# 1. Beadandó feladat

**Feladat:**

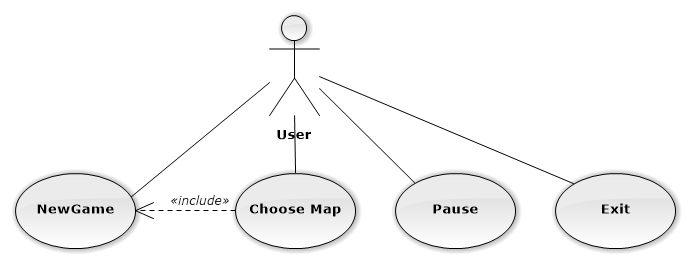
Labirintus

Készítsünk programot, amellyel a következő játékot játszhatjuk. Adott egy 𝑛 × 𝑛 elemből álló játékpálya, amely labirintusként épül fel, azaz fal, illetve padló mezők találhatóak benne, illetve egy kijárat a jobb felső sarokban. A játékos célja, hogy a bal alsó sarokból indulva minél előbb kijusson a labirintusból. A labirintusban nincs világítás, csak egy fáklyát visz a játékos, amely a 2 szomszédos mezőt világítja meg (azaz egy 5 × 5-ös négyzetet), de a falakon nem tud átvilágítani. A játékos figurája kezdetben a bal alsó sarokban helyezkedik el, és vízszintesen, illetve függőlegesen mozoghat (egyesével) a pályán. A pályák méretét, illetve felépítését (falak, padlók) tároljuk fájlban. A program legalább 3 különböző méretű pályát tartalmazzon. A program biztosítson lehetőséget új játék kezdésére a pálya kiválasztásával, valamint játék szüneteltetésére (ekkor nem telik az idő, és nem léphet a játékos), továbbá ismerje fel, ha vége a játéknak. A program játék közben folyamatosan jelezze ki a játékidőt.

**Elemzés:**

* A perzisztencia réteg a játék tábla betöltéséért, a model pedig a játék logikájáért lesz felelős.
* A játék grafikus felületét egy ablakos asztali alkalmazásként készítjük el WPF grafikus felülettel.
* A felületen elkell helyezni két gombot, új játék és pálya betöltése gombot ezeket a menüben helyezzük el.
* A felületen elkell helyezni egy Címkét ami a játék kezdete óta eltelt időt méri ezt a statusbaron helyezzük el.
* A grafikus felületen a játéktáblát képekből állítjuk össze és tesszük bele egy gridbe.
* A játék automatikusan feldob egy ablakot amikor vége a játéknak.

**Használati esetek:**



**Tervezés:**

1. **Programtervezés:**
   * A játékot 4 rétegű architektúrában készítjük el. A megjelenés és a megjelenést kiszolgáló modell a **WPFView,** a modell a **Model**, míg a perzitencia a **Data** névtérben helyezkedik el.
2. **Perzisztencia**:
   * Az adatkezelés feladata a játéktáblával kapcsolatos információk tárolása valamint betöltése.
   * A hosszú távú adattárolás lehetőségeit az **IDataCommunication** interfész adja meg, amely lehetőséget ad a tábla betöltésére (Load).
   * Az interfészt szöveges fájl alapú adatkezelésre a **MyDataDirector** osztály valósítja meg. A fájlkezelés során fellépő hibákat a DataException kivétel jelzi.
   * Helyes file felépítése esetén a file egy 'W','R' karakterkből álló n\*n es mátrixot tartalmaz, szóközökkel elválasztva.
3. **Modell**
   * A modell lényegi részét a **Game** osztály valósítja meg, amely szabályozza a játék tevékenységeit, valamint a játék egyéb paramétereit úgy mint az idő (**time**) ,hol tartózkodik a játékos (**player**), és magát a játék táblát (**gameTable**) is.
   * Új játék indításakor példányosításakor megadhatjuk a játéktábla paramétereit és magát a táblát is, ha nem adunk meg új pályát az előző pályát indítja újra. Biztosítunk lehetőséget pálya betöltésére is ekkora ha helyes pálya lett kiválasztva új játékot indít az új pályával.
   * A modell események segítségével kommunikál a grafikus felülettel, ilyen például amikor a modell tájékoztat, hogy eltelt idő ez a **newTime** esemény, ha vége lett a játéknak **gameEnded** és akkor is ha a betöltés után új játék indult **newGameStarted.**
   * A játék idelyét egy időzítő végzi (**timer**) amelyet a játék állapota ( folyamatban, szüneteltetve, játékvége) alapján aktiválunk vagy inaktiválunk
4. **Nézet**
   * A nézetet a **MainWindow.xaml** leíró fájl biztosítja ami *Bindingokkal* kapcsolódik a nézet modellhez (**GameViewModel**), ami biztosítja az xaml számára az adatokat.
   * A **GameViewModel** kommunikál közvetlenül a Modellel.
5. **Tesztelés**
   * A modell funkcionalitása egységtesztek segítségével lett ellenőrizve a **LabyrinthModelTest** osztályban.
   * Az alábbi tesztesetek kerültek megvalósításra:
     1. TestNewGame:

Új játék indítása, a játék méret a játék idő helyes beállítása a játékos a helyes kezdőpozícióját ellenőrzi.

* + 1. MovePlayer

A játékbeli lépés hatásának ellenőrzése, sikerül-e a játékosnak helytelen mezőre lépnie- vagy sem. Helyes mezőre lép-e a Játékos. Helytelen mezőre lépés kisérlete során a játékos poziciója ne változzon.

* + 1. ModelNextFieldTeszt

Ez a model egy segéd fügvényét teszteli, ami többek között szomszédos mezők elérhetőségét vizsgálja. Illetve, hogy ha egy olyan mezővel szeretnénk valamit kezdeni ami nem helyezkedik el a játéktáblán ne történjen semmi.

**Osztály szerkezet:**

