Első beadandó feladat dokumentáció

Készítette:

Kocsis Márk

Feladat:

Készítsünk programot, amellyel a potyogós amőba játékot lehet játszani, vagyis az amőba azon változatát, ahol a jeleket felülről lefelé lehet beejteni a játékmezőre. A játékmező itt is $n \times n$ -es tábla, és ugyanúgy X, illetve O jeleket potyogtathatunk a mezőre. A játék akkor ér véget, ha betelik a tábla (döntetlen), vagy valamelyik játékos kirak 4 egymás melletti jelet (vízszintesen, vagy átlósan). A program minden lépésnél jelezze, hogy melyik játékos következik, és a tábla egy üres mezőjére kattintva helyezhessük el a megfelelő jelet. Természetesen csak a szabályos lépéseket engedje meg a program. A program biztosítson lehetőséget új játék kezdésére a táblaméret megadásával (10 × 10, 20 × 20, 30 × 30), játék szüneteltetésére, valamint játék mentésére és betöltésére. Ismerje fel, ha vége a játéknak, és jelenítse meg, melyik játékos győzött (a táblán jelölje meg a győztes 4 karaktert). A program folyamatosan jelezze külön-külön a két játékos gondolkodási idejét (azon idők összessége, ami az előző játékos lépésétől a saját lépéséig tart, ezt is mentsük el és töltsük be).

Elemzés:

- A játékot három különböző méretű pályán játszhatjuk. Az alapbeállítás a 10x10 négyzet méretű pálya, de választhatunk 20x20, illetve 30x30 méretű pályát is.
- A feladatot egyablakos asztali alkalmazásként Windows Presentation Foundation grafikus felülettel valósítjuk meg. Az ablakban elhelyezünk egy menüt a következő menüpontokkal: Fájl (Új játék, Játék betöltése, Játék mentése, Kilépés), Beállítások (10x10 tábla méret, 20x20 tábla méret, 30x30 tábla méret). Az ablak alján megjelenítünk egy státuszsort, amely az egyes játékosok lépéssel töltött idejét számlálja.
- A játéktáblát egy megadott méretű gombráccsal alkotjuk meg. A játékosnak, ha lépni akar, akkor a megfelelő gombra kell rákattintania, és annak függvényében, hogy az első vagy a második játékoson van a sor, ha a lépés szabályos volt, egy 'X' vagy egy 'O' jelenik meg a gombon. Az 'X' az első játékos jele, míg a 'O' a második játékosé minden esetben. A játékosnak ebben a programban muszáj az adott gombra kattintania, ahova szeretné „ejteni" a jelet.
- A játéknak akkor van vége, ha van kijön 4 ugyan olyan jel vízszintesen vagy átlósan, vagy ha betelik a tábla. A játék végét felugró ablakban jelezzük.
- A felhasználó esetei:
    - Kilépés
    - Mentés
    - Új játék
    - Betöltés
    - Lépés
        - A játék véget érhet ebből az eseményből
    - Táblaméret módosítása
        - 10x10-es tábla méret

        - 20x20 tábla méret
        - 30x30 tábla méret

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.
- Perzisztenica:
    - Az adatkezelés feladata a Potyogtatós amőba játéktáblájának a tárolása, valamint a mentés és a betöltés.
    - A **Connect4Table** osztály egy érvényes játéktáblát biztosít, amely mindig ellenőrzi, hogy a rajta elvégzett lépés szabályos-e. A van egy paraméteres konstruktora, ez várja a méretet, ami rendre 10, 20 vagy 20 lehet, illetve egy paraméter nélküli konstruktora is van, ez alapjáraton 10x10 méretű táblát hoz létre. A tábla tárolja az utolsónak lépett játékost is **_lastPlayer** adattagként. További metódusai információ lekérdezésére, értékbeállításra vagy ellenőrzésre szolgálnak: **IsEmpty**, **GetValue**, **SetValue**, **SetValueWhenLoad**, **CheckStep**.
    - A hosszú távú adattárolás lehetőségeit az **IConnect4DataAccess** 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 **Connect4FileDataAccess** osztály valósítja meg. A fájlkezelés során fellépő hibákat a **Connect4DataException** kivétel jelzi.
    - A program az adatokat szöveges fájlként tudja eltárolni, melyek az **ctl** kiterjesztést kapják. Ezeket az adatokat a programban bármikor be lehet tölteni, illetve ki lehet menteni az aktuális állást.
    - A **ctl** fájl első sorában a táblaméret és az utolsónak lépett játékos sorszáma található, majd a játéktábla egy „méret x méret” típusú mátrix, az üres hely: 0, X jel: 1 és O jel: 2 megfeleltetéssel.
- Modell:
    - A modell lényegi részét a **Connect4GameModel** osztály valósítja meg. Ez az osztály felel a játék logikájáért. Tárolja az aktuális játékost (**_currentPlayer**), a játékosok idejét (**_player1Time**, **player2Time**), nyert-e az aktuális játékos, illetve van egy potyogtatós amőba játéktábla (**_table**) típusa is. Az osztály metódusával lehet lépni (**Step**), játékot betölteni és menteni.
    - A modell osztályban van két esemény: a játék vége és a játék előrehaladása, ezek argumentuma külön osztályban van megvalósítva, a **Connect4EventArgs**-ban.
- Nézetmodell:
    - A nézetmodell megvalósításához felhasználunk egy általános utasítás (**DelegateCommand**), valamint egy ős változásjelző (**ViewModelBase**) osztályt.
    - A nézetmodell feladatait a **Connect4ViewModel** 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 (**Connect4Field**), amely eltárolja a pozíciót, szöveget, engedélyezettséget, valamint a lépés parancsát (**StepCommand**). 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.
    - A fájlnév bekérését betöltéskor és mentéskor, valamint a figyelmeztető üzenetek megjelenését beépített dialógusablakok segítségével végezzük.
- 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 játék léptetéséhez tárol egy időzítőt is (**_timer**), amelynek állítását is szabályozza az egyes funkciók hatására.
- Teszt:
    - A modell funkcionalitása egységtesztek segítségével lett ellenőrizve a **Connect4GameModelTest** osztályban.
    - Az alábbi tesztesetek lettek megvalósítva:
        - **Connect4GameModelLoadTest**
        - **Connect4GameModelNewGame10Test**
        - **Connect4GameModelNewGame20Test**
        - **Connect4GameModelNewGame30Test**
        - **Connect4GameModelStepTest**
        - **Connect4GameModelAdvanceTimeTest**

Fejlesztési tervek:

- Kisebb méretű képernyőkre optimalizálás
- Fix méretű ablak, a játéktér változik csak
- Potyogtatós mechanizmus megvalósítása, tehát bármely oszlopra lehet kattintani, ha oda még lehet „jelet” dobni

Megjegyzés:

A diagrammok a bemutatásra belekerülnek a dokumentációba, az idő szűkössége miatt a Canvas-ra feltöltött verzióban nincsenek benne. (2021.12.19. 23:49).