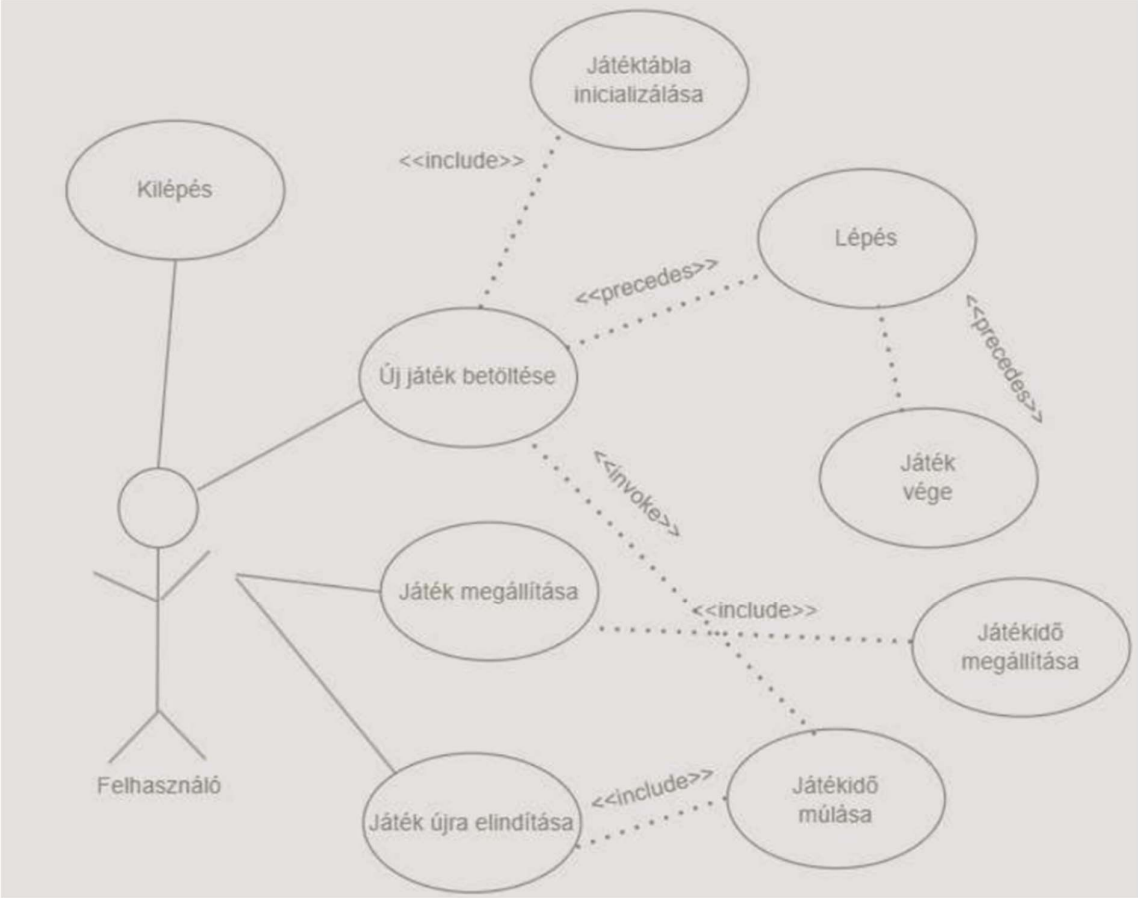
Feladat:

Készítsünk programot, amellyel a következő játékot játszhatjuk. Adott egy □□ ×□□ elemből álló játékpálya, amely falakból és padlóból áll, valamint őrök járőröznek rajta. A játékos feladata, hogy a kiindulási pontból eljusson a kijáratig úgy, hogy közben az őrök nem látják meg. Természetesen a játékos, illetve az őrök csak a padlón tudnak járni. Az őrök adott időközönként lépnek egy mezőt (vízszintesen, vagy függőlegesen) úgy, hogy folyamatosan előre haladnak egészen addig, amíg falba nem ütköznek. Ekkor véletlenszerűen választanak egy új irányt, és arra haladnak tovább. Az őr járőrözés közben egy 2 sugarú körben lát (azaz egy 5 × 5-ös négyzetet), ám a falon nem képes átlátni. A játékos a pálya előre megadott pontján kezd, és vízszintesen, illetve függőlegesen mozoghat (egyesével) a pályán. A pályák méretét, illetve felépítését (falak és kijárat helyzete, játékos és őrök kezdőpozíciója) 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. Ekkor jelenítse meg, hogy győzött, vagy veszített-e a játékos.

