Programozás technológia
3. beadandó feladat
Tron (4.)
Kübler Balázs
U5WGQF

Feladat

Készítsünk programot, amellyel a Tronból ismert fény-motor párbajt játszhatjuk felülnézetből. Két játékos játszik egymás ellen egy-egy olyan motorral, amely fénycsíkot húz maga mögött a képernyőn. A motor minden másodpercben a legutoljára beállított irányba halad feltéve, hogy a játékos nem változtatja meg azt egy megfelelő billentyű lenyomásával. (WASD az első játékos, nyilak a második játékos.)

Az a játékos veszít, aki előbb neki ütközik a másik játékos fénycsíkjának vagy a képernyő szélének. A játék elején kérjük el a játékosok nevét és engedjük meg, hogy maguk válasszák ki a fényük színét. A játék végekor a győztes játékos eredményét növeljük meg az adatbázisban. Ha a játékos még nem található meg az adatbázisban, úgy szúrjunk be egy új sort. Egy menüpontban legyen lehetőségünk a 10 legjobb eredménnyel rendelkező játékost megtekinteni, az elért pontszámukkal, továbbá lehessen bármikor új játékot indítani egy másik menüből.

Feladat elemzése

Egy felülnézetes, azaz 2 dimenziós játékot kell elkészíteni elemi grafika használatával.

3 ablakot kell majd létrehozni:

- egy kezdőablak, ahol a két játékos megadhatja a nevét és kiválaszthatja a fénye színét
- egy ablak (JPanel), amiben magát a játékot játszhatjuk
- egy dicsőség tábla, ahol a 10 legtöbb pontszámú játékos nevét és pontszámát tekinthetjük meg

A fent említett első két ablakhoz tartozni fog egy menü is, ahonnan a dicsőség táblát nyithatom meg és új játékot indíthatok, valamint kiléphetek.

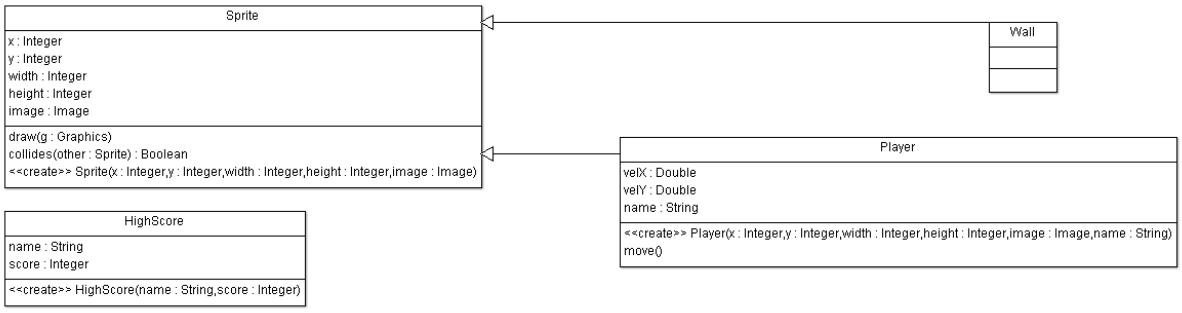
Irányítás: WASD és nyilak segítségével fog történni, mozgás pedig az idő függvényében fog alakulni. Plusz én implementálni fogok egy pause (P) gombot is.

A játék akkor ér véget, ha egy játékos nekiütközik egy másik játékos fénycsíkjának, vagy a képernyő szélének. Ezt kiegészítem azzal, hogy a képernyő szélére, és különböző pályákon egyéb helyekre is elhelyezek falakat, és ha ezek közül bármelyiknek neki megy az egyik játékos, akkor a másik fog nyerni.

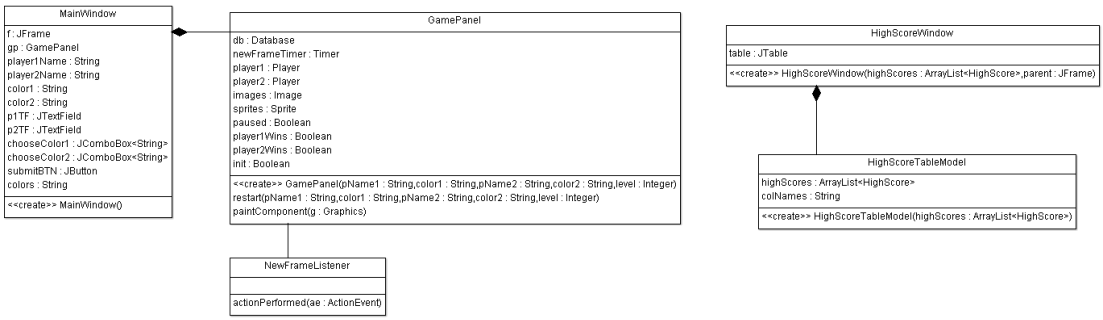
Az adatbázisomban két oszlop lesz, az egyikben a játékos nevét, a másikban a pontszámot fogom tárolni. A név nem lehet NULL, és ez lesz a kulcs. A pontszám lehetne nulla, de gyakorlatilag soha nem lesz az, hiszen csak a minimum 1 győzelemmel rendelkező játékosokat fogom beszúrni az adatbázisba.

