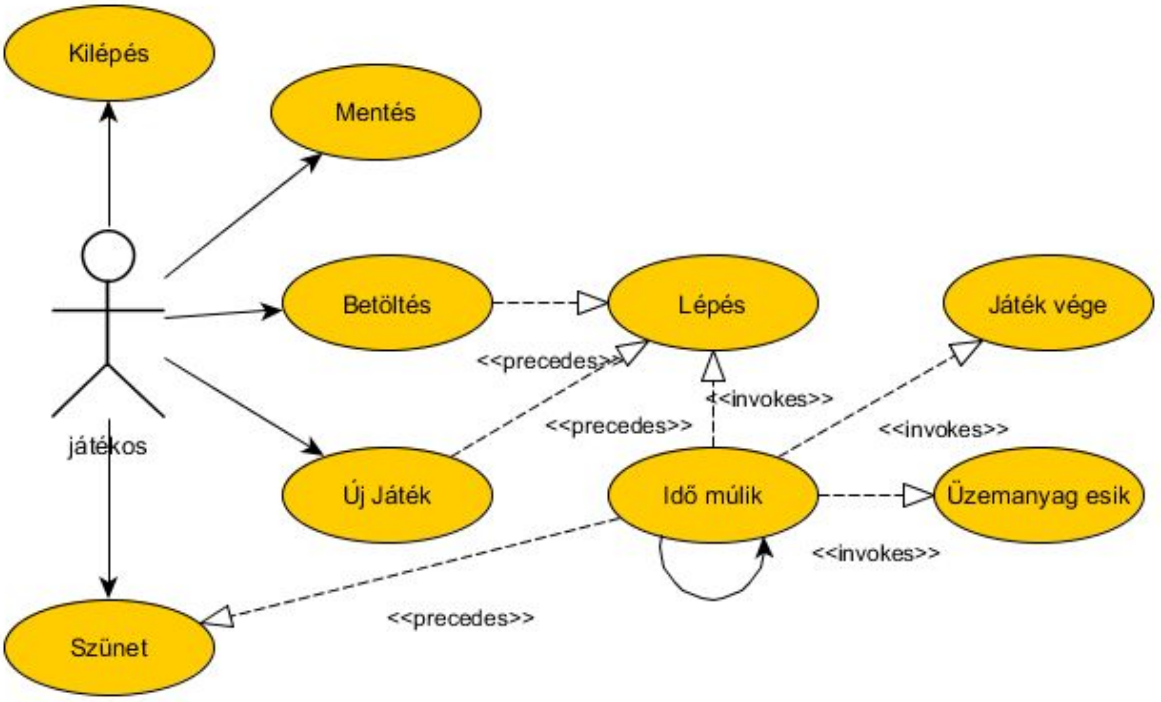
2. beadandó feladat dokumentáció

Készítette:
Szalóki Sándor
H8L59S
setr@inf.elte.hu

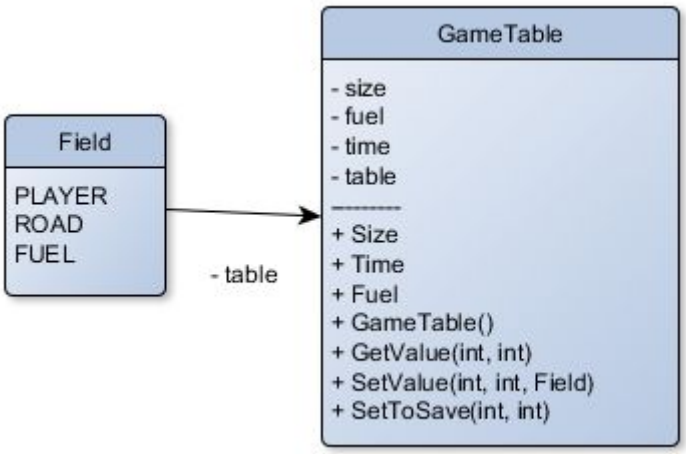
Feladat:
Készítsünk programot, amellyel az alábbi motoros játékot játszhatjuk. A feladatunk, hogy egy gyorsuló motorral minél tovább tudjunk haladni. A gyorsuláshoz a motor üzemanyagot fogyaszt, egyre többet. Adott egy kezdeti mennyiség, amelyet a játék során üzemanyagcellák felvételével tudunk növelni. A motorral a képernyő alsó sorában tudunk balra, illetve jobbra navigálni. A képernyő felső sorában meghatározott időközönként véletlenszerű pozícióban jelennek meg üzemanyagcellák, amelyek folyamatosan közelednek a képernyő alja felé. Mivel a motor gyorsul, ezért a cellák egyre gyorsabban fognak közeledni, és mivel a motor oldalazó sebessége nem változik, idővel egyre nehezebb lesz felvenni őket, így egyszer biztosan kifogyunk üzemanyagból. A játék célja az, hogy a kifogyás minél később következzen be. A program biztosítson lehetőséget új játék kezdésére, valamint játék szüneteltetésére (ekkor nem telik az idő, és nem mozog semmi a játékban). Ismerje fel, ha vége a játéknak, és jelenítse meg, mennyi volt a játékidő.

