1. Beadandó feladat dokumentáció

Készítette:

Restye János Barnabás

f8u9i2@inf.elte.hu

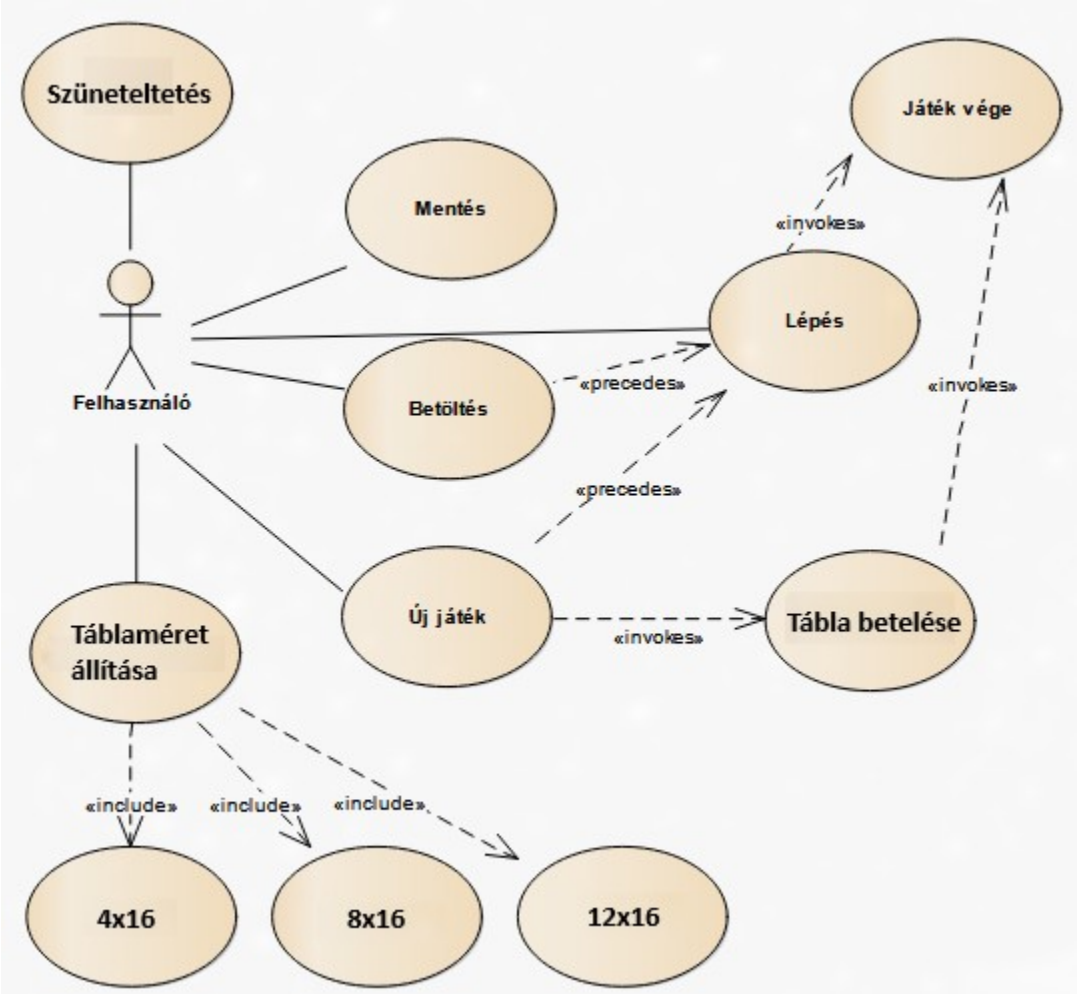
Feladat:

Készítsünk programot a közismert Tetris játékra. Adott egy $n \times m$ pontból álló tábla, amely kezdetben üres. A tábla tetejéről egymás után új, 4 kockából álló építőelemek hullanak, amelyek különböző formájúak lehetnek (kocka, egyenes, L alak, tető, rombusz). Az elemek rögzített sebességgel esnek lefelé, és az első, nem telített helyen megállnak. Amennyiben egy sor teljesen megtelik, az eltűnik a játékmezőről, és minden felette lévő kocka eggyel lejjebb esik. A játékosnak lehetősége van az alakzatokat balra, jobbra mozgatni, valamint forgatni óramutató járásával megegyező irányba, így befolyásolhatja azok mozgását. A játék addig tart, amíg a kockák nem érik el a tábla tetejét. A program biztosítson lehetőséget új játék kezdésére a pályaméret megadásával (4 × 16, 8 × 16, 12 × 16), valamint játék szüneteltetésére (ekkor nem telik az idő, és nem mozognak az elemek). 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 játékot három táblamérettel játszhatjuk: 4x16, 8x16 és 12x16. A program indításakor a játékos játék megkezdése előtt lehetlőséget kap ezek közül választani.
- A feladatot egyablakos asztali alkalmazásként Windows Forms grafikus felülettel valósítjuk meg.
- Az ablak bal oldalán lesz a tábla, a jobb oldalán a menü a következő menüpontokkal: Új játék, Szünet, Mentés, Betöltés. A menüpontok alatt lesz a időszámláló.
- A játéktábla a játékos által választott méretnek fog megfelelni. Új játék menüpont választása után a játék elindul egy tetrominó megjelenésével. A játékos a tetrominó mozgását a WASD gombokkal tudja irányítani.

- Ha a játékos veszít, egy dialógusablak értesíti a játék végéről. Dialógusablak értesíti még a játékost a játék sikeres mentése és betöltése után.
- A felhasználói esetek az 1. ábrán láthatóak.



1. ábra felhasználói esetek diagramja

Tervezés:

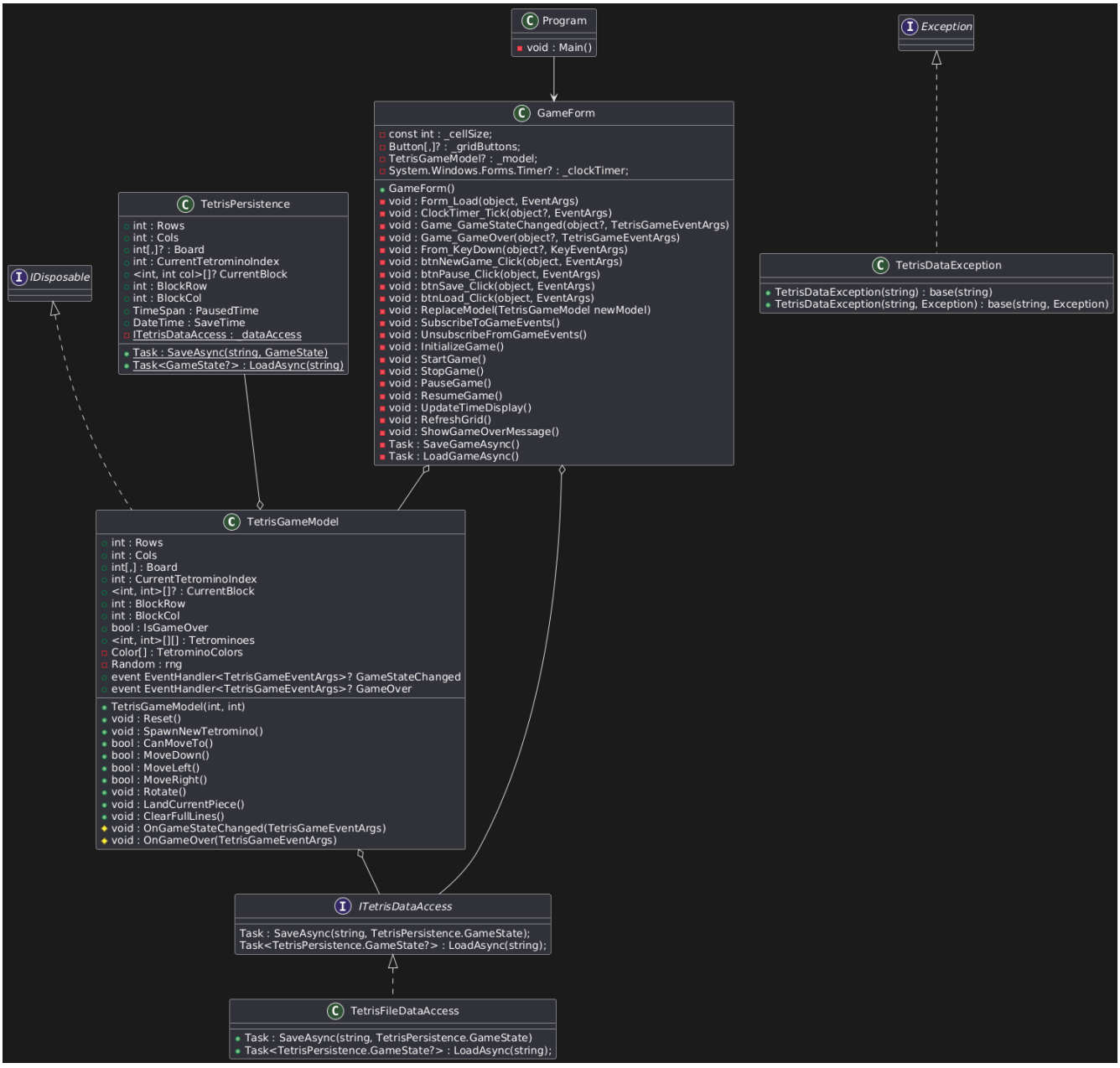
- A programszerkezet:
  - A programot háromrétegű architektúrában valósíjuk meg. A megjelenítés a View, a modell a Model, míg a perzisztencia a Persistence névtérben helyezkedik el. A program csomagszerkezete a 2. ábrán látható.

