3. csoport Dokumentáció

Készítette:

Ács Botond – DF9FL6 – acs.botond42@gmail.com

Dóra László – IO89I5 – doralaci98@gmail.com

Jakab Olivér – CAGJCQ – jakaboliver98@gmail.com

Ismertetés:

A feladat egy robotrendszer kialakítása, ami egy raktár logisztikáját automatizálja, mely precíz és gyors feladatvégzést tud végezni folyamatosan. A robotok közvetlen a polcok alá mennek, melyet felemelnek és az egész polcot viszik a célállomáshoz. Így a raktárnak szükséges rendelkezni a robotokhoz tartozó energia-töltőállomással, illetve célállomással ahová a robot a polcokat viszik közvetlen, melyekről az alkalmazottak a megfelelő termékeket leveszik további csomagolásra. Ilyenkor a robot visszaviszi a polcot az eredeti helyére.

A program számolja a robotok elhasznált energiáját, a leszállított csomagok darabszámát. A programnak támogatnia kell mint a beolvasott szimulációs adatokkal való szimuláció újraindítását illetve egy új adatokkal való új szimuláció kialakítását.

Ezzel fel lehet gyorsítani a raktári logisztikát, hosszú távon pénzt lehet spórolni a munkaerőn, illetve precíz állandó munkavégzésre lehet számítani.

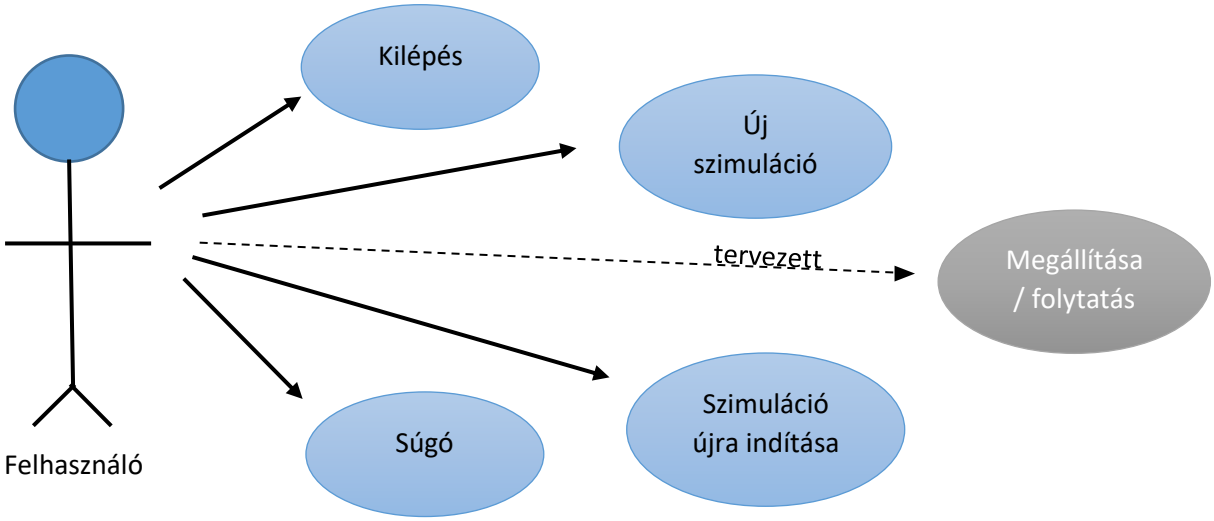
Mérföldkövek:

1. 7. hétre alap prototípus és teljes git használat.
2. 10. hétre 90%-os usercase teljesítmény, unit testek.
3. 13. hétre végső termék.

Ezeken a nagyobb mérföldköveken kívül a fejlesztési munkák folyamatosan zajlanak. Hetente minimum egy élőben / internetes platformon összeülés, folyamatok megbeszélése, munkák további kiosztása.

Elemzés:

- A feladatot egyablakos asztali alkalmazásként *Windows Presentation Foundation* grafikus felülettel valósítjuk meg.
- A program indításakor, fájlbeolvasással olvasunk be egy előre megadott NxM alakú táblát.
- Az ablakban elhelyezünk egy menüt a következő menüpontokkal: File (Új szimuláció, Jelenlegi szimuláció újraindítása, Kilépés), Sugó.
- Az ablak alján megjelenik egy státuszsor, ami számolja a lépésszámot, leadott termékeke és az összes energiafogyasztást.
- A szimulációs táblát egy NxM-es címkékből álló rács reprezentálja.
    - Üres mező – fehér
    - Robot – Sárga
    - Polc – Szürke
    - Töltőállomás – Kék
    - Leadási állomás – Zöld
    - Robot polc alatt – Sötétebb narancssárga
    - Robot viszi a polcot – Piros
- A felhasználói esetek az első ábrán láthatóak.

