1. Beadandó feladat dokumentáció

Készítette:
Ócsai Máté Zsigmond
E-mail: coqfuy@inf.elte.hu

12. Fénymotor párbaj

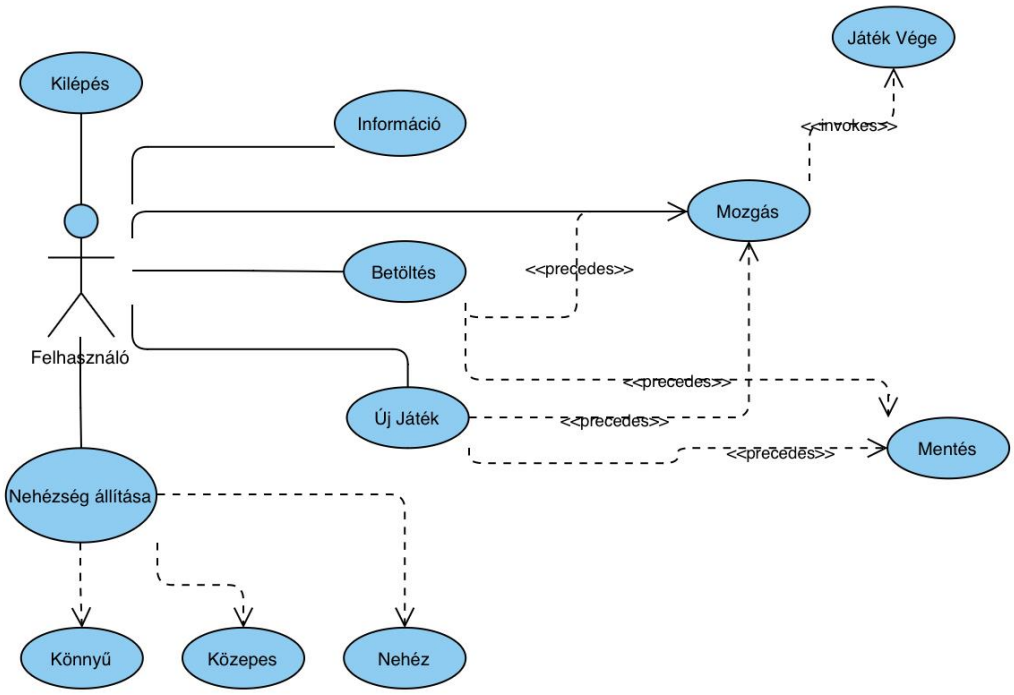
Készítsük programot, amellyel a Tronból ismert fénymotor párbajt játszhatjuk. Adott egy $n \times n$ elemből álló játékpálya. A két játékos a bal, illetve jobb oldal közepén indul egy-egy fénymotorral, amely egyenesen halad (rögzített időközönként) a legutoljára beállított irányba (függőlegesen, vagy vízszintesen). A motorokkal lehetőség van balra, illetve jobbra fordulni. A fénymotor mozgás közben fénycsíkot húz, ami a játék végéig ott marad. Az a játékos veszít, aki előbb nekiütközik a másik játékos motorjának, bármelyikük fénycsíkjának vagy a pálya szélének.
A program biztosítson lehetőséget új játék kezdésére a pályaméret megadásával (12×12, 24×24, 36×36), valamint játék szüneteltetésére (ekkor nem telik az idő, és nem mozognak a motorok). Ismerje fel, ha vége a játéknak, és jelenítse meg, melyik játékos győzött. Ezen felül szüneteltetés alatt legyen lehetőség a játék elmentésére, valamint betöltésére.

Elemzés:

- A játékot három nehézségi szinttel játszhatjuk: könnyű (12*12-es pálya méret), közepes (24*24-es pálya méret), nehéz (36*36-os pálya méret.). A program indításkor könnyű nehézséget állít be, tehát játék nehézségi szint beállítás nélkül elindítható a játék.
- A feladatot egyablakos asztali alkalmazásként Windows Forms grafikus felülettel valósítjuk meg.
- Az ablakban elhelyezünk egy menüt a következő menüpontokkal: Game (New Game, Resume Game, Finish Game, Save Game, Load Game, Quit Game), Difficulty (Easy (12*12 board), Medium (24*24 board), Hard (36*36 board)), Information of Game.
- A játéktáblát egy panel reprezentálja, amelyet nehézségi szinttől függően osztunk n*n méretre. A játék a New Game gomb lenyomására elindul, és megjelenik a pálya két szélén a kék, illetve piros játékos. Irányítani a fénymotorokat a w,a,s,d (kék) illetve i,j,k,l (piros) billentyűkkel tudjuk.
- A játék automatikusan feldob egy dialógusablakot, amikor vége a játéknak (valamelyik játékos neki megy a falnak, vagy saját vagy ellenfele testének), és döntetlen is kialakulhat. Szintén dialógusablakokkal