A program szerkezete

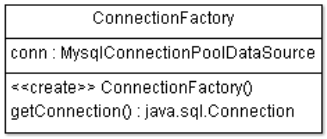
Package: model

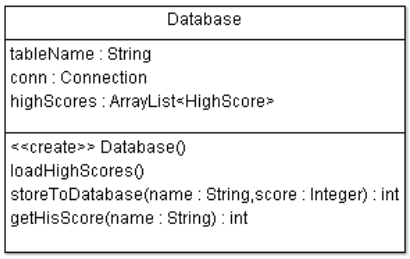


Package: view



Package: db





Erőforrás spórolás: restart metódus

A program megfelelő és kicsit energiatakarékosabb működéséhez egy újra inicializáló algoritmust készítettem. Ez ugyan olyan paraméterekkel dolgozik, mint a GamePanel konstruktora. Ez a függvény akkor kerül meghívásra, mikor a játék menüből új pályát szeretnék választani. Ekkor egy újabb konstruktorhívás, azaz újabb memóriafoglalás helyett hívom meg ezt a metódust, ami például az eddig használt ArrayList-eket újrahasznosíthatóvá teszi azzal, hogy kiüríti az elemeit és nem pedig egy teljesen újat hozok létre valahol a heap-en. Ezen kívül a játék vezérléséhez szükséges logikai változókat hamisra állítom (például paused). Timerből sem hozok létre újat, hanem az előző körben megállítottat indítom újra.

Esemény-eseménykezelő párok

Az irányításhoz tartozó eseménykezelőket összegyűjtöm a GamePanel osztályomon belül egy registerControlls nevű metódusban.

Irányításhoz tartozó események:

- W
- A
- S
- D
- felfelé nyíl
- jobbra nyíl

- lefelé nyíl
- balra nyíl
- P

betűk lenyomása. Ezekhez eseménykezelőt a getInputMap().put(„valami”, new AbstractAction()) és a getActionMap().put(„valami”, new AbstractAction()) metódusokkal társítok. Az AbstractAction()-ök actionPerformed(ActionEvent ae) metódusát felülírom úgy, hogy megfelelően mozogjanak, azaz:

W és felfelé nyíl esetén a Player Y irányú sebességét negálom,

A és balra nyíl esetén a Player X irányú sebességét negálom,

S és lefelé nyíl esetén a Player Y irányú sebességét pozitív értékre állítom.

D és jobbra nyíl esetén a Player X irányú sebességét pozitív értékre állítom.

A P betű lenyomásakor egy „paused” nevű logikai változó értékét állítom az ellenkezőjére.

Szintén az irányításhoz tartozik, de nem a billentyűzettel, hanem egy Timer-rel van összefüggésben a NewFrameListener osztály, ami gyakorlatilag azt csinálja, hogy egy előre definiált FPS számmal folyamatosan mozgatja a 2 játékost az utoljára beállított irányba, kivéve ha a játék le van állítva (lenyomták a P betűt). Ekkor mindig vizsgálom azt is, hogy a falnak vagy egy másik játékosnak nem-e ütközött neki az egyik játékos, mert ha igen, akkor a játék véget ért és beállítom a győztest.

További események: amikor a kezdőablakban (MainWindow) rákattintok az elküldés gombra, menüben pályát választok, vagy dicsőség táblát szeretném megnézni, valamint a kilépés gombra való kattintás is események. Ezeket szerintem fölösleges részletezni, egyedül talán az elküldés gombnál érdemes kiemelni azt, hogy amennyiben valaki nem választott színt, vagy nem adta meg a nevét a gombra ráírom a hibaüzenetet és nem engedem játszani a felhasználókat, amíg nem adnak meg valamit a névhez és nem választják ki a színüket.

Tesztelési terv

A tesztelés próbajátékokkal fog történni.

Szempontok:

- kezdőablak megjelenése, menü
- ki nem választott szín és/vagy meg nem adott név/nevek esetén ne engedjen játszani
- játékablak megjelenése
- játéktér, pálya betöltés, megjelenés
- játékos mozgása
- játék végének tesztelése
- adatbázisba való adatrögzítés tesztelése
- dicsőség tábla megjelenése, több adat esetén is csak az első 10 jelenik-e meg