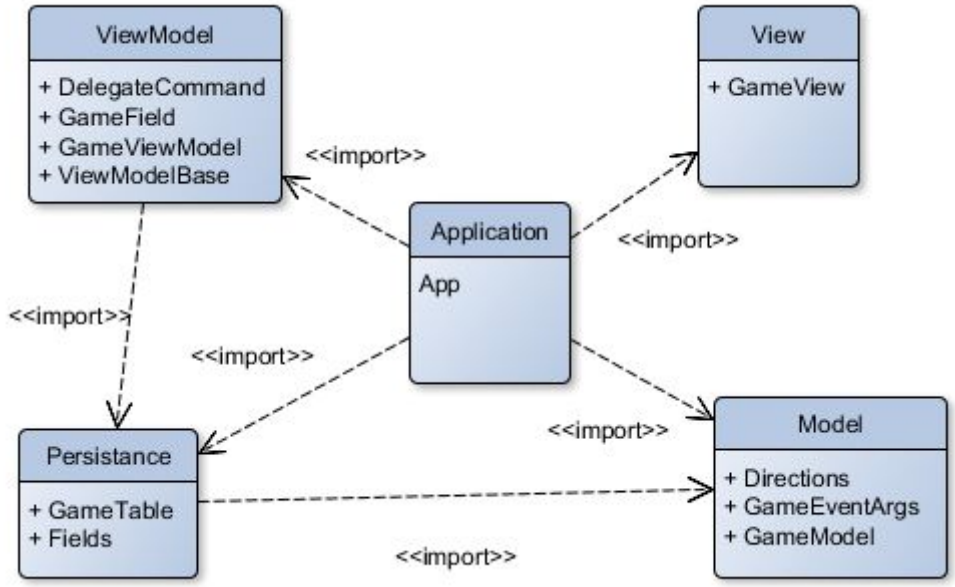
Elemzés:
- A játék nehézségét nem állíthatjuk, illetve alapértelmezett beállításokon kívül más opció nincs a játék inditísához. A játék 20 üzemanyaggal indul, illetve a pályán az új üzemanyagok 3 másodpercenként jelennek meg, majd 1 másodpercenként mozognak lefelé. Ez a játék folyamán gyorsul.
- A feladatot egyablakos asztali alkalmazásként WPF grafikus felülettel valósítjuk meg.
- Az ablakban található egy “File” menü, ahol található az “Új játék”, “Mentés”, “Betöltés”, illetve a “Kilépés”. Emellett található a “Szünet”. Az ablak közepén egy panelra rajzoljuk a játékot. A status bar-ban található az idő és az üzemanyag mennyiség.
- Játék végekor a játék automatikusan feldob egy dialógusablakot, ahol kiírja az elért időt. A játék nem nyerhető meg.



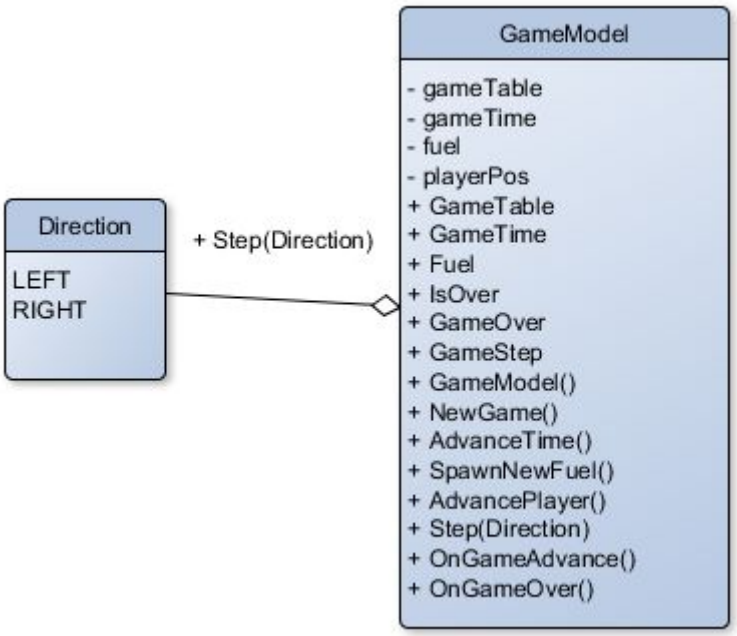
Tervezés:

- Programszerkezet:
  - A programot MVVM architektúrában valósítjuk meg, ennek megfelelően View, Model, ViewModel és Persistence névtereket valósítunk meg az alkalmazáson belül. A program környezetét az alkalmazás osztály (App) végzi, amely példányosítja a modellt, a nézetmodell és a nézetet, biztosítja a kommunikációt, valamint felügyeli az adatkezelést
- Perzisztencia:
  - Az adatkezelés feladata a pályával kapcsolatok információk tárolása, valamint a betöltés/mentés biztosítása.
  - A GameTable osztály tárolja az aktiális állását a játéknak. Ez egy 10x10-es tábla, amin 1 játékos van, és bármennyi üzemanyag cella. A tábla adatai lekérhetőek a megfelelő Getterekkel.
  - A hosszú távú adattárolás lehetőségeit az IDataAccess 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 FileDataAccess osztály valósítja meg. A program az adatokat szöveges fájlként tudja eltárolni.
  - Ezeket az adatokat a programban bármikor be lehet tölteni, illetve ki lehet menteni az aktuális állást.

