Készítette:

Név: **Kis Gergely Domonkos**

E-mail: vmt982@inf.elte.hu

Feladat:

Készítsünk programot, amellyel az alábbi két személyesjátékot játszhatjuk. Adott egy $n \times n$ pontból álló játéktábla, amelyen a játékosok két szomszédos pont között vonalakat húzhatnak (vízszintesen, vagy függőlegesen). A játék célja, hogy a játékosok a húzogatással négyzetet tudjanak rajzolni (azaz ők húzzák be a negyedik vonalat, független attól, hogy az eddigieket melyikük húzta).Ilyen módon egyszerre akár két négyzet is elkészülhet. A játék addig tart, amíg lehet húzni vonalat a táblán. A játékosok felváltva húzhatnak egy-egy vonalat, de ha egy játékos berajzolt egy négyzetet, akkor ismét ő következik. A program biztosítson lehetőséget új játék kezdésére a pályaméret megadásával (3×3, 5×5, 9×9), játék mentésére és betöltésére. Ismerje fel, ha vége a játéknak, és jelenítse meg, melyik játékos győzött (ha nem döntetlen). Játék közben a vonalakat, illetve a négyzeteket színezze a játékos színére.

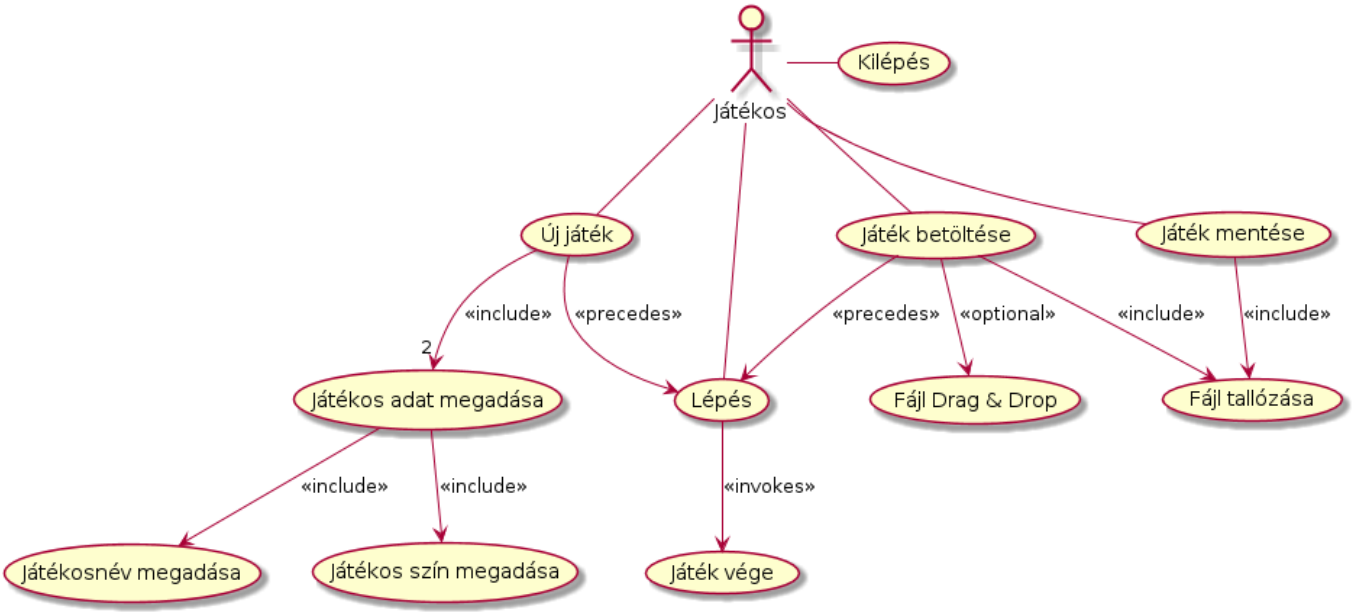
Elemzés

- A játékot 3 táblamérettel lehet játszani (3x3, 5x5, 7x7), melyet a „New Game” gomb megnyomása után feljövő ablakban lehet beállítani. A program indulása után nem indul játék egészen addig amíg a felhasználó nem adja meg az új játék indításához szükséges adatokat (játékosok adatai és táblaméret)
- A feladatot ablakos alkalmazással valósítjuk meg Windows Forms grafikus felülettel. Külön ablakot kap a játék és az új játék indítása.
- Az ablakban elhelyezünk 3 gombot („New Game”, „Save Game” és „Load Game”) feliratokkal, melyek az új játék indításáért, játék mentésért, és játék betöltésért felelősek.
- A játék indulását követően megjelenik 2-2 címke, amely a játékosok neveit és pontszámait jeleníti meg
- A játéktábla egy panel, amelynek a kirajzoló eseménye lett felülírva, illetve figyeljük a vele kapcsolatos egér eseményeket.

