1.Beadandó feladat

Készítette:

Csóka Máté
E-mail: gt9bq9@inf.elte.hu

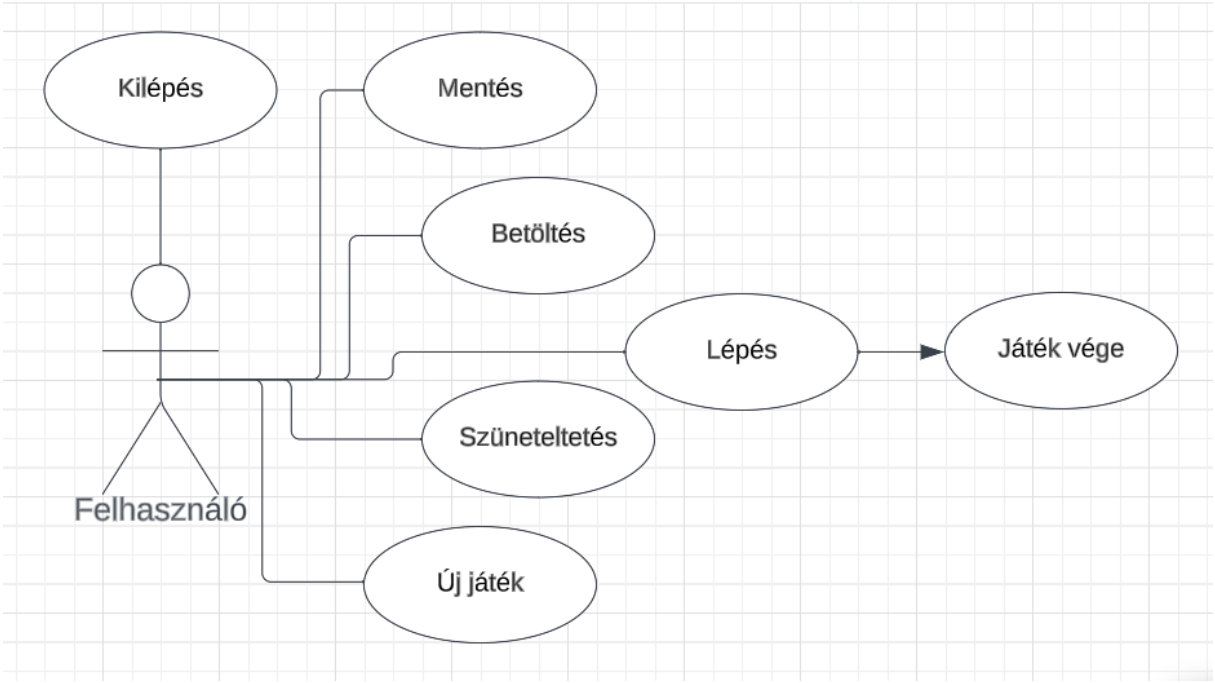
Feladat:

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. Ezen felül szüneteltetés alatt legyen lehetőség a játék elmentésére, valamint betöltésére.

Elemzés:

- A program indításkor automatikusan új játékot indít, szüneteltetett állapotban.
- A feladatot egyablakos asztali alkalmazásként Windows Forms grafikus felülettel valósítjuk meg.
- Az ablakban elhelyezünk egy menüt a következő menüpontokkal: New Game, Start, Pause, Save, Load. Az ablak tetején elhelyezünk 2 labelt, ami a játékosok üzemanyagszintjét és idejét jelzi.
- A játéktáblát egy 13x13 gombokból álló rács reprezentálja. A rács alsó sorában van a motor (feketével jelzett mező), ezt irányíthatja a játékos a nyilak megnyomásával. A rács felső sorában véletlenszerűen jelennek meg üzemanyagcellák (pirossal jelzett mező), egyszerre csak egy és erre is csak 30% esély van. Az üzemanyagcellák egyre jobban gyorsulva esnek lefelé. Az üzemanyagszint folyamatosan csökken, az idő múlásával egyre gyorsabban. Ha az üzemanyagcella, érintkezik a motor cellával, akkor az üzemanyagszinthez, +5 egység adódik. Maximum üzemanyagszint 50 egység lehet.
- A játék automatikusan kidob egy dialógusablakot, amikor vége a játéknak (elfogyott az üzemanyag). Szintén dialógusablakkal végezzük el a mentés, betöltést.

