**Projekt dokumentáció**

**Készítette:**

Kerényi Gergő

**E-mail:** gergokerenyi02@gmail.com

**Feladat:**

Készítsünk programot, amellyel a következő két személyes játékot lehet játszani. Adott egy 𝑛 × 𝑛 mezőből álló tábla, ahol egy menekülő és egy támadó játékos helyezkedik el. Kezdetben a menekülő játékos figurája középen van, míg a támadó figurái a négy sarokban helyezkednek el. A játékosok felváltva lépnek. A figurák vízszintesen, illetve függőlegesen mozoghatnak 1-1 mezőt, de egymásra nem léphetnek. A támadó játékos célja, hogy adott lépésszámon (4𝑛) belül bekerítse a menekülő figurát, azaz a menekülő ne tudjon lépni. A program biztosítson lehetőséget új játék kezdésére a táblaméret (3 × 3, 5 × 5, 7 × 7) és így a lépésszám (12, 20, 28) megadásával, folyamatosan jelenítse meg a lépések számát, és ismerje fel, ha vége a játéknak. Ekkor jelenítse meg, melyik játékos győzött, majd kezdjen automatikusan új játékot. Ezen felül legyen lehetőség a játék elmentésére, valamint betöltésére

**Elemzés:**

• A játékot három pályaméreten lehet játszani: 3x3, 5x5, 7x7

A program inditásakor lehetőség van a pályaméret kiválasztására, vagy játék betöltésére.

• A feladatot egyablakos asztali alkalmazásként Windows Forms grafikus felülettel

valósítjuk meg.

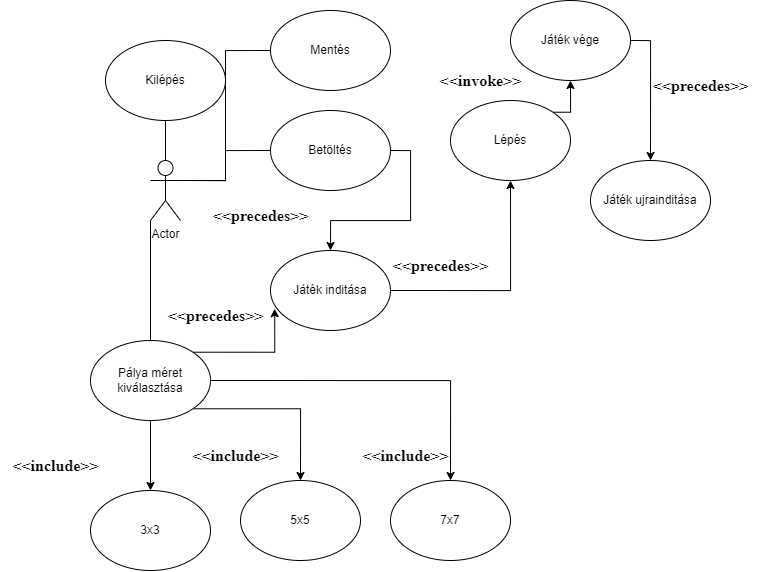
• Az ablakban elhelyezünk egy menüt a következő menüpontokkal: File (Játék betöltése, Játék mentése). 3 gombot a pálya méretének kiválasztásához, illetve egy játék inditása/ujrainditása gombot.

Az ablak alján megjelenítünk egy státuszsort, amely a hátralévő lépésszámot és a soron következő játékost mutatják.

• A játéktáblán a 2-es (kék) játékos TAB-al tud a 4 karaktere közül választani, majd a nyilakkal szabad mezőre léphet.

• A játék automatikusan feldob egy dialógusablakot, amikor vége a játéknak (a kék játékos körbekeritette a piros játékost, vagy a kék játékosnak elfogytak a lépésszámai (4\* a pálya mérete)). Szintén dialógusablakokkal végezzük el a mentést, illetve betöltést, a fájlneveket a felhasználó adja meg.

• A felhasználói esetek az 1. ábrán láthatóak.



1. ábra

**Tervezés:**

• Programszerkezet:

• A programot MVVM architektúrában valósítjuk meg, ennek megfelelően View, Model, ViewModel, Persistence névtereket valósitunk meg az alkalmazáson belül. A program környezetét az alkalmazás osztály (App) végzi, amely példányositja a modellt, a nézetmodellt és a nézetet, biztositja a kommunikációt, valamint felügyeli az adatkezelést.

