Dokumentáció

Feladat:

Készítsünk programot, amellyel az aszteroidák játékot játszahtjuk. A feladatunk az, hogy egy űrhajó segítségével átnavigáljunk egy aszteroidamezőn. Az űrhajóval a képernyő alsó sorában tudunk balra, illetve jobbra navigálni. A képernyő felső sorában meghatározott időközönként véletlenszerű pozícióban jelennek meg az aszteroidák, amelyek folyamatosan közelednek állandó sebességgel a képernyő alja felé. Az idő múlásával egyre több aszteroida jelenik meg egyszerre, így idővel elkerülhetetlenné válik az ütközés. A játék célja az, hogy az űrhajó minél tovább elkerülje az ütközést. A program biztosítson lehetőséget új játék kezdésére, valamint a játék szüneteltetésére (ekkor nem telik az idő, és nem mozog semi a játékban). Ismerje fel, ha vége a játéknak, és jelenítse meg mennyi volt a játékidő. Ezen felül szüneteltetés alatt legyen lehetőség a játék elmentésére, valmaint betöltésére.

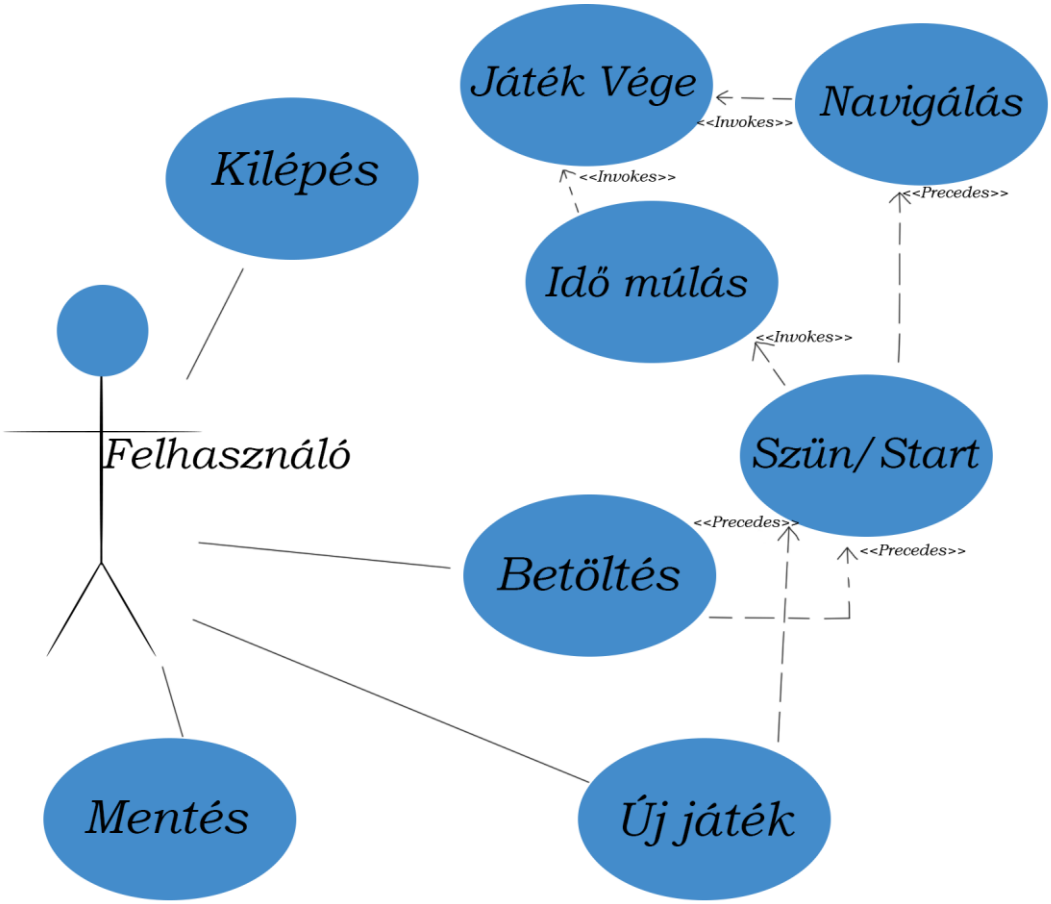
Elemzés:

