Feladat

A meséből jól ismert Maci Laci bőrébe bújva a Yellowstone Nemzeti Park megmászhatatlan hegyei és fái között szeretnénk begyűjteni az összes rendelkezésre álló piknik kosarat. Az átjárhatatlan akadályok mellett Yogi élelem szerzését vadőrök nehezítik, akik vízszintesen vagy függőlegesen járőröznek a parkban. Amennyiben Yogi egy egység távolságon belül a vadőr látószögébe kerül, úgy elveszít egy élet pontot. (Az egység meghatározása rád van bízva, de legalább a Yogi sprite-od szélessége legyen.) Ha a 3 élet pontja még nem fogyott el, úgy a park bejáratához kerül, ahonnan indult. A kalandozás során, számon tartjuk, hogy hány piknik kosarat sikerült összegyűjtenie Lacinak. Amennyiben egy pályán sikerül összegyűjteni az összes kosarat, úgy töltsünk be, vagy generáljunk egy új játékteret. Abban az esetben, ha elveszítjük a 3 élet pontunkat, úgy jelenjen meg egy felugró ablak, melyben a nevüket megadva el tudják menteni az aktuális eredményüket az adatbázisba. Legyen egy menüpont, ahol a 10 legjobb eredménnyel rendelkező játékost lehet megtekinteni, az elért pontszámukkal, továbbá lehessen bármikor új játékot indítani egy másik menüből.

A megvalósításról általánosságban

A feladat tehát egy olyan játék elkészítése, amit minden esetben egy játékos játszhat. A program elindítása után egy menü ablakban a felhasználó eldöntheti, hogy kezd egy új játékot, megnézi az eddigi legjobb 10 eredményt, vagy kilép a játékból. Ha az „New Game” menüpontot választja, akkor azonnal generálódik egy új játéktér külön ablakban. A játék közben a menü ablakában akármikor új játékot lehet kezdeni, de ezzel a jelenlegi eredmények törlődnek. A játék akkor ér véget, ha Maci Laci háromszor a vadászok látókörébe kerül. Ezután egy felugró ablakban a játékos megadhatja a nevét és az SQL adatbázisba menti a játék során elért ponttal, majd a játékos a menü ablakból új játékot indíthat. Név hiányában „Anonymous” névvel menti az adatbázisba. A „Statistics” menüpontra kattintva új ablakban megjeleníti az adatbázisban található eddigi legjobb 10 eredményt sorrendben. Az „Exit” menüponttal kilép a felhasználó a programból.

A feladat elemzése

A feladatot három fő lépésben hajtjuk végre:

1. A program elindulásakor megjelenik a menüablak, ahol választhat a felhasználó a menüpontok közül.

2. A „New Game” kiválasztása után megjelenítjük az újonan generált játékot egy másik ablakban.

3. Ha ez megtörtént, akkor a szabályoknak megfelelően megvalósítjuk a játékot.

4. A „Statistics” menüpont alatt megjelenítjük az adatbázisban található eddigi legjobb 10 eredményt egy másik ablakban.

A feladat elemzése

Az első részfeladat elemzése:

Az első részfeladatban tehát egy olyan keretre van szükségünk, ahol a felhasználók könnyedén képesek kiválasztani, hogy új játékot akarnak-e játszani, vagy az eddigi eredményekre kíváncsiak. Ennek a megvalósítása nagyon egyszerűen, két nyomógomb segítségével történik. A „New Game” lenyomásakor új játékteret generálunk és elindul a játék külön ablakban. A „Statistics” lenyomásakor új ablakban megjelenítjük sorrendben az eddigi 10 legjobb eredményt. Ha a menü ablakán kívül aktív egy másik ablak és így kattintanak valamelyik menüpontra, akkor az eddig megnyitott új ablak

bezáródik. Ezen kívül van a menü ablakban egy „Exit” gomb, amellyel bármikor bezárhatja a felhasználó a programot.

A második részfeladat elemzése:

Amint a felhasználó kiválasztotta a „New Game” menüpontot, véletlenszerű játékteret generálunk és megjelenítjük a játékosnak. Amint az adott szintet teljesítette a játékos, új véletlenszerű játékteret generálunk és szintenként nehezedik a játék. A játékos, vagyis „Yogi Bear” elhelyezése minden szint kezdetén adott pozícióra helyezzük, és akkor is, ha életet veszít a játékos. A vadászok, a tereptárgyak és a kosarak véletlenszerűen vannak generálva a pálya egész méretére nézve.