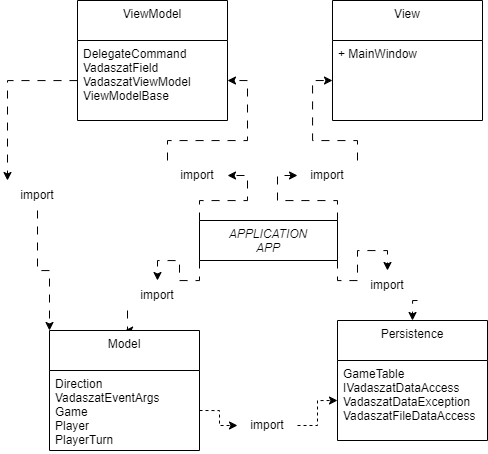
A program csomagszerkezete a 2. ábrán látható.

• A program szerkezetét két projektre osztjuk implementációs

megfontolásból: a Persistence és Model csomagok a program

felületfüggetlen projektjében, míg a ViewModel és View csomagok a WPF függő projektjében kapnak helyet.

1. ábra: Az alkalmazás csomagdiagramja



• Perzisztencia (3.ábra):

• Az adatkezelés feladata a Vadászat táblával kapcsolatos információk

tárolása, valamint a betöltés/mentés biztosítása.

• A GameTable osztály egy érvényes Vadászat táblát biztosít.

A tábla lehetőséget ad a játékosok kezdő helyzetének lerakására, játékos mozgatására, annak helyességének ellenőrzésére, a lépések számának számontartására, illetve a jelenlegi (currentPlayer) játékos váltására.

• A hosszú távú adattárolás lehetőségeit az IVadaszatDataAccess interfész

adja meg, amely lehetőséget ad a tábla betöltésére (LoadAsync), valamint

mentésére (SaveAsync). A műveleteket hatékonysági okokból aszinkron

módon valósítjuk meg.

• Az interfészt szöveges fájl alapú adatkezelésre a VadaszatFileDataAccess

osztály valósítja meg. A fájlkezelés során fellépő hibákat a

VadaszatDataException kivétel jelzi.

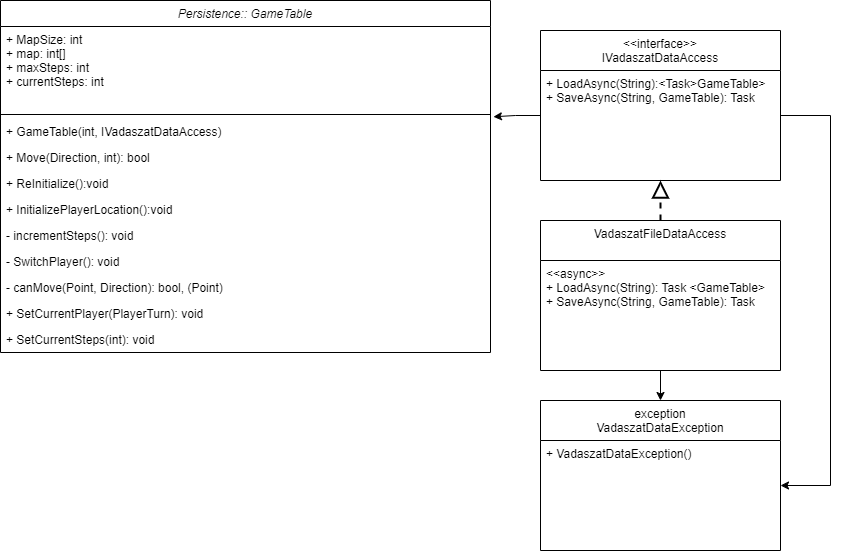
• A program az adatokat szöveges fájlként tudja eltárolni, melyek az txt

kiterjesztést kapják. Ezeket az adatokat a programban bármikor be lehet

tölteni, illetve ki lehet menteni az aktuális állást.

• A fájl első sora megadja a tábla méretét (3/5/7) a hátralévő lépések számát és a soron lévő játékost.

A fájl többi része izomorf leképezése a játéktáblának. 0 szabad mező, 1-es az 1-es játékos, 2-es pedig a 2-es játékost jelöli.



