemplu, valorile Red pot reprezenta diferite tipuri de iarbă, valorile Green pot indica tipurile de inamici, iar valorile Blue pot reprezenta obiecte precum poțiuni, containere sau tunuri. Clasa oferă, de asemenea, metode pentru a obține informații despre nivel, cum ar fi datele nivelului, offseturile nivelului, poziția de spawn a jucătorului și liste de entități sau obiecte prezente în nivel.
- Clasa **LevelManager** este responsabilă de gestionarea nivelurilor din joc. Aceasta păstrează o referință către obiectul jocului (Game), încarcă și stochează sprite-urile nivelurilor, gestionează animația apei și permite încărcarea nivelului următor. Clasa conține și o listă de obiecte Level, unde fiecare obiect Level reprezintă un nivel din joc.
- Clasa **Game** este clasa principală a jocului, responsabilă de gestionarea componentelor și logicii jocului. Aceasta inițializează și controlează instanțe ale altor clase precum Menu, Playing, GameOptions, AudioOptions și AudioPlayer. De asemenea, implementează interfața Runnable pentru a permite rularea jocului într-un fir de execuție separat. Clasa Game conține metode pentru actualizarea și desenarea jocului, precum și un ciclu de joc în metoda run().
- Clasa **GamePanel** extinde clasa JPanel și reprezintă panoul de joc în care se desenează elementele grafice ale jocului. Aceasta gestionează evenimentele de tastatură și mouse și desenează jocul pe panou.
    - *GamePanel(Game game):* Constructorul clasei care primește obiectul Game și inițializează obiectul MouseInputs. Setează dimensiunea panoului conform dimensiunii jocului și adaugă ascultători pentru tastatură și mouse.
    - *setPanelSize():* Metodă privată care setează dimensiunea panoului la dimensiunea jocului specificată în obiectul Game.

    - *paintComponent(Graphics g):* Suprascrierea metodei paintComponent pentru a desena componentele grafice pe panou. Aceasta apelează metoda render a obiectului Game pentru a desena jocul.
    - *getGame():* Metodă care returnează obiectul Game asociat panoului.
- Clasa **GameWindow** este responsabilă de crearea și gestionarea ferestrei de joc. Aceasta creează un obiect JFrame și îi configurează diversele proprietăți, precum titlul, dimensiunea, rezizabilitatea și comportamentul la închidere. De asemenea, adaugă un obiect GamePanel la fereastra principală. De asemenea, clasa implementează interfața WindowFocusListener pentru a gestiona evenimentele de când fereastra pierde sau câștigă focus-ul, permițând astfel manipularea corectă a jocului.
- Clasa **BackgroundTree** din pachetul objects reprezintă o implementare a unui obiect de fundal reprezentând un copac. Această clasă conține variabilele membru x, y, type, aniIndex și aniTick, care stochează informații despre poziția copacului, tipul acestuia și starea animației. Clasa are un constructor care primește coordonatele x și y ale copacului, precum și un tip, și inițializează variabilele membru în consecință. De asemenea, constructorul generează un număr aleatoriu pentru variabila membru aniIndex. Clasa BackgroundTree conține metode pentru a accesa și a modifica variabilele membru, cum ar fi getAniIndex(), setAniIndex(), getX(), setX(), getY(), setY(), getType() și setType(). Aceste metode permit obținerea și actualizarea valorilor asociate obiectului BackgroundTree. Clasa conține și metoda update(), care este responsabilă de actualizarea stării copacului. Aceasta incrementează aniTick și verifică dacă trebuie să actualizeze aniIndex și să reseteze aniTick la anumite intervale.
- Clasa **GameContainer** extinde clasa GameObject și conține variabilele membru și metodele specifice pentru gestionarea unui obiect de tip container în joc. Constructorul se ocupă de inițializarea obiectului, iar metoda createHitbox() creează hitbox-ul în funcție de tipul obiectului. Metoda update() actualizează starea obiectului în funcție de variabila doAnimation.
- Clasa **GameObject** servește ca o clasă de bază pentru alte obiecte în joc. Aceasta conține variabilele membru și metodele comune utilizate în gestionarea obiectelor din joc, cum ar fi inițializarea hitbox-ului, actualizarea animației și furnizarea accesului la informații despre obiect (tipul, hitbox-ul, starea etc.).
- Clasa **Grass** reprezintă o entitate "iarbă" în contextul aplicației. Aceasta stochează informații despre poziția și tipul ierbii. Constructorul permite inițializarea obiectului Grass cu valorile corespunzătoare, iar metodele getter furnizează acces la aceste valori.
- Clasa **ObjectManager** este responsabilă de gestionarea obiectelor din joc, inclusiv poțiuni, containere, capcane, tunuri și proiectile. Aceasta se ocupă de coliziuni, efectele obiectelor și actualizarea stării acestora în funcție de nivelul de date și jucătorul curent. De asemenea, se ocupă de încărcarea imaginilor necesare pentru obiecte și de actualizarea arborilor din fundal. Clasa conține metode pentru verificarea coliziunilor cu obiecte specifice, cum ar fi obiectele de tip "spike" sau "potion", și pentru aplicarea efectelor acestora asupra jucătorului. De asemenea, gestionează proiectilele lansate de tunuri și efectele acestora asupra jucătorului și nivelului.