A harmadik részfeladat elemzése:

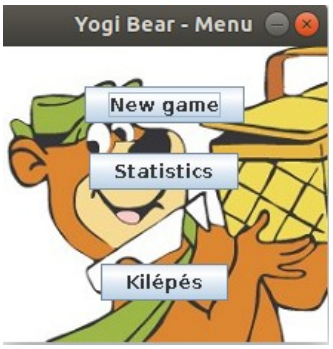
A játék azonnal megkezdődik és addig tart, amíg a játékosnak van rendelkezésre álló életpontja, vagy ki nem lép az ablakból (akár új menüpont kiválasztásával). Életpontot úgy veszít a játékos, ha egy vadász látóterébe kerül. A program Timer segítségével mindig az aktuális helyzetet mutatja. Ha a játékos veszít egy életpontot, akkor a program ellenőrzi, hogy van-e még életpontja a játékosnak. Ha van, akkor a kezdeti pozícióra helyezi a játékost, ellenkező esetben pedig egy felugró ablak jelenik meg, melyben a játékos megadhatja a nevét és mentjük a szükséges adatokat az adatbázisba. A játékos ezután egy „Game Over” feliratot lát a játék ablakában és a menüből tud új játékot kezdeni.

A negyedik részfeladat elemzése:

Amint a felhasználó kiválasztotta a „Statistics” menüpontot, egy új ablak ugrik fel, melyben növekvő sorrendben megjelenítjük az SQL adatbázisban található eddigi legjobb 10 eredményt a játékos nevekkel együtt.

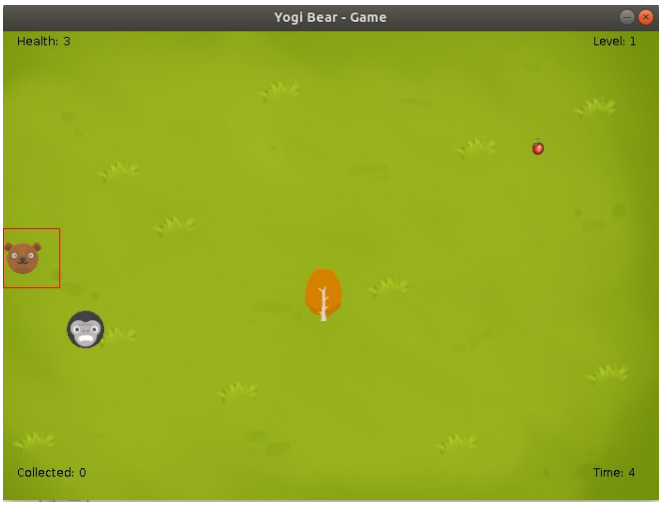
A feladat megvalósítása:

Az első részfeladat:



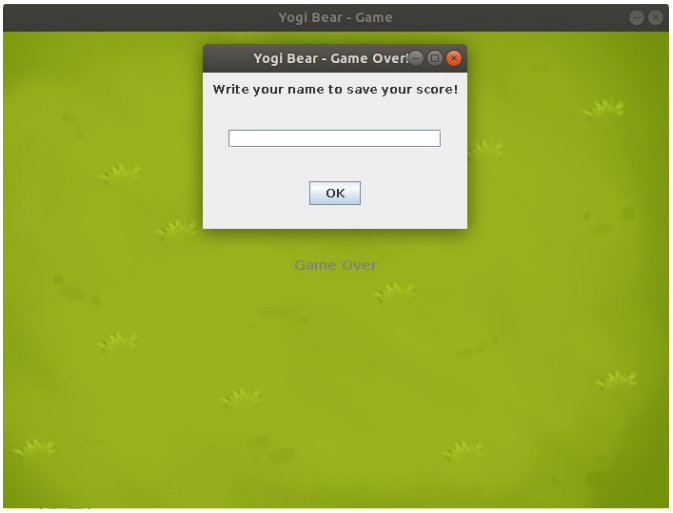
A második és harmadik részfeladat (minta):

Kiindulási helyzet:



A játék vége előtti állapot:

A játék véget ért:



Az osztályok megvalósítása:

A program összesen 15 osztályból áll. Ezek rendre a DatabaseManager, ErrorPopupWindow, GameOverPopup, GameWindow, MainWindow, StatisticsWindow, Basket, Hunter, Sprite, TerrainBlock, Yogibear, ExitButton, GamePanel, MainPanel, Exit. Az ErrorPopupWindow, ExitButton, Exit tartalma elhanyagolható, hiszen az ErrorPopupWindow egy errorMessage-et jelenít csak meg külön ablakban, az ExitButton gomb formájában felhasználja az Exit osztályt a program bezárására, az Exit a program végleges bezárásáért felel.

MainWindow:

A MainWindow osztály tartalmazza a komponenseket, amelyek az üdvözlő képernyőn megjelennek. Főleg mezőkből, valamint egy konstruktorból áll. A JFrame osztály alosztálya, mely megvalósítja az ActionListenert, melyhez tartozik egy actionPerformed() metódus.