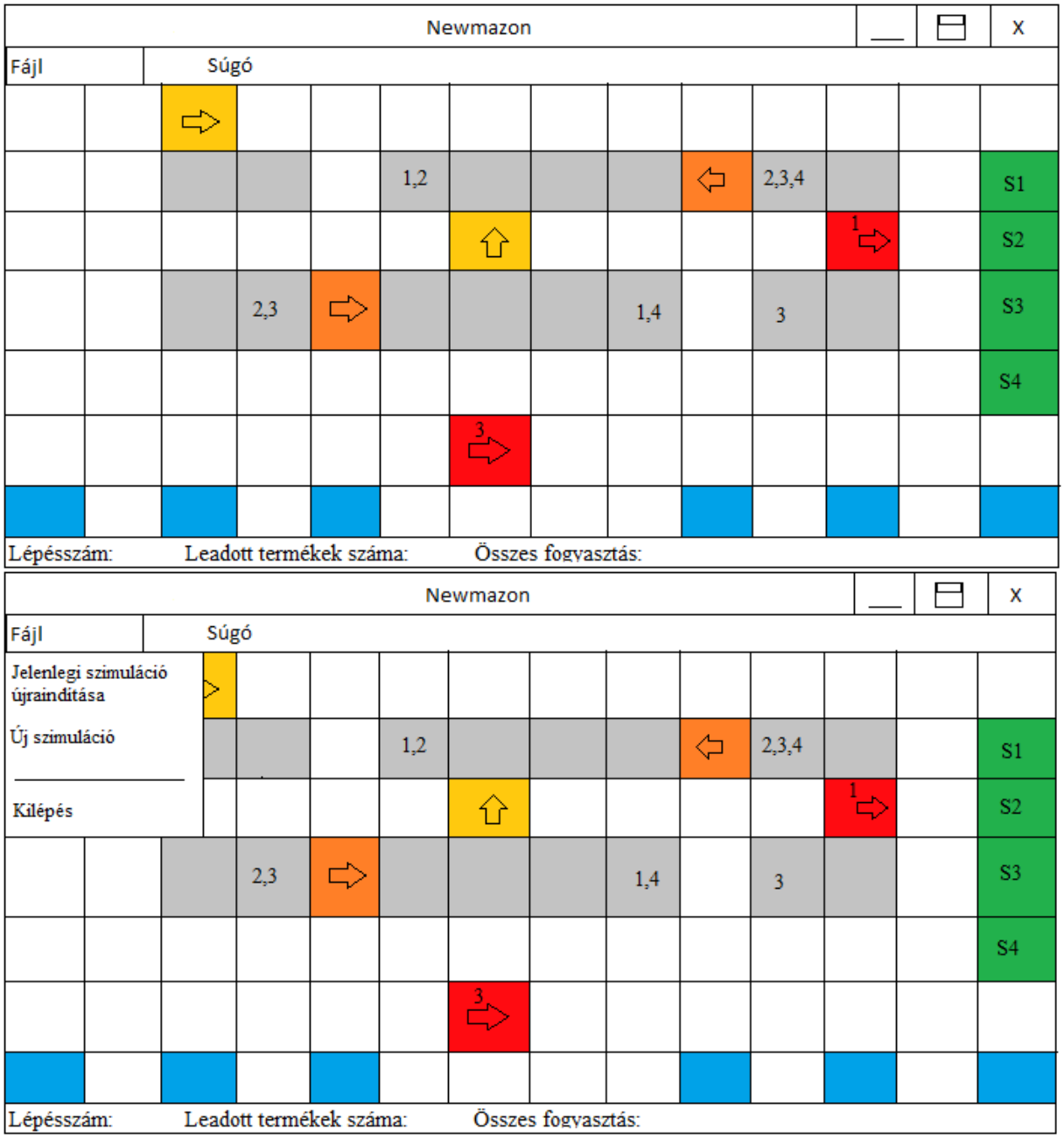


Felhasználói esetek

Nemfunkcionális követelmények:

- A program biztonságos, hisz egy központi vezérlő egység vezérli a robotokat, mely folyamatosan frissíti azok helyeit.
- A program válaszideje a robotok útkereső algoritmusával megegyező.
- Maga a program tárfoglalása minimális.
- A program egyszerűen hordozható és skálázható.
- Jogi korlátok egyelőre nincsenek, mivel iskolai project és nem forgalmazható.

Felhasználói felület:



Felhasználói történetek/funkcionális tesztesetek:

Felhasználóként szeretném, hogy:

- fájlbeolvasással tudjak választani előre megírt alapállapotok között azért, hogy többféle szimuláció működését tudjam ellenőrizni.
- a robotok időben fel tudják magukat tölteni a lehető legközelebbi töltőállomáson azért, hogy ne merüljenek le menet közben.
- a robotok egy polc alatt tartózkodva meg tudják emelni azt azért, hogy utána el tudják vinni a célállomásra.
- a robotok tudjanak az polcok alatt közlekedni (ha éppen nem cipelnek polcot) a könnyebb közlekedés érdekében.

- a robotok valamilyen módon ki tudják kerülni egymást azért, hogy fennakadásmentesen működjön a szimuláció. (nem fő prioritás, csak ha már minden mással kész vagyunk)

Felhasználói eset	Leírás	
Alkalmazás indítása	*GIVEN*	az alkalmazás telepítve van
	WHEN	alkalmazás indítása
	THEN	felhasználó által megadott fájlból szimuláció elindítása
Kilépés	*GIVEN*	fut a szimuláció
	WHEN	lezáró ikonra kattintás / Fájl menüből a "Kilépés" menüpont kiválasztása
	THEN	szimuláció, és ezzel együtt az alkalmazás befejezése
Szimuláció újraindítása	*GIVEN