- Clasa **Potion** extinde clasa GameObject și reprezintă un obiect de tip poțiune într-un joc. Această clasă conține variabile și funcții specifice manipulării și afișării poțiunii. Variabilele hoverOffset, maxHoverOffset și hoverDir controlează animația de hover a poțiunii. Funcțiile din clasa Potion se ocupă de actualizarea stării și a offset-ului de hover al poțiunii.
- Clasa **Projectile** reprezintă un obiect proiectil într-un joc. Are un obiect hitbox de tip Rectangle2D.Float care reprezintă zona de coliziune a proiectilului. Proiectilul are o direcție și poate fi activ sau inactiv.
- Clasa **Spike** extinde clasa GameObject și reprezintă un obiect spike într-un joc. Această clasă are un constructor care primește poziția inițială și tipul obiectului spike. Constructorul inițializează hitbox-ul obiectului și setează offset-ul de desenare. Clasa nu conține alte funcții sau metode, astfel încât majoritatea funcționalității este moștenită din clasa de bază.
- Clasa **AudioOptions** este responsabilă de gestionarea opțiunilor audio în cadrul jocului. Aceasta oferă funcționalitatea de a crea și afișa butoane pentru setările de volum și sunet (muzică și efecte sonore). Clasa permite actualizarea stării butoanelor în funcție de evenimentele de interacțiune cu mouse-ul și desenarea lor pe ecran. De asemenea, gestionează acțiunile asociate apăsării și eliberării butonului de mouse, precum modificarea volumului sau activarea/dezactivarea sunetului. Scopul acestei clase este de a oferi o interfață utilizatorului pentru a controla setările audio în cadrul jocului.
- Clasa **GameCompletedOverlay** este responsabilă de afișarea și gestionarea ecranului de finalizare a jocului în cadrul unei aplicații Java. Aceasta conține metode și funcționalități pentru desenarea imaginii de fundal, crearea și afișarea butonului de ieșire din joc și tratarea evenimentelor de mouse.
- Clasa **GameOverOverlay** este responsabilă de afișarea și gestionarea ecranului de înfrângere într-un joc. Aceasta conține metode și funcționalități pentru desenarea imaginii de fundal, crearea și afișarea butoanelor de meniu și de reluare a jocului, precum și tratarea evenimentelor de mouse.
- Clasa **LevelCompletedOverlay** este responsabilă de afișarea și gestionarea ecranului de finalizare a nivelului într-un joc. Aceasta conține metode și funcționalități pentru desenarea imaginii de fundal, crearea și afișarea butoanelor de meniu și de nivel următor, precum și tratarea evenimentelor de mouse.
- Clasa **MenuButton** este responsabilă de gestionarea și afișarea butoanelor din meniul jocului. Aceasta conține funcții pentru încărcarea imaginilor butoanelor, desenarea butoanelor pe ecran, actualizarea stării butoanelor în funcție de interacțiunea cu cursorul mouse-ului, gestionarea evenimentelor mouse-ului și obținerea dreptunghiului de coliziune al butoanelor. Clasa este utilizată în interfața utilizatorului pentru a permite utilizatorului să selecteze opțiuni din meniu și să schimbe starea jocului în funcție de butonul apăsat.
- Clasa **PauseButton** reprezintă un buton utilizat pentru funcționalitatea de pauză într-un joc. Această clasă definește coordonatele, dimensiunile și dreptunghiul de coliziune al butonului. Ea oferă metode pentru obținerea și setarea proprietăților butonului, precum și accesul la dreptunghiul de coliziune. Clasa PauseButton furnizează funcționalitatea de bază necesară pentru gestionarea butonului de pauză în interfața utilizatorului.