- A táblaméret nincs megadva, így egy alap beállított 10x11-es pályánk lesz.
- A feladatot egy egy-ablakos Windows Form grafikus felülettel valósítjuk meg.
- Egy leugró menüvel választhatunk "*Új Játék / New Game*","*Játék Mentése / Save Game*","*Játék Betöltése / Load Game*" és "*Kilépés / Quit*" opciók között.
- Az időt folyamatosan egy bal alul elhelyezett mérő mutatja.
- Továbbá a jobb felső sarokban egy "Start / Pause" gombot is el van helyezve.
- Az aszteroidapályát egy 10x11-es "Panel" objektumokból álló táblázattal reprezentáljuk.
- A design-ból adódóan az elérhető maximális idő 30 másodperc.
- Ha vége a játéknak akkor egy eredményt mutató felugró ablak után új játék kezdődik.
- A játék véget ér, ha bármilyen módon aszteroidának ütközünk (Frontálisan vagy oldalról)

Felhasználió Esetek:

	Felhasználói Eset	Leírás	
1	Alkalmazás indítása	GIVEN:	Az alkalmazás telepítve van
		WHEN:	Alkalmazás indítása
		THEN:	Kezdő tábla megjelenik
2	Kilépés	GIVEN:	Játék felülete
		WHEN:	Ablak záró ikonja vagy "Quit" menüpont
		THEN:	Az alkalmazás bezárul
3	Új játék	GIVEN:	A játék felülete

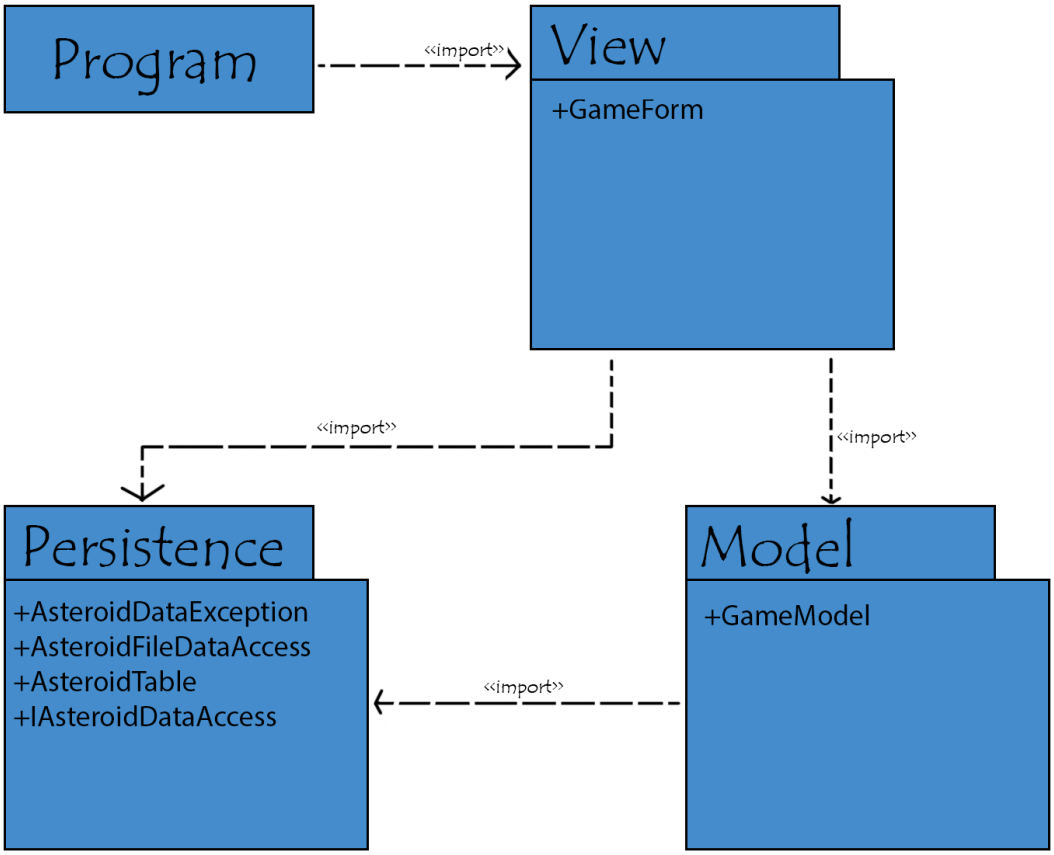
		WHEN:	“New Game” menüpont megnyomása
		THEN:	Új üres játékmező töltődik be alapállapotban
4	Navigálás	GIVEN:	Nem szünetelt játékfelület
		WHEN:	“A” és “D” gombok menyomása
		THEN:	A játékos (Narancs mező also sorban) balra, illetve jobbra mozdul.
5	Játék Vége	GIVEN:	Futó nem szünetelt játékfelület
		WHEN:	Aszteroidával való ütközés
		THEN:	A játék befejeződik és új játékoz kezd az eredmény megadása után
6	Mentés	GIVEN:	Futó szünetelt játékfelület
		WHEN:	“Save Game…” menüpont megnyomása
		THEN:	A játékállás elmentődik egy megadott nevű file-ba
7	Betöltés	GIVEN:	Futó szünetelt játékfelület
		WHEN:	“Load Game…” menüpont megnyomásával egy file kiválasztása
		THEN:	Ha megfelelő a file akkor egy mentett játékállás betöltése