- Új vonalat úgy húzhatunk, ha rákattintunk egy kirajzolt pontra és az egér lenyomva tartása mellett ráhúzzuk egy másik (érvényes) pontra.

- A játék végét követően egy dialógusablak jelzi a győztes játékost, vagy a döntetlent, ez követően új játék indul ugyanazokkal a beállításokkal

- A felhasználói esetek az 1. ábrán láthatóak

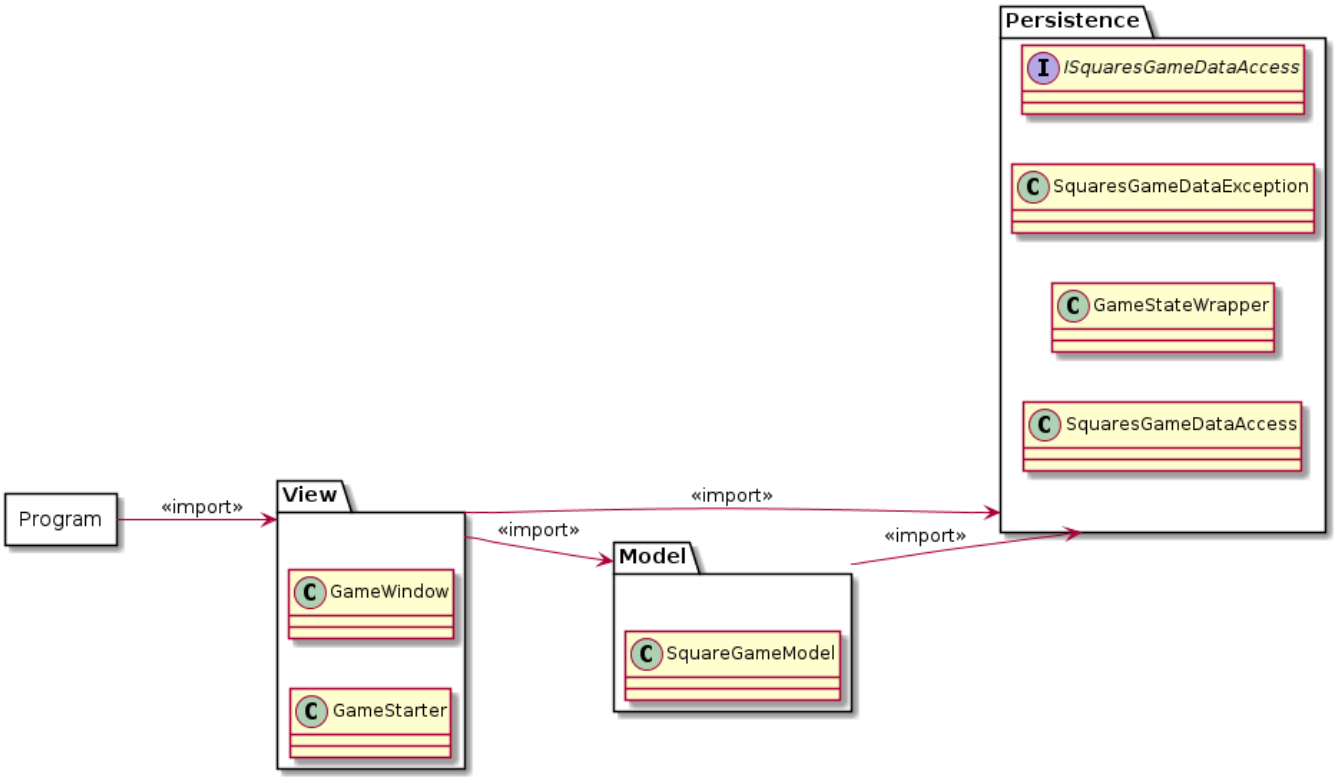


1. ábra

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.