1. ábra

• Modell (4.ábra):

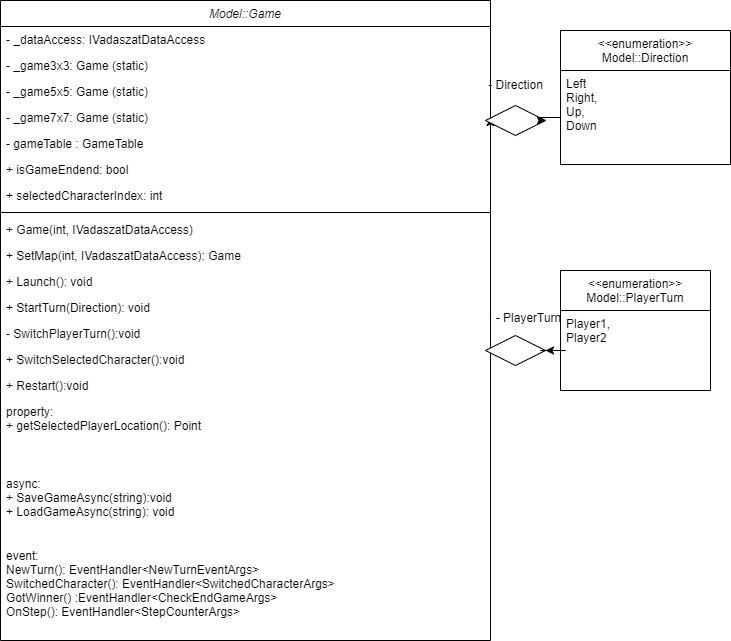
• A modell lényegi részét a Game osztály valósítja meg, amely szabályozza a tábla tevékenységeit, valamint a játék egyéb paramétereit, úgymint a 2-es játékos éppen kijelölt játékosa, a soron lévő játékos. A típus lehetőséget ad új játék kezdésére (SetMap), valamint lépésre (Move), egy kör lefolytatására (StartTurn) játékos váltására (SwitchPlayerTurn), illetve a nyertes ellenőrzésére (checkWinner).

• A játékállapot megváltozásáról (következő játékos, kijelölt játékos (2-es játékos esetén)) a NewTurn esemény, míg a játék végéről a GotWinner esemény

tájékoztat. Az esemény argumentuma (CheckEndGameArgs) tárolja a győzelem állapotát, illetve a nyertest, amennyiben van.

• A modell példányosításkor megkapja az adatkezelés felületét, amelynek segítségével lehetőséget ad betöltésre (LoadGameAsync) és mentésre (SaveGameAsync)