Tervezés:

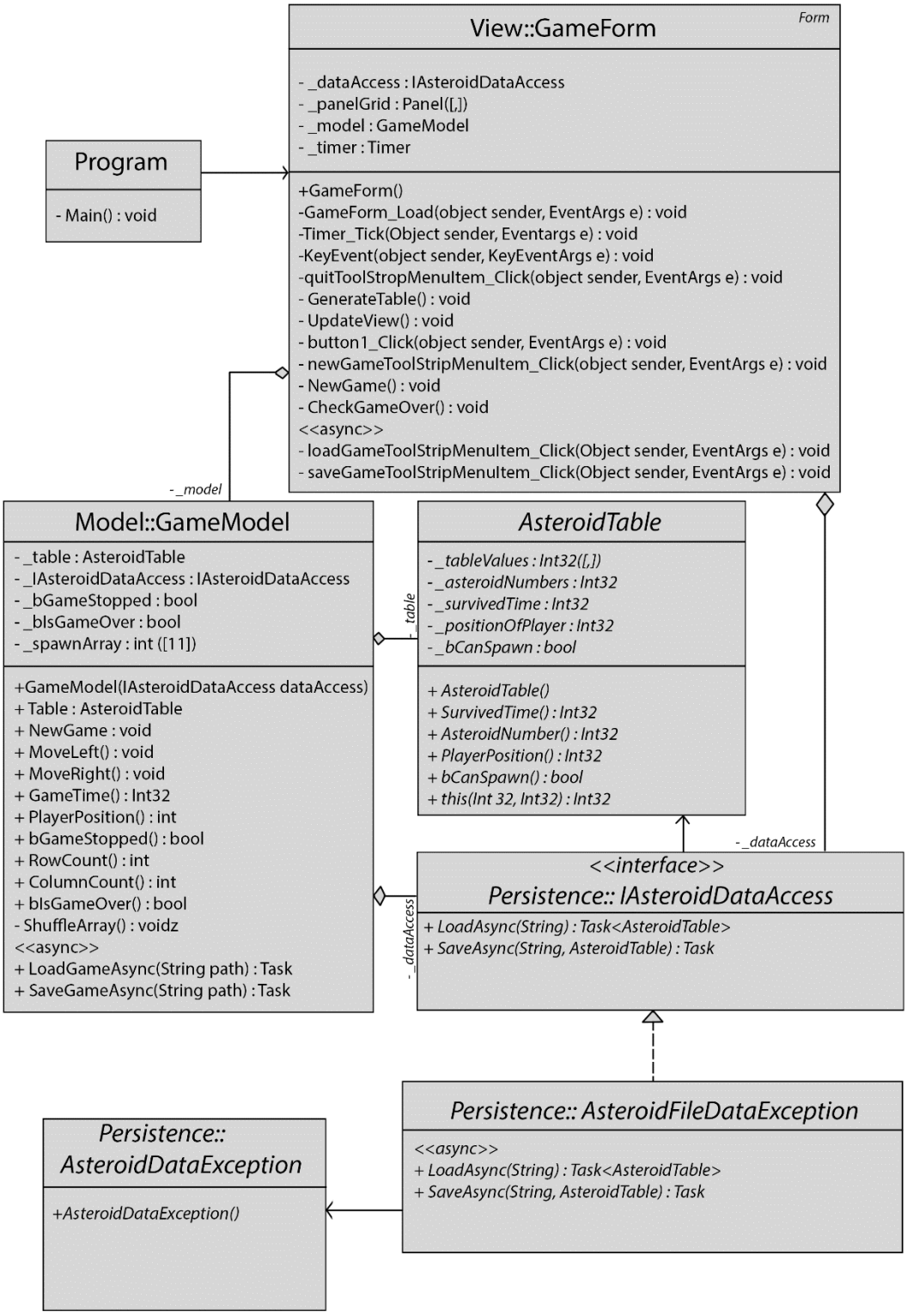
- **Programszerkezet:**
  A programot háromrétegű architektúrában valósítjuk meg. A megjelenítés a View, a modell a Model, míg a perzisztencia a *Persistence* névtérben helyezkedik el. A program csomagszerkezete a 2. ábrán látható.
- **Perszisztencia:**
  - Feladata az adatkezelés valamint a *mentés/betöltés* biztosítása.
  - Az AsteroidTable egy érvényes táblát és játékhoz fontos információkat tartalmaz, a mezők értéke (**_fieldValues**) ez minden esetben 10x11(lásd. Tervezés), újonnan hívott aszteroidák száma(**_asteroidNumbers**), az eddig túlélt idő (**_survivedTime**) a játékos pozíciója(**_positionOfPlayer**) egy Boolean, hogy az adott tick-ben hívhatóak-e az újabb aszteroidák(**_bCanSpawn**) és definiálva vannak ezeknek az értékeknek a Get/Set-erei.
  - Hosszútávú adattárolásához lehetőséget ad az IAsteroidDataAccess interface.
    Így elérhető egy Betöltés(**LoadAsync**) valamint egy Mentés(**SaveSync**) funkció.
    A műveleteket hatékonysági okokból aszinkron módon valósítjuk meg.