2. ábra

Perzisztencia

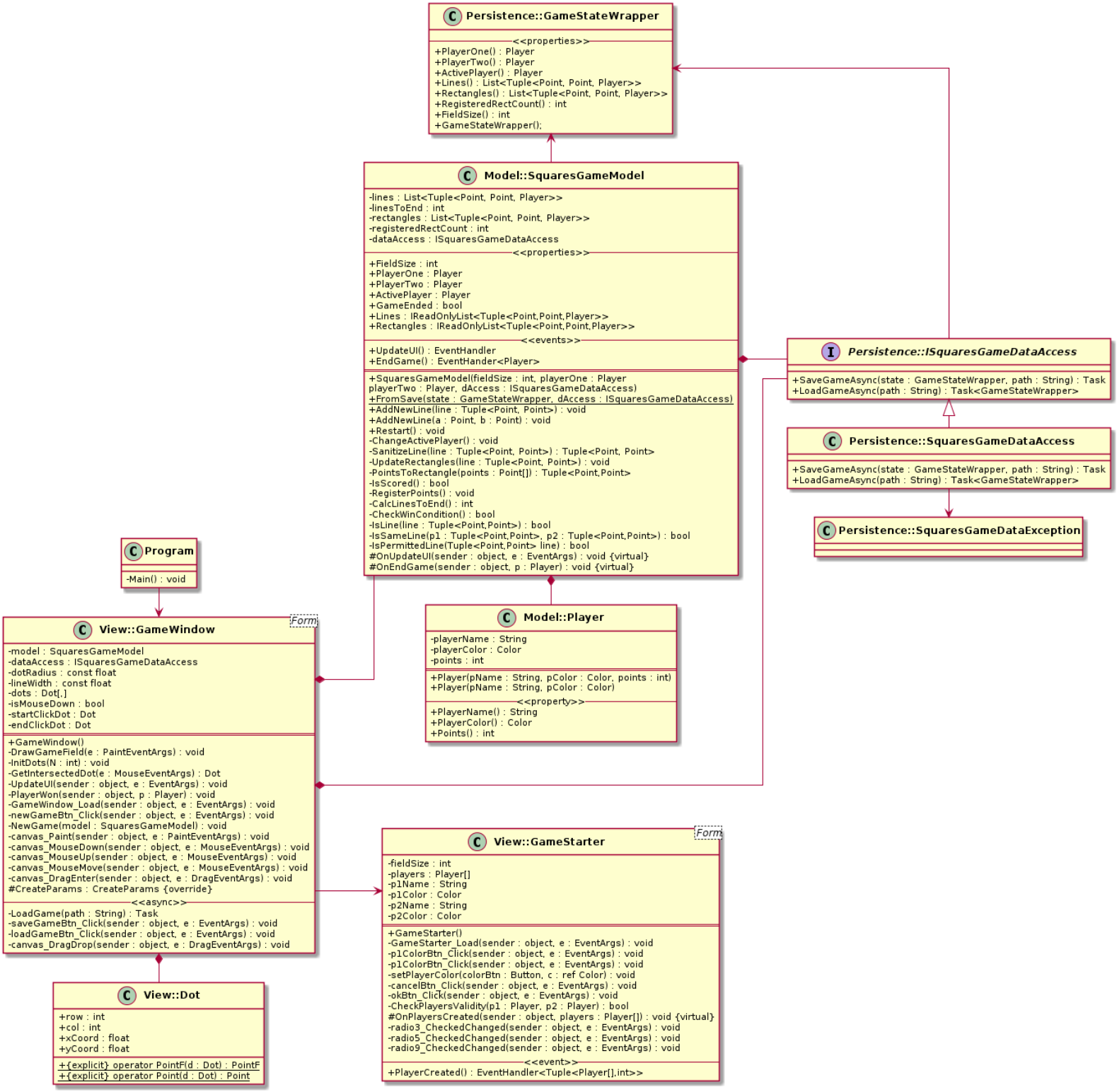
- Az adatkezelés feladata a játékállapot tárolása, valamint ennek a mentése, és az ebből való betöltés.
- A **GameStateWrapper** nevű osztály egy játékállapotot csomagoló osztály, amelyből a teljes játékállapot előállítható. Ez az alábbi adattagokat tartalmazza:
  - **PlayerOne** – Az első játékost leíró adattag, ez tárol további 3 mezőt (PlayerName, PlayerColor, Points) – (Játékosnév, Játékos szín, Pontok)
  - **PlayerTwo** – A második játékost leíró adattag hasonlóan az előzőhöz
  - **ActivePlayer** – Az aktív játékosra mutató referencia (egyike a PlayerOne-nak és a PlayerTwo-nak)
  - **Lines** – A behúzott vonalak végpontjait tároló lista
  - **Rectangles** – A kialakult négyzetek bal felső és jobb alsó végpontjait tároló lista
  - **RegisteredRectCount** – A kialakult négyzetek száma (összpontszám)
  - **FieldSize** – A tábla mérete
- A hosszútávú adattárolás lehetőségeit az **ISquaresGameDataAccess** interfész adja meg, amely lehetőséget ad a játékállapot betöltésére (*LoadAsync*), valamint mentésére (*SaveAsync*). A műveleteket hatékonysági okokból aszinkron módon valósítjuk meg.
- Azinterfészt szöveges fájl alapú adatkezelésre a **SquaresGameDataAccess** osztály valósítja meg. A fájlkezelés során fellépő hibákat a **SquaresGameDataException** kivétel jelzi.
- A program az adatokat szöveges fájlként tudja eltárolni. Ezeket az adatokat a programban bármikor be lehet tölteni, illetve ki lehet menteni az aktuális állást.
- A fájl:
  - 1-2. sora megadja a két játékos adatait (**Játékosnév**, **Játékos szín** ARGB kódja, **pontszám**)
  - 3. sora megadja az **aktív játékos indexét**, és a **táblaméretet**
  - 4. sora megadja a **behúzott vonalak számát** (N)
  - A következő N sor egy-egy **behúzott vonal adatait tartalmazza** (2 pont sor és oszlop indexeit, valamint egy játékosindexet, ami jelzi melyik játékos húzta be)
  - Az ezt követő sor a **kialakult négyzetek számát** jelöli (M)
  - A következő M sor egy-egy négyzet bal felső és jobb alsó csúcspontjának sor és oszlop indexeit, valamint egy játékosindexet, ami jelzi melyik játékos húzta be az utolsó vonalat.