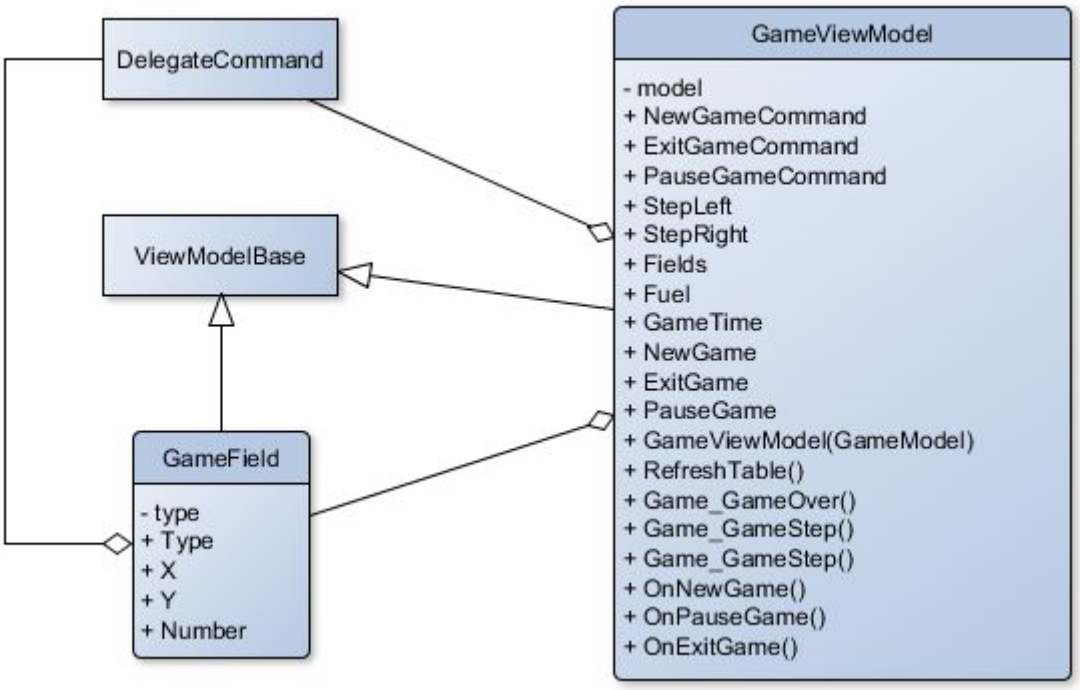




- Modell:
    - A modellt a GameModel osztály valósítja meg, amely szabályozza, hogy a GameTable elemen mi történik, illetve tárolja az aktuális játékidőt, üzemanyag mennyiséget.
    - Lehetőséget ad új játék kezdésére, az idő léptetésére, új üzemanyag létrehozására, illetve a játékos jobbra, illetve balra mozgatására.
    - A játékállapot változásáról a GameStep esemény, a játék végéről a GameOver esemény tájékoztat.
    - A modell példányosításakor megkapja az adatkezelés felületét, amelynek segítségével lehetőséget ad betöltésre és mentésre



- Nézetmodell:
    - A nézetmodell megvalósításához felhasználunk egy általános utasítás (DelegateCommand), valamint egy ős változásjelző (ViewModelBase) osztályt.
    - A nézetmodell feladatait a GameViewModel 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 játéknehézséget szabályozza. Direkt nem avatkozik a játék futtatásába.
    - A játékmező számára egy külön mezőt biztosítunk (GameField), amely eltárolja a pozíciót, színt. A mezőket egy felügyelt gyűjteménybe helyezzük a nézetmodellbe (Fields).



- 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.

- Környezet:
    - 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.
    - A játékban az időt három Timer segítségével mérjük, hiszen szükséges egy timer a valós idő számításához, egy az új üzemanyagok közti idő mérésére, illetve egy, amitől a játék nehezedik, az üzemanyagok esési sebességéhez.

Tesztelés:

- A modell funkcionális egységtesztek segítségével lett ellenőrizve a UnitTest1 osztályban.
- Az alábbi tesztek kerültek megvalósításra:
    - Új játék kezdete: Üzemanyagmennyiség helyes, az idő nulla és létezik játékos a pályán
    - Új üzemanyag: Új üzemanyag hozzáadása ténylegesen megtörténik
    - Játékos mozgása: A játékos képes jobbra, illetve balra mozogni.
    - Játék vége: Ha az üzemanyag eléri a nullát, a játéknak vége van.