- Az interface szöveges file alapú kezelését az **AsteroidFileDataAccess** osztály teszi lehetővé. A felmerülő hibákat az **AteroidDataException** kezeli.
- A program az adatokat szöveges fileként tárolja ezeket az adatokat a programba bel lehet tölteni amikor a játék szünetel.
- A file első sora megadja, hogy mennyi a jelenleg soron következő aszteroidák száma, hogy mennyi idő telt el eddig a játékban(milisec-ben), hogy a játékos pozíciója az alsó soron hol van jelenleg, és a többi sor a mentett aszteroida pálya izomorf megfeleltetése.



- Modell:
  - A modell-t a ***GameModel*** osztály valósítja meg, ami a tábla logikai lépéseit kezeli le.
    Ebben az osztályban példányosítjuk az AsteroidTable osztályt.
  - Itt hívható meg a **ModelTick()** ami lépteti a játékot azaz mozgatja és spawnolja az aszteroidákat.
  - A **MoveLeft()** és a **MoveRight()** amivel a játékos pozíciója változik.
  - A **NewGame()** amivel alaphelyzetbe állítjuk a játékot.
  - Továbbá egy **ShuffleArray()** ami megkeveri a spawn pontokat.

- Nézet:
    - A nézetet a **GameForm** osztály valósítja meg ami tárloja a **GameModel** és a **DataAccess** egy példányát.
    - A játéktábla egy **TableLayoutPanel** amiben el van helyezve egy 10x11-es **Panel** mező
    - A játék idejét a **_timer** kezeli ami a modell léptetését (**GameTick()** függvényét) hívja meg másodpercenként.



Tesztelés:

- A modell funcionalitását egységtesztekkel ellenőrizzük az **AsteroidaTest** osztályban
- Az álábbi tesztek kerültek megvalósításra

- ***TestNewGame()*** →
  Új játék indítása és kezdőértékek ellenőrzése
- ***TestMovement()*** →
  Játékos mozgásának ellenőrzése
- ***TestTick()*** →
  Játékállapot haladásának ellenőrzése
- ***TestAsteroidNumbers()*** →
  Az aszteroidák spawn-olásának ellenőrzése
- ***TestGameOver()*** →
  Játék elvesztésének ellenőrzése

Készítette: Márton Zoltán (B44T65)