- Clasa **PauseOverlay** este responsabilă de gestionarea ecranului de pauză într-un joc. Aceasta afișează imaginea de fundal a ecranului de pauză și conține butoane pentru revenirea la meniu, reluarea nivelului și reluarea jocului. De asemenea, clasa gestionează evenimentele de mouse precum apăsarea, eliberarea și deplasarea, pentru a reacționa la acțiunile utilizatorului. Clasa include, de asemenea, o componentă pentru opțiunile audio. În ansamblu, clasa PauseOverlay facilitează interacțiunea utilizatorului cu ecranul de pauză și controlează acțiunile aferente, precum revenirea la meniu sau reluarea jocului.
- Clasa **SoundButton** este o clasă care reprezintă un buton de sunet în cadrul unui meniu de pauză. Aceasta extinde clasa PauseButton și conține funcționalități pentru încărcarea imaginilor butonului, actualizarea stării și afișarea grafică a acestuia. De asemenea, permite gestionarea stării de hover și apăsare a butonului, precum și activarea sau dezactivarea sunetului. Clasa SoundButton oferă metode pentru interacțiunea cu starea butonului și permite desenarea acestuia pe ecran.
- Clasa **UrmButton** extinde clasa PauseButton și reprezintă un buton specific pentru meniul de pauză al unei aplicații. Această clasă gestionează afișarea și interacțiunea utilizatorului cu butonul, inclusiv stările de hover și apăsare.
- Clasa **VolumeButton** este o subclasă a clasei PauseButton și reprezintă un buton de volum într-o interfață grafică. Această clasă oferă funcționalitate pentru gestionarea și afișarea stărilor butonului de volum, precum și pentru calcularea și obținerea valorii volumului în format float.
- În cadrul clasei **Constants**, sunt definite mai multe clase interne care grupează constantele în funcție de domeniul lor de utilizare. Mai jos sunt prezentate câteva detalii despre fiecare clasă internă și constantele definite în cadrul acestora:
    - Clasa ***Projectiles*** definește constante legate de proiectilele folosite în joc, cum ar fi dimensiunile, viteza etc.
    - Clasa ***ObjectConstants*** definește constante legate de obiectele din joc, cum ar fi potiunile, butoaiele, cutiile, cuțitele etc. Aceste constante includ dimensiuni implicite și dimensiuni scalate în funcție de valoarea Game.SCALE. De asemenea, sunt definite și alte metode care returnează diferite proprietăți ale obiectelor în funcție de tipul obiectului.
    - Clasa ***EnemyConstants*** definește constante legate de inamicii din joc, cum ar fi dimensiunile, stările, sănătatea etc. Sunt definite metode pentru obținerea numărului de cadre ale animației și alte proprietăți specifice inamicilor.
    - Clasa ***Environment*** definește constante legate de mediul din joc, cum ar fi dimensiunile norilor etc.
    - Clasa ***UI*** definește constante legate de interfața utilizatorului din joc, cum ar fi dimensiunile butoanelor, dimensiunile butoanelor de pauză, dimensiunile butoanelor de volum etc.
    - Clasa ***Directions*** definește constante pentru direcțiile de bază, cum ar fi stânga, sus, dreapta, jos.
    - Clasa ***PlayerConstants*** definește constante legate de jucătorul din joc, cum ar fi stările, acțiunile, numărul de cadre ale animației etc.
- Clasa **HelpMethods** din pachetul utilz conține o serie de metode statice utile pentru manipularea datelor de joc și pentru efectuarea diverselor verificări în cadrul unui joc Java. Această clasă oferă funcționalități cum ar fi:

- Verificarea posibilității de deplasare a unui obiect într-o anumită poziție, ținând cont de obiectele solide din nivel ***(CanMoveHere)***.
- Verificarea dacă un proiectil lovește nivelul ***(IsProjectileHittingLevel)***.
- Verificarea dacă o entitate se află în apă ***(IsEntityInWater)***.
- Determinarea valorii unui tile în funcție de poziția sa ***(GetTileValue)***.
- Verificarea dacă un tile este solid ***(IsTileSolid)***.
- Determinarea poziției X a unei entități în apropierea unui perete ***(GetEntityXPosNextToWall)***.
- Determinarea poziției Y a unei entități sub un acoperiș sau deasupra podelei ***(GetEntityYPosUnderRoofOrAboveFloor)***.
- Verificarea dacă o entitate se află pe podea ***(IsEntityOnFloor)***.
- Verificarea existenței unei podele sub o entitate ***(IsFloor)***.
- Verificarea dacă un tun poate vedea un jucător în nivel ***(CanCannonSeePlayer)***.
- Verificarea dacă toate tile-urile într-un anumit interval sunt libere ***(IsAllTilesClear)***.
- Verificarea dacă toate tile-urile dintr-un interval pot fi parcurse ***(IsAllTilesWalkable)***.
- Verificarea existenței unei vederi clare între două hitbox-uri ***(IsSightClear)***.

- Clasa **LoadSave** este responsabilă de încărcarea și salvarea datelor în joc. Aceasta conține două funcții principale:
  - Funcția ***GetSpriteAtlas(String fileName)*** primește numele unui fișier de imagine și returnează o imagine BufferedImage care reprezintă un atlas de sprite-uri. Funcția încearcă să încarce imaginea din fișierul specificat utilizând clasa ImageIO. Această funcție poate fi folosită pentru a obține atlasuri de sprite-uri pentru diferite elemente grafice din joc, cum ar fi personaje, obiecte sau efecte vizuale.
  - Funcția ***GetAllLevels()*** returnează un tablou de imagini BufferedImage care reprezintă toate nivelele disponibile în joc. Această funcție caută fișierele de nivel într-un director specific (/res/lvls/), le sortează în ordine și le încarcă într-un tablou de imagini. Această funcție permite obținerea nivelurilor jocului pentru a le utiliza ulterior în procesul de joc.

Concluzie:

Super Java Battle este un joc captivant de platformă, cu o poveste interesantă și personaje amuzante și pline de viață.
Jocul oferă o experiență distractivă și provocatoare pentru jucătorii de toate vârstele, cu niveluri diferite si obstacole de depășit.

SuperJava este un personaj puternic si îndrăzneț, care luptă pentru a deveni stăpânul absolut al limbajelor de programare, și jucătorii trebuie sa o ajute pentru a-și îndeplini visul.

Bibliografie:

- ✓ https://www.istockphoto.com/ro/vector/anima%C8%9Bia-monedelor-pixel-art-joc-ui-monede-de-aur-etape-de-rota%C8%9Bie-pixel-joc-bani-gm1319250392-406165003
- ✓ https://www.flaticon.com/free-icons/health-bar
- ✓ https://www.reinerstilesets.de/graphics/2d-grafiken/2d-humans/
- ✓ https://80.lv/articles/pyxel-edit-a-pixel-art-tool/?amp_markup=1
- ✓ https://opengameart.org/content/2d-simple-grass-tileset
- ✓ https://www.youtube.com/playlist?list=PL_QPQmz5C6WUF-pOQDsbsKbaBZqXj4qSq
- ✓ https://drive.google.com/drive/folders/1OBRM8M3qCNAfJDCaldg62yFMiyFaKgYx
- ✓ https://www.kaaringaming.com/platformer-tutorial
- ✓ https://www.youtube.com/watch?v=6_N8QZ47toY&list=PL4rzdwizLaxYmltJQRjq18a9gsSyEQQ-0&ab_channel=KaarinGaming