2. ábra: Az alkalmazás csomagdiagramja

- Modell:
  - A modell lényegi részét a TetrisGameModel osztály valósítja meg, amely szabályozza a tábla tevékenységeit, valamint a játék egyéb paramétereit. Új játéknál megadható a kiinduló játéktábla is.
  - A játék időbeli kezelését egy időzítő végzi (_timer), amelyet inaktiválunk majd (PauseGame), amennyiben bizonyos menüfunkciók futnak, majd újraindítjuk (ResumeGame).
  - A játékállapot megváltozásáról a GameStateChange esemény, míg a játék végéről a GameOver esemény tájékoztat. Az események argumentuma (TetrisGameEventArgs) tárolja a győzelem állapotát, a lépések számát, valamint a játékidőt.
  - A modell példányosításkor megkapja az adatkezelés felületét, amelynek segítségével lehetőséget ad betöltésre (LoadGameAsync) és mentésre (SaveGameAsync)

- Nézet:
  - A nézetet a GameForm osztály biztosítja, amely tárolja a modell egy példányát (_model).
  - A játéktáblát egy dinamikusan létrehozott gombmező (_gridButtons) reprezentálja. A felületen létrehozzuk a megfelelő menüpontokat, valamint dialógusablakokat, és a hozzájuk tartozó eseménykezelőket.
  - A program teljes statikus szerkezete a 3. ábrán látható

3. ábra: Az alkalmazás osztálydiagramja



- Tesztelés:
  - A modell funkcionalitása egységtesztek segítségével lett ellenőrizve a TetrisGameModelTest osztályban.
  - A modell időzítőjét egy ITimer interfészből származtattuk le, így azt a teszt projektben egy MockTimer megvalósítással mockolhatjuk.
  - Az alábbi tesztesetek kerültek megvalósításra:
    - InitializeTest,
    - ResetTest,
    - MoveDownTest,
    - MoveRightTest,
    - MoveDownBlockedTest,
    - LandPieceTest,
    - SpawnTetrominoTest,
    - ClearLineTest,
    - DetectWallsTest,
    - RotatesBlockCorrectly