- actionPerformed(): A gombokhoz tartozó eseménykezelésért felel.

GameWindow:

A GameWindow osztály, amely a JFrame alosztálya, a játékablak megjelenítéséért felel, a konstruktora létrehoz egy új GamePanel objektumot.

GamePanel:

A GamePanel osztály, amely a Jpanel alosztálya, valósítja meg a játék háttérműveleteit és jeleníti meg. Az osztály megvalósítja az ActionListener osztályt. Az osztály konstruktora generálja a játék alaphelyzetét (meghívja az initBoard() metódust).

- initBoard(): Létrehozza a kezdeti véletlenszerűen generált pályát.
- reInitBoard(): Növeli a pálya nehézségét és generál az új számokkal egy pályát.
- initBaskets(): Létrehozza és elhelyezi a kosarakat.
- initHunters(): Létrehozza és elhelyezi a vadászokat.
- initTerrains(): Létrehozza a természeti blokkokat.
- paintComponent(Graphics g): Kirajzolja a játék aktuális helyzetét.
- drawObject(Graphics g): Kirajzolja az objektumokat (paintComponent hívja meg).
- drawGameOver(Graphics g): A játék végét rajzolja ki. (paintComponent hívja meg).
- actionPerformed(ActionEvent e): Frissíti az objektumokat és meghívja azok újra kirajzolását.
- updateHunters(): Mozgatja a vadászokat.
- updateBaskets(): Eltünteti és törli a kosarakat, ha a játékos felvette őket.
- updateYogi(): Mozgatja a játékost.
- checkCollisions(): Az objektumok ütközését ellenőrzi és végrehajtja a szabályok szerint a helyzet módosítását.
- inGame(): Leállítja a Timer-t, ha vége a játéknak.
- Tadapter osztály, amely bővíti a KeyAdapter osztályt: keyReleased() és keyPressed() eseményeket átadja a Yogibear objektumnak.

GameOverPopup:

A GameOverPopup osztály, amely a JFrame alosztálya, azokat a komponenseket tartalmazza, amelyek segítségével megjelenik a játék végén az ablak, melyben a játékos megadhatja a nevét. Az osztály megvalósítja az ActionListener és a KeyListener osztályokat.

- actionPerformed() : Az „OK” gomb használata után meghívja a DatabaseManager insert() metódusát.
- keyPressed(): Az ENTER megnyomása után meghívja a DatabaseManager insert() metódusát.

StatisticsWindow:

A StatisticsWindow osztály a Jframe osztály alosztálya és megvalósítja az ActionListener() osztályt. A konstruktora meghívja az initStats() metódust.

- initStats(): Beállítja az ablak értékeit és meghívja a DatabaseManager getTop10() metódusát, majd kíírja az ablakba.
- actionPerformed(): Az ablak bezárás gomba miatt.

MainPanel:

A MainPanel a Jpanel alosztálya. A paintComponent() metódussal egészíti ki a Jpanel-t, mely kirajzolja a menü háttérképét.

Basket:

A Basket osztály a Sprite osztály alosztálya. Az initBasket() metódus betölti a kosarak képét.

TerrainBlock:

A TerrainBlock osztály a Sprite osztály alosztálya. Az initTerrainBlock() metódus betölti a blokkok képét. getBounds() metódust túlterheli, ha létrehozáshoz kéri le az objektum méreteit.

Hunter:

A Hunter osztály a Sprite osztály alosztálya.

- initHunter(): metódus betölti a vadászok képét és eldönti, hogy x vagy y irányba haladjon a vadász.
- move(): Mozgatja az objektumot.
- getBoundsWithSight(): visszaadja a vadászok területét a látómezőjükkel együtt.

Yogibear:

A Yogibear osztály a Sprite osztály alosztálya.

- initBear(): Betölti az objektum képét.
- move(): Mozgatja az objektumot.
- keyPressed(): Változtatja a dx és dy értékeket a kapott KeyEventnek megfelelően.
- keyReleased(): Változtatja a dx és dy értékeket 0-ra, ha az eddig lenyomott gomb el lett engedve.

Sprite:

A Sprite osztály konstruktora létrehozza az objektumot kapott x és y input szerinti koordinátára.

- getImageDimensions(): Visszaadja a kép szélességét és magasságát.
- loadImage(String imageName): Betölti a kapott útvonalon elérhető képet az objektum képeként.
- getBounds(): Visszaadja a kép méretével létrehozott téglalapot.

DatabaseManager:

A DatabaseManager alap konstruktorral rendelkezik.