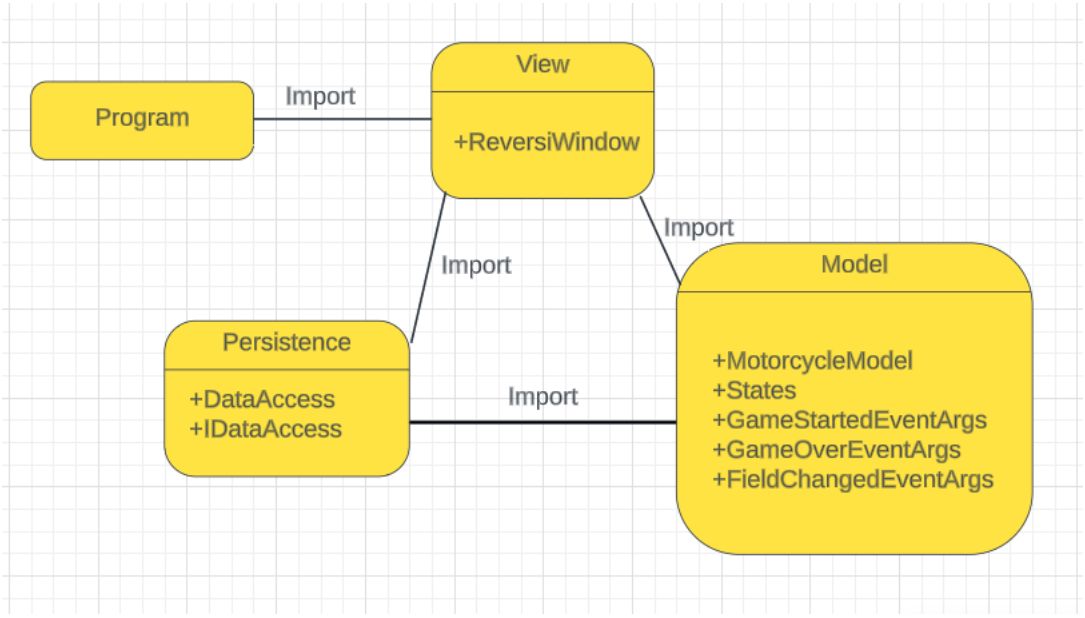
- A felhasználói esetek az 1. ábrán láthatóak.



1.ábra: Felhasználói esetek diagramja

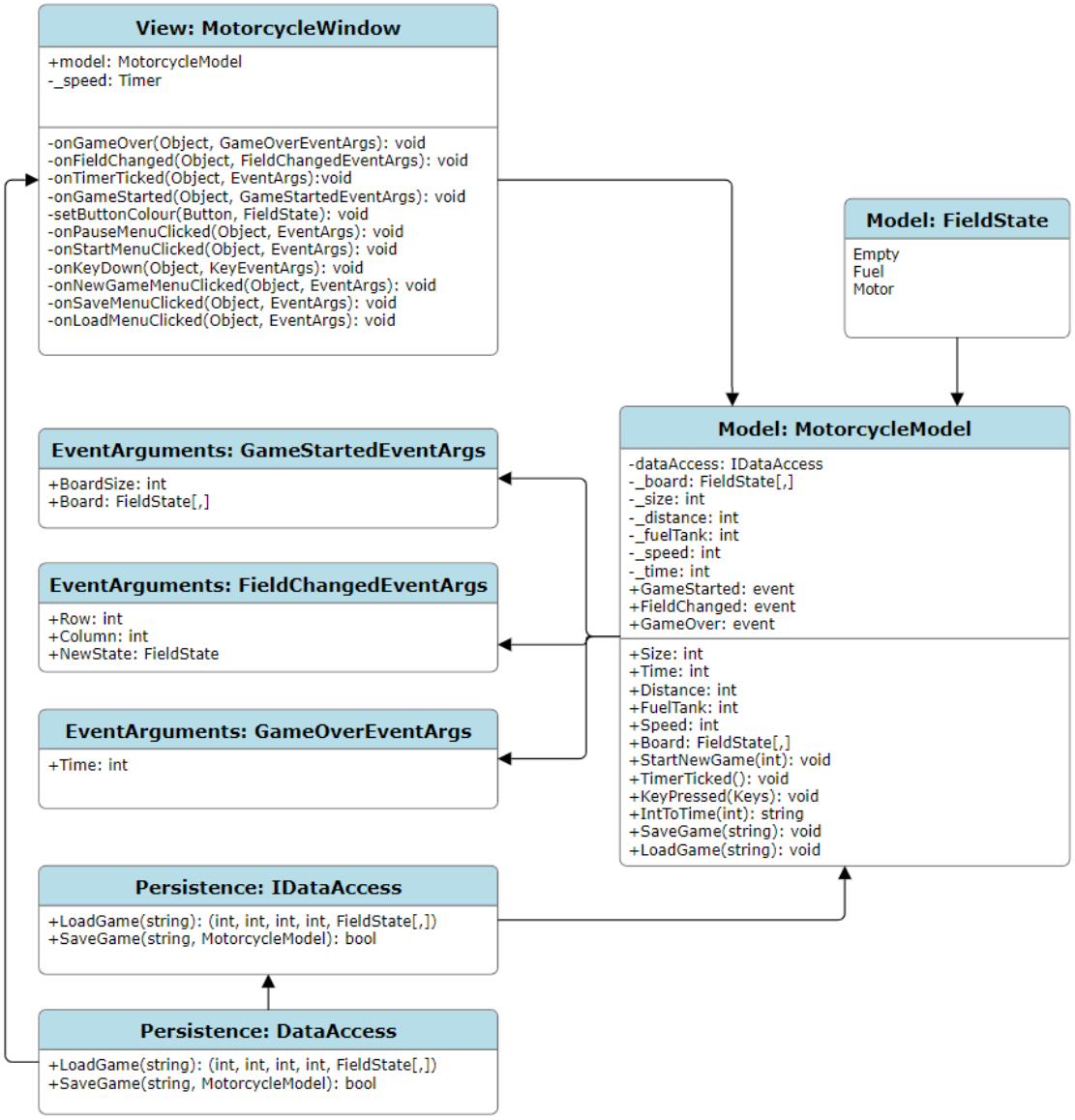
Tervezés:

- Programszerkezet:
  - A programot háromrétegű architektúrában valósítjuk 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 Motorcycle táblával kapcsolatos információk tárolása, valamint a betöltés/mentés biztosítása.
  - A hosszú távú adattárolás lehetőségeit az **IDataAccess** interfész adja meg, amely lehetőséget ad a tábla betöltésére (**LoadGame**), valamint mentésére (**SaveGame**).
  - Az interfész szöveges fájl alapú adatkezelésre a DataAccess 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.
  - A fájl első sora megadja a tábla méretét. A második sor az időt, majd sorban következik: az üzemanyagszint, a sebesség, majd a mátrix értékei.



2.ábra: Az alkalmazás csomagdiagramja

- Modell:
  - A modell lényegi részét a **MotorcycleModel** osztály valósítja meg, amely szabályozza a tábla tevékenységeit, valamint a játék egyéb paramétereit, úgymint az idő (**TimerTicked**). A típus lehetőséget ad új játék kezdésére (startNewGame), valamint lépésre (KeyPressed).
  - A játékállapot változásáról a **FieldChangedEventArg** esemény, míg a játék végéről a **GameOverEventArg** esemény tájékoztat.
  - A modell példányosításkor megkapja az adatkezelés felületét, amelynek segítségével lehetőséget ad betöltésre (**LoadGame**) és mentésre (**SaveGame**)
- Nézet:
  - A nézetet a **MotorcycleWindow** osztály biztosítja, amely tárolja a modell egy példányát (**model**), valamint az adatelérés konkrét példányát (**dataAccess**).
  - A játéktábla egy dinamikusan létrehozott gombmező () reprezentálja. A felületen létrehozzuk a megfelelő menüpontokat, illetve státuszsort, valamint dialógusablakokat, és a hozzájuk tartozó eseménykezelőket. A játéktábla generálását (**onGameStarted**), illetve az értékek beállítását (**setButtonColour**) külön metódusok végzik.
  - A játék időbeli kezelését egy időzítő végzi (**_timer**), amelyet mindig aktiválunk játék során, illetve inaktiválunk, amennyiben bizonyos menüfunkciók futnak.
  - A program teljes statikus szerkezete a 3. ábrán található



3.ábra: Az alkalmazás osztálydiagramja

Tesztelés:

- A modell funkcionalitása egységtesztek segítségével lett ellenőrizve a **MotorcycleGameTest** osztályban.
- Az alábbi tesztek kerültek megvalósításra:
    - **MotorcycleModelNewGameTest**: Új játék indítása, a mezők kitöltése, valamint a gyorsaság, üzemanyagmennyiség és az eltelt idő ellenőrzése.
    - **MotorcycleModelStepRightTest, MotorcycleModelStepLeftTest**: Játékbeli lépések hatásainak ellenőrzése.
    - **MotorcycleModelTickedTest:** Játékbeli időtelés hatásainak ellenőrzése. (Üzemanyagcella lejjebb esése, gyorsulás, idő megnövelése)
    - **MotorcycleModelCatchedFuelCellTest:** A motor által elkapott üzemanyagcellák hatásainak ellenőrzése. (Üzemanyagtank megfelelő növelése)
    - **MotorcycleModelLoadTest:** A játék modell betöltésének tesztelése mockolt perzisztencia réteggel.