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. Mivel mobilra készült az alkalmazás, így az eredeti játéktábla méretek le lettek csökkentve, 5x5, 7x7 és 10x10-es méretre
- A feladatok Xamarin Forms alkalmazásként, Android platformon valósítjuk meg, amely két lapból fog állni. Az alkalmazás portré tájolást támogat.
- A játék négy képernyőn fog megjelenni:
    - Az első képernyő (Játék) tartalmazza a játéktáblát, a játék állását (első és második játékos idejét) a lap alján, az új játék, valamint a beállítások gombjait a lap tetején.
    - A második képernyőn van lehetőség betöltésre, illetve mentésre, valamint a táblaméret állítására (három kapcsolóval).
    - A további két képernyő a betöltésnél, illetve mentésnél megjelenő lista, ahol a játékok elnevezése mellett a mentés dátuma is látható. Mentés esetén ezen felül lehetőség van új név megadására is.
- 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 másodiki 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.
    - A szoftvert két projektből építjük fel, a Xamarin Forms megvalósítást tartalmazó osztálykönyvtárból (.NET Standard Class Library), valamint az Android platform projektből. Utóbbi csupán a perzisztencia Android specifikus megvalósítását tartalmazza, minden további programegységet az osztálykönyvtárban helyezünk el.
    - A megvalósításból külön építjük fel a játék, illetve a betöltés és mentés funkciót, valamennyi rétegben. Utóbbi funkcionalitást újrahasznosítjuk egy korábbi projektből, így nem igényel újabb megvalósítást.
    - A program vezérlését 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 **AndroidDataAccess** osztály valósítja meg. A fájlkezelés során fellépő hibákat a **Connect4DataException** kivétel jelzi.
    - 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 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 nézetet navigációs lapok segítségével építjük fel.
    - A **GamePage** osztály tartalmazza a játéktáblát, amelyet egy **FlowListView** segítségével valósítunk meg (ez egy külső komponens), amelyben Button elemeket helyezünk el.
    - A **SettingsPage** osztály tartalmazza a betöltés, mentés gombjait, illetve **Switch** példányokat a nehézség állítására.
    -
- Környezet:
    - Az **App** osztály feladata az alkalmazás vezérlése, a rétegek példányosítása és az események feldolgozása.
    - • Kezeljük az alkalmazás életciklust, így felfüggesztéskor (**OnSleep**) elmentjük az aktuális játékállást (**SuspendedGame**), folytatáskor (**OnResume**) és újraindításkor (**OnStart**) pedig folytatjuk, amennyiben történt mentés.
- Teszt:
    - A modell funkcionalitása egységtesztek segítségével lett ellenőrizve a **Connect4GameModelTest** osztályban.
    - Az alábbi tesztesetek lettek megvalósítva:
        - **Connect4GameModelLoadTest**
        - **Connect4GameModelNewGame10Test**
        - **Connect4GameModelNewGame20Test**
        - **Connect4GameModelNewGame30Test**
        - **Connect4GameModelStepTest**
        - **Connect4GameModelAdvanceTimeTest**

Fejlesztési tervek:

- 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. (2022.01.04. 23:40).