1. ábra

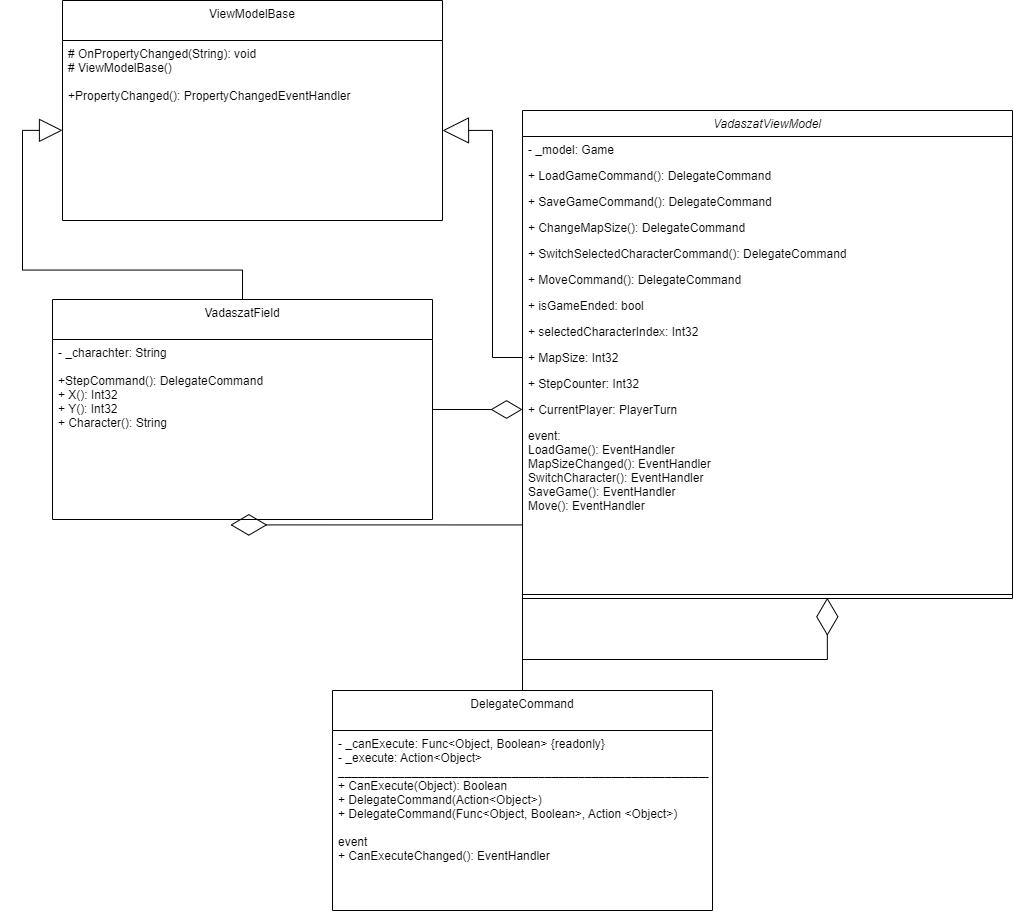


• Nézetmodell (5.ábra) :

• A nézetmodell megvalósításához felhasználunk egy általános utasítás (DelegateCommand), valamint egy ős változásjelző (ViewModelBase) osztályt.

• A nézetmodell feladatait a VadaszatViewModel osztály látja el, amely parancsokat biztosít az új játék kezdéséhez, játék betöltéséhez, mentéséhez, valamint a kilépéshez. A parancsokhoz eseményeket kötünk, amelyek a parancs lefutását jelzik a vezérlőnek. A nézetmodell tárolja a modell egy hivatkozását (\_model), de csupán információkat kér le tőle, illetve a pályaméretet szabályozza. Direkt nem avatkozik a játék futtatásába.

A játékmező számára egy külön mezőt biztosítunk (VadaszatField), amely eltárolja a mezőtipust (0/1/2) , valamint a lépés parancsát (StepCommand). A mezőket egy felügyelt gyűjteménybe helyezzük a nézetmodellbe (Fields)

5. ábra

• Nézet:

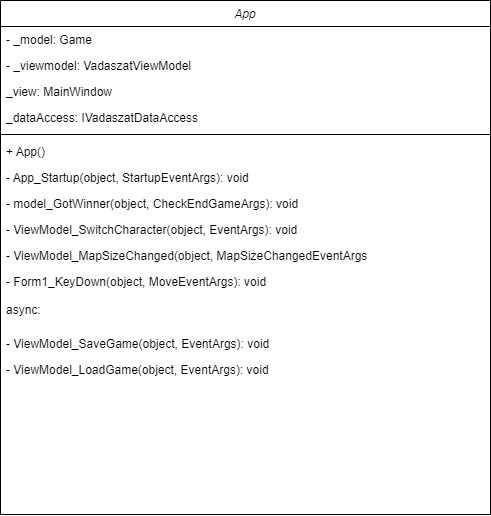
• A nézet csak egy képernyőt tartalmaz, a MainWindow osztályt. A nézet egy rácsban tárolja a játékmezőt, a menüt és a státuszsort. A játékmező egy ItemsControl vezérlő, ahol dinamikusan felépítünk egy rácsot (UniformGrid), amely gombokból áll.

Minden adatot adatkötéssel kapcsolunk a felülethez, továbbá azon keresztül szabályozzuk a gombok színét is. ICommand DelegateCommand - \_canExecute: Func {readOnly} - \_execute: Action

• A fájlnév bekérését betöltéskor és mentéskor, valamint a figyelmeztető üzenetek megjelenését beépített dialógusablakok segítségével végezzük.

Környezet (6. ábra):

• Az App osztály feladata az egyes rétegek példányosítása (App\_Startup), összekötése, a nézetmodell, valamint a modell eseményeinek lekezelése, és ezáltal a játék, az adatkezelés, valamint a nézetek szabályozása.



**Tesztelés:**

• A modell funkcionalitása egységtesztek segítségével lett ellenőrizve a VadaszatTest osztályban.

• Az alábbi tesztesetek kerültek megvalósításra:

• TestStartOfANewGame,

TestSwitchSelectedCharacter,

TestStartTurn: Új játék indítása, 2-es játékos kijelöltjének ellenőrzése, kezdeti értékek ellenőrzése, 1. lépés megtétele.

• TestPlayerPosition: Játékosok helyzetének ellenőrzése.

• TestRestart: Játék újrainditásának ellenőrzése

• TestWinner: A játék nyertesének ellenőrzése (ha van), mockolt perzisztencia réteggel.

• TestFalseMove: hibás lépés tesztelése, mockolt perzisztencia réteggel.