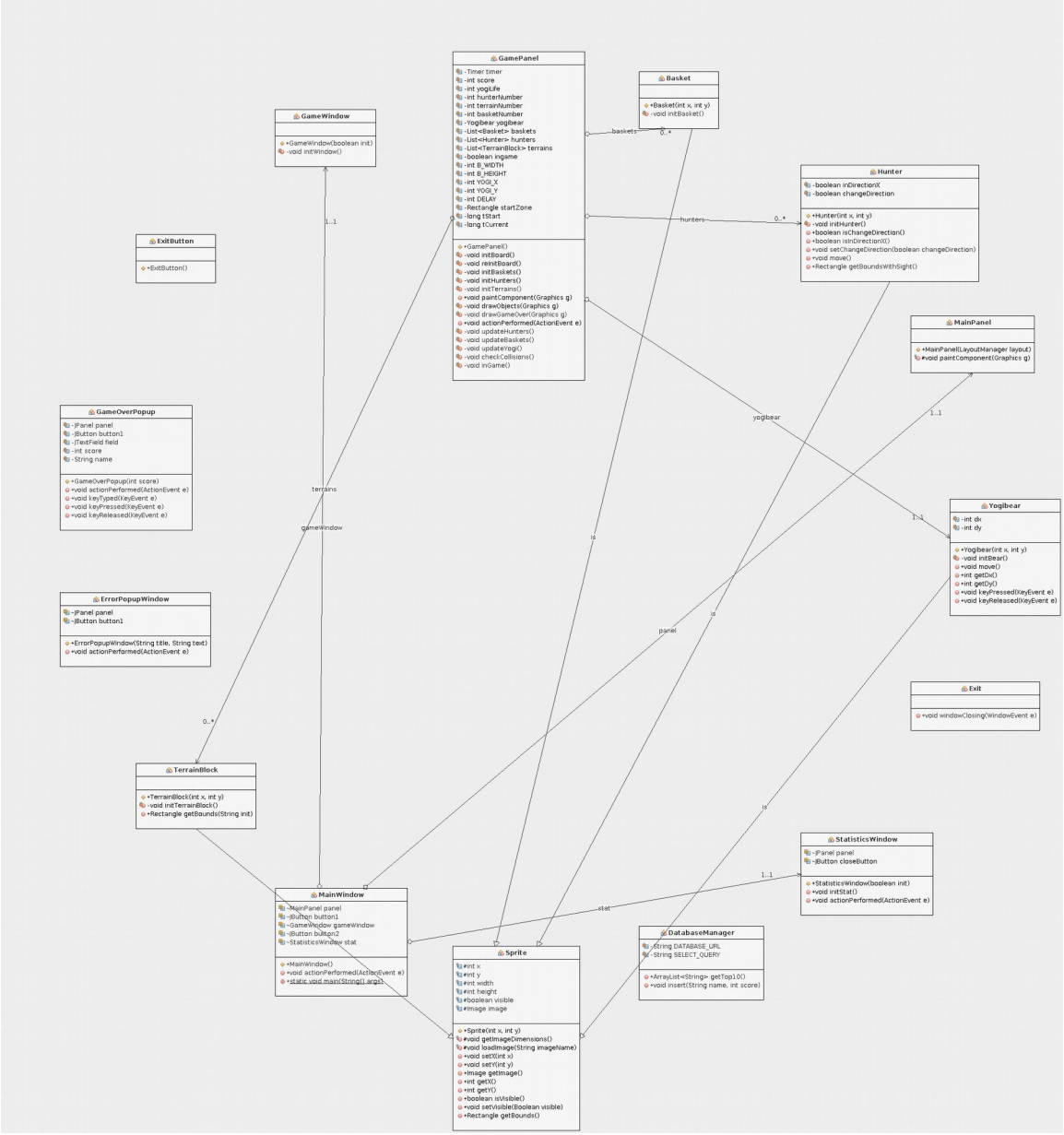
- getTop10(): Visszaadja az SQL adatbázisban található adatok közül a legnagyobb 10 pontszámot és az azokhoz tartozó játékos neveket.
- insert(String name, int score): A kapott értékeket hozzáadja az adatbázishoz.

Exit:

Az Exit osztály az ablak, illetve a program bezárásáért felel. Kiterjesztése a WindowAdapter osztálynak és csak egy metódusa van: windowClosing().

- windowClosing(): kilép a programból.

A program UML-je



Mellékelt lényeges fájlok

- A dist/javadoc könyvtárban megtalálható a NetBeans által generált HTML formátumú JavaDoc.
- Az src könyvtár tartalmazza a program forráskódját.

- A yogi_database könyvtár tartalmazza az SQL adatbázist, illetve a src/database könyvtár tartalmaz egy .sql scriptet, amely létrehozza a táblát néhány minta adattal.
- A projektmappa gyökerében található ez a dokumentáció több formátumban, illetve a NetBeansben easyUML pluginnal elkészített, osztálydiagramokat és azok kapcsolatait tartalmazó képet png formátumban.