Modell

- A modell lényegi részét a **SquaresGameModel** osztály valósítja meg, amely kezeli a játék belső állapotát (Játékosok pontszámai, játék vége, behúzott vonalak, behúzott négyzetek).
- Lehetőséget ad a játék újraindítására (**Restart**) az előző beállításokkal, játékállapot mentésére és betöltésére
- A fő játékmetódus a (**AddNewLine**) amely a játéktér 2 pontját regisztrálja behúzott vonalként, amennyiben ezek megfelelőek
- Az **UpdateUI** esemény kiváltódik amennyiben van újonnan regisztált vonal, ez jelzi a nézet felé, hogy a táblát újra kell rajzolni az új állapottal
- Az **EndGame** esemény kiváltódik amennyiben az összes lehetséges vonal be lett húzva, átadva a győztes játékosra mutató referenciát, vagy null-t, ha döntetlen.

Nézet

- A nézetet a **GameWindow** és a **GameStarter** osztályok biztosítják, a GameWindow tárolja a modell egy példányát, valamint az adatelérés egy konkrét példányát. A GameStarter csupán egy segédosztály, ami egy modell objektum példányosítását segíti a játékbeállítások felhasználótól való bekérésével.
- A játéktábla egy panel felületére van rajzolva (**canvas**) , amelynek rajzoló eseménye van lekezelve egy **DrawGameField** nevű metódussal, amely a modell példányosításakor van hozzárendelve
- A kirajzolni kívánt pontok egy **dönts** nevű **Dot**[] típusú tömbben vannak eltárolva, ez segít azonosítani egy pontot a sor és oszlop indexével, illetve a képernyőn a középpontjának x és y koordinátájával
- A játék **UpdateUI** eseményét a nézet egy megegyező nevű **UpdateUI** metódusa kezeli le, ez újra rajzolja a panelt és frissíti a pontszámokat megjelenítő címkéket.
- A játék **PlayerWon** eseményét a nézet egy megegyező nevű **PlayerWon** metódusa kezeli le, ez egy dialógus ablakot nyit, amely tartalmazza a győztest, ha van, illetve a döntetlent, ha nincs.
- A nézet további eseménykezelői csak az egérgomb eseményeket kezelik le, ezek működései a metódusnevekből egyértelműen megérthetők.



3. ábra Az alkalmazás osztálydiagramja

Tesztelés

- A modell funkcionalitása egységtesztek segítségével lett ellenőrizve a **SquaresGameTest** osztályban

- Az alábbi tesztesetek kerültek megvalósításra
  - **Helyes** vonalak behúzása
    - Bal-jobb irány horizontális
    - Jobb-bal irány horizontális
    - Fel-le irány vertikális
    - Le-fel irány vertikális
  - **Helyes** vonalak **többszöri behúzása nem történik meg**
  - **Helytelen** vonalak behúzása nem történik meg
    - Átlós vonal
    - Horizontális – egy egységnél hosszabb
    - Vertikális – egy egységnél hosszabb
    - Nem vonal – Két azonos pont
    - Játéktéren kívül eső
  - **Helyes** egyszeres négyzet behúzása
  - **Helyes** kétszeres nézet behúzása
  - **Helyes** játékmenet szimuláció