Elemzés:

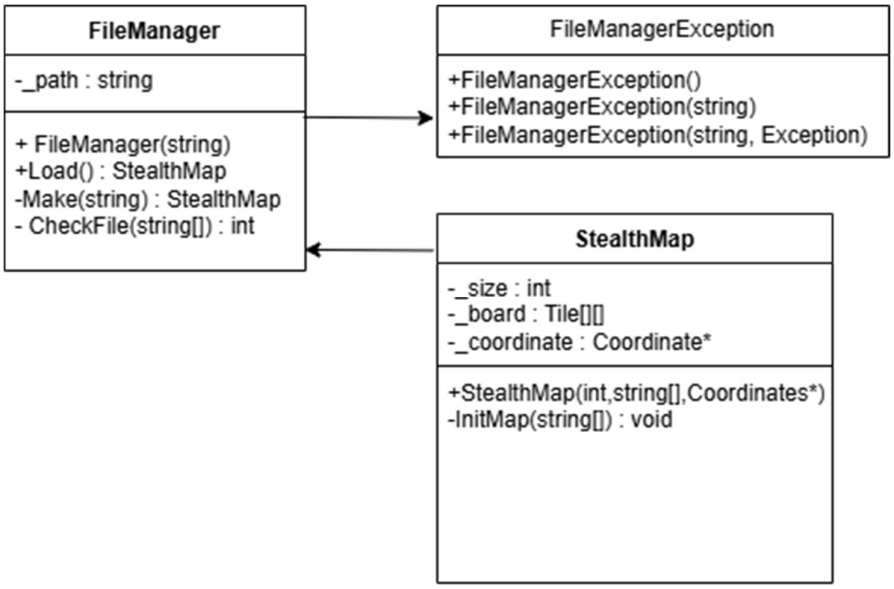
- A játékot legalább 3 pályán játszhatjuk, amit a felhasználó gombok segítségével
- választhat ki.
- A feladatot egyablakos asztali alkalmazásként Windows Forms grafikus felülettel
- valósítjuk meg.
- Az ablakban egy menüsávot, benne két gombbal, egyiket a játék indításához pálya
- kiválasztásával, a másikat a játék megállítására használjuk
- A játék pályáját dinamikusan állítjuk elő a méreteknek megfelelően
- A játék végével a program egy dialógusablak segítsével közli a felhasználóval, hogy
- véget ért a játék.

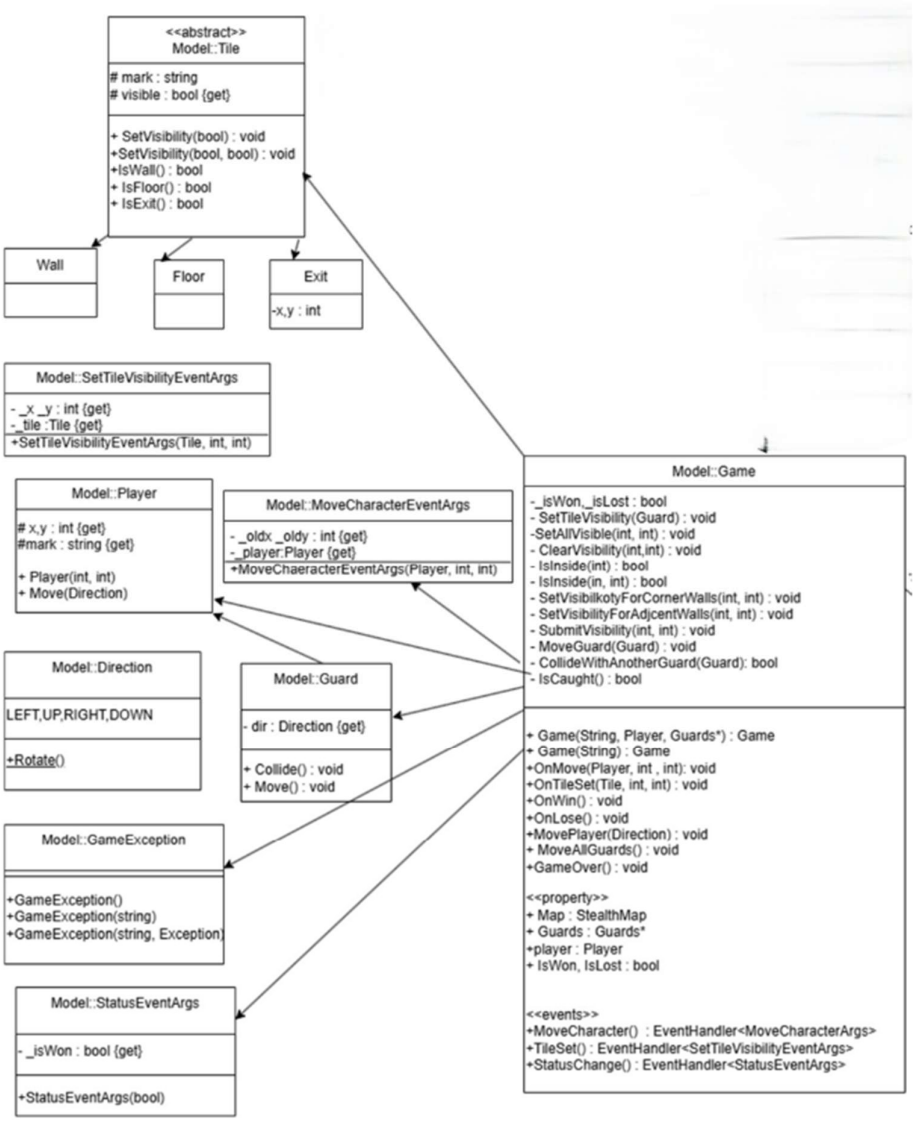


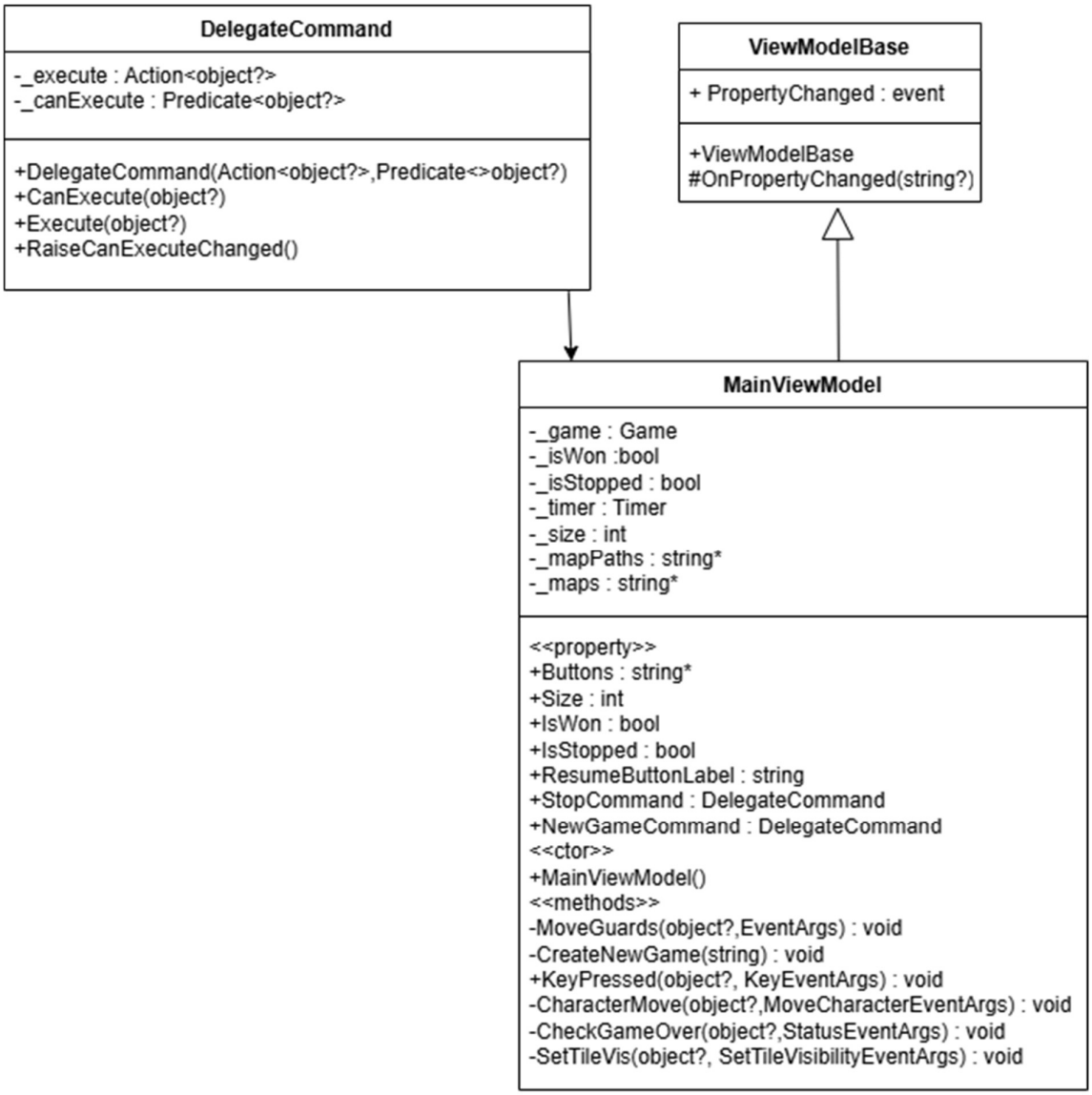
Tervezés:

- Programtervezet
  - A programot négyrétegű architektúrában valósíjuk meg. A megjelenítés a View, a Nézet modelljét a ViewModel,a modell a Model, míg a perzisztencia a Persistence névtérben helyezkedik el. A program csomagszerkezete a következő ábrán látható.
  - A program szerkezetét két projektre osztjuk implementációs megfontolásból: a Persistence és Model csomagok a program felületfüggetlen projektjében, míg a View csomag a és a View Model a WPF-től függő projektjében kap helyet.
- Perzisztencia:
  - A táblával kapcsolatos adatok tárolása a feladata
  - A StelthMap osztály egy érvényes pálya leírását tartalmazza
  - A FileManager osztály lehetőséget biztosít a pálya betöltésére
  - A FileManager osztály szöveges file-okat olvas be, ellenőrzi helyességüket, amennyiben a file helytelen, FileManagerException váltódik ki
  - A fájl első n sora a táblát, majd utána legalább 2 sort tartalmaz a pályán lévő karakterek helyzetéről

- Modell:
  - A modellt javarészt a Game osztály valósítja meg, egy játékot ír le, teszteli a játék végét, változtatja a karakterek pozícióját, a tábla mezőnek tulajdonságaiit.
  - A SetTileVisibiltyEventArgs felelős a pálya adott elemeinek a változásait tárolni.
  - A StatusEventArgs tárolj a játék végállapotát
  - A karakterek irányát a Direction felsoroló írja le.
  - A MoveCharacterEventArgs a karakterek helyváltoztatásával kapcsolatos adatok leírására szolgál.
- NézetModell
  - Megvalósítja a modell-t
  - A nézetmodell tárolja a megjelenítendő adatokat
  - Tárolja a pálya aktuális méretét
  - Parancsokat tárol amik bizonyos feladatokat látnak el
  - Időzítőhöz kötött eseményei vannak
  - A tualjdonságai változása mindig kivált egy eseményt
- Nézet
  - Köt a nézetmodell adattagjaihoz
  - Tartalmazza a szükséges vezérlőket
  - A fájlok kiválasztása itt történik meg
  - Dinamikus változik a nézetmodell alapján


FileManager
-_path : string
+ FileManager(string)
+Load() : StealthMap
-Make(string) : StealthMap
- CheckFile(string[]) : int
FileManagerException
+FileManagerException()
+FileManagerException(string)
+FileManagerException(string, Exception)
StealthMap
-_size : int
-_board : Tile[][]
-_coordinate : Coordinate*
+StealthMap(int,string[],Coordinates*)
-InitMap(string[]) : void


<<abstract>>
Model::Tile
mark : string
visible : bool {get}
+ SetVisibility(bool) : void
+SetVisibility(bool, bool) : void
+IsWall() : bool
+ IsFloor() : bool
+ IsExit() : bool
Wall
Floor
Exit
-x,y : int
Model::SetTileVisibilityEventArgs
- _x ,_y : int {get}
- _tile :Tile {get}
+SetTileVisibilityEventArgs(Tile, int, int)
Model::Player
x,y : int {get}
#mark : string {get}
+ Player(int, int)
+ Move(Direction)
Model::MoveCharacterEventArgs
- _oldx ,_oldy : int {get}
- _player:Player {get}
+MoveChaeracterEventArgs(Player, int, int)
Model::Game
-_isWon,_isLost : bool
- SetTileVisibility(Guard) : void
-SetAllVisible(int, int) : void
- ClearVisibility(int,int) : void
- IsInside(int) : bool
- IsInside(in, int) : bool
- SetVisibilkotyForCornerWalls(int, int) : void
- SetVisibilityForAdjcentWalls(int, int) : void
- SubmitVisibility(int, int) : void
- MoveGuard(Guard) : void
- CollideWithAnotherGuard(Guard): bool
- IsCaught() : bool
+ Game(String, Player, Guards*) : Game
+ Game(String) : Game
+OnMove(Player, int , int): void
+OnTileSet(Tile, int, int) : void
+OnWin() : void
+OnLose() : void
+MovePlayer(Direction) : void
+ MoveAllGuards() : void
+GameOver() : void
<<property>>
+ Map : StealthMap
+ Guards : Guards*
+player : Player
+ IsWon, IsLost : bool
<<events>>
+MoveCharacter() : EventHandler<MoveCharacterArgs>
+TileSet() : EventHandler<SetTileVisibilityEventArgs>
+StatusChange() : EventHandler<StatusEventArgs>
Model::Direction
LEFT,UP,RIGHT,DOWN
+Rotate()
Model::Guard
- dir : Direction {get}
+ Collide() : void
+ Move() : void
Model::GameException
+GameException()
+GameException(string)
+GameException(string, Exception)
Model::StatusEventArgs
- _isWon : bool {get}
+StatusEventArgs(bool)


DelegateCommand
-_execute : Action<object?>
-_canExecute : Predicate<object?>
+DelegateCommand(Action<object?>,Predicate<>object?)
+CanExecute(object?)
+Execute(object?)
+RaiseCanExecuteChanged()
ViewModelBase
+ PropertyChanged : event
+ViewModelBase
#OnPropertyChanged(string?)
MainViewModel
-_game : Game
-_isWon :bool
-_isStopped : bool
-_timer : Timer
-_size : int
-_mapPaths : string*
-_maps : string*
<<property>>
+Buttons : string*
+Size : int
+IsWon : bool
+IsStopped : bool
+ResumeButtonLabel : string
+StopCommand : DelegateCommand
+NewGameCommand : DelegateCommand
<<ctor>>
+MainViewModel()
<<methods>>
-MoveGuards(object?,EventArgs) : void
-CreateNewGame(string) : void
+KeyPressed(object?, KeyEventArgs) : void
-CharacterMove(object?,MoveCharacterEventArgs) : void
-CheckGameOver(object?,StatusEventArgs) : void
-SetTileVis(object?, SetTileVisibilityEventArgs) : void

Tesztelés:

- A játékos kilépése a pályáról
- Játékos mozgás
- Megfelelő őr lépés
- Őrök ütköznek egymással
- Játék betöltése
- Rossz fájlból való betöltés
- Nem létező fájl betöltése