jelezzük, ha befejezünk egy játékot, és a mentés illetve betöltés esetén a felhasználó adja meg a file neveket.

- Ha játék közben a Game gombra nyomunk, szünetel a játék, tehát nem mozognak a motorok, és a Resume Game gombbal tudjuk folytatni.
- A felhasználói esetek az 1. ábrán láthatóak.



1. ábra: Felhasználói esetek diagramja

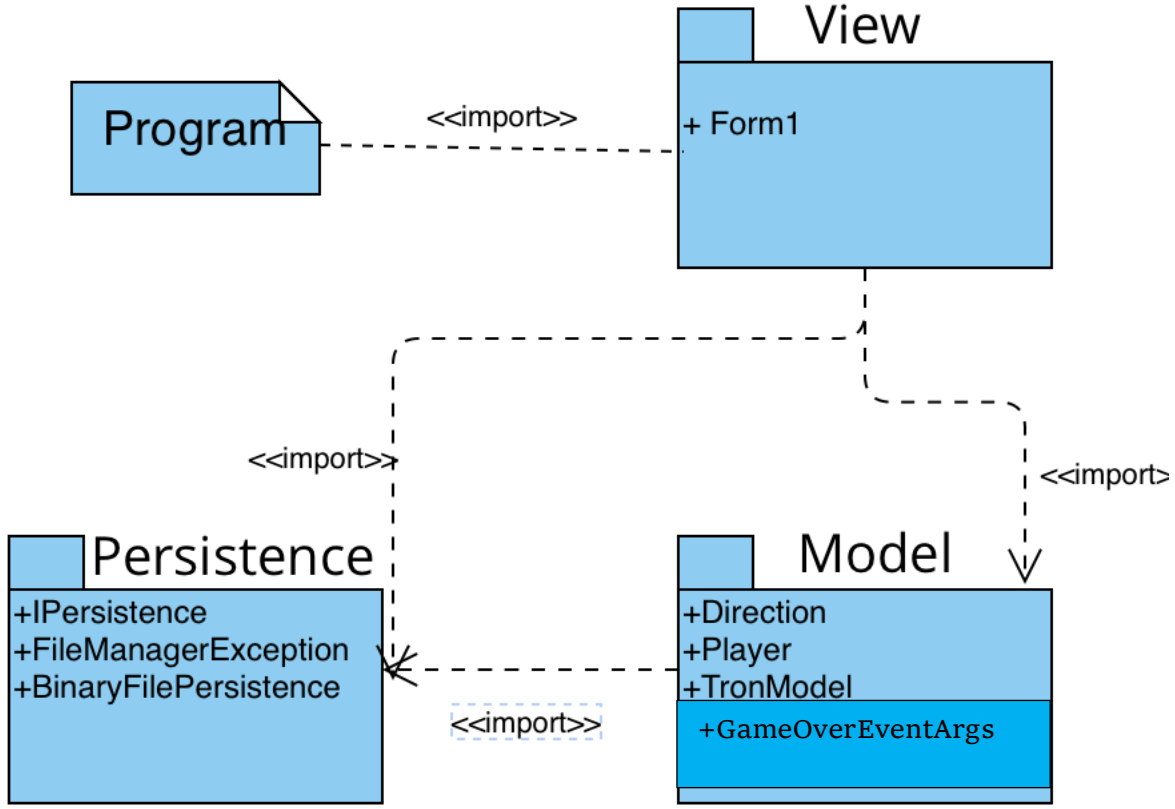
Tervezés:

• Programszerkezet:

- A programot háromrétegű architektúrában valósíjuk meg. A megjelenítés a View, a modell a Model, míg a perzisztencia a Persistence névtérben helyezkedik el. A program csomagszerkezete a 2. ábrán látható.
- A program szerkezetét két projektre osztjuk implementációs megfontolásból: a Persistence és Model csomagok a program felületfüggetlen projektjében, míg a View csomag a Windows Formstól függő projektjében kap helyet.

• Perzisztencia:

- Az adatkezelés feladata a Fénymotor párbaj táblával kapcsolatos információk tárolása, valamint a betöltés/mentés biztosítása.
- A hosszú távú adattárolás lehetőségeit az IPersistence interfész adja meg, amely lehetőséget ad a tábla betöltésére (Load), valamint mentésére (Save). A műveleteket hatékonysági okokból aszinkron módon valósítjuk meg.
- Az interfészt bináris fájl alapú adatkezelésre a BinaryFilePersistence osztály valósítja meg. A fájlkezelés során fellépő hibákat a FileManagerException kivétel jelzi.
- A program az adatokat bináris fájlként tudja eltárolni, melyek a .dat kiterjesztést kapják. Ezeket az adatokat a programban amikor nem megy egy aktív játék, bármikor be lehet tölteni, illetve ki lehet menteni az aktuális állást.
- A fájl egyetlen sora sorban tartalmazza a pálya mátrixát, pálya mérettől függően, amely tartalmazza, hogy melyik mező melyik játékos birtokában volt, a fájl vége pedig tartalmazza a két játékos utolsó pozícióját, és akkori irányukat.



2. ábra: Az alkalmazás csomagdiagramja

3.

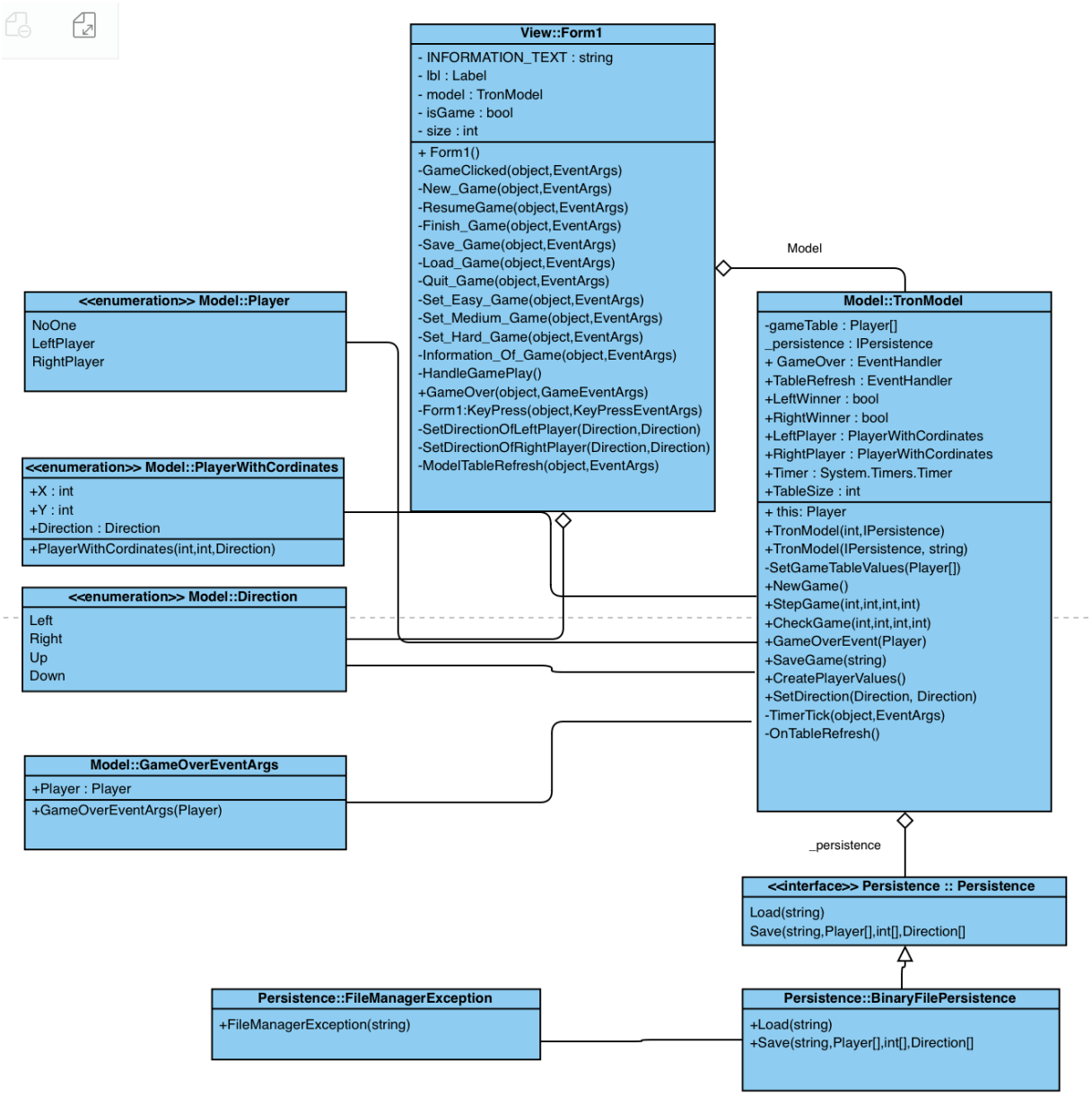
• Nézet:

- A nézetet a Form1 osztály biztosítja, amely tárolja a modell egy példányát (model), amely tartalmazza az adatelérést is.
- A játéktáblát egy dinamikusan létrehozott panel reprezentálja. A felületen létrehozzuk a megfelelő menüpontokat, illetve státuszsort, valamint dialógusablakokat, és a hozzájuk tartozó eseménykezelőket.
- A játék időbeli kezelését egy időzítő végzi (myTimer), amelyet mindig aktiválunk játék során, illetve inaktiválunk, amennyiben bizonyos menüfunkciók futnak.
- Az idő múlásával automatikusan mozognak a fénymotorok, amelyet tetszőleges időintervallumra állíthatunk be.

• Modell:

- A modell lényegi részét a TronModel osztály valósítja meg, amely szabályozza a tábla tevékenységeit, valamint a játék egyéb paramétereit. A típus lehetőséget ad új játék kezdésére (NewGame), valamint lépésre (StepGameLeft, StepGameRight).
- A modell példányosításkor megkapja az adatkezelés felületi méretét, amelynek segítségével lehetőséget ad betöltésre és mentésre (SaveGame).
- A játékosok típusát, és helyüknek elmetésére a PlayerWithCordinates osztáy ad lehetőséget, illetves található egy felsorolási típusú Player osztáy is, ami a pálya kirajzolásához segít.
- A betöltéshez, mentéshez, illetve a játék menetéhez ad segítséget a Direction osztály, ami szintén egy felsorolási típus, amiben az irányok találhatóak meg.
- A GameOverEventArgs arra szolgál, hogy a felületnek jelezni tudjuk, ha valamelyik játékos nyert.

A Program teljes statikus szerkezetét a 3. ábra mutatja be.



3. ábra: Az alkalmazás osztálydiagramja

Tesztelés:

- A modell funkcionalitása egységtesztek segítségével lett ellenőrizve a Test osztályban.
- Az alábbi tesztesetek kerültek megvalósításra:

- TronModelConstructorTest12,
- TronModelConstructorTest24,
- TronModelConstructorTest36: Új játék indítása, a pálya kirajzolása mérettől függően.
- CheckGameTest: Játékbeli lépés hatásainak ellenőrzése, játék közben.
- StepLeftTest: A játékban a balra lépések működése.
- StepRightTest: A játékban a jobbra lépések működése.
- StepUpTest: A játékban a felfele lépések működése.
- StepDownTest: A játékban a lefele lépések működése.
- TestInputString: A játék tesztelése random betöltéssel.
- EndGameLeftHitTheRightTest: Tesztelés, ha a bal játékos neki megy a jobb játékosnak, akkor vége.
- EndGameRightHitTheLeftTest: Tesztelés, ha a jobb játékos neki megy a bal játékosnak, akkor vége.
- EndGameLeftHitTheWallTest: Tesztelés, ha a bal játékos neki megy a falnak, akkor vége.
- EndGameRightHitTheWallTest: Tesztelés, ha a jobb játékos neki megy a falnak, akkor vége.
- EndGameTheyHitEachOtherTest: Tesztelés, ha a játékosok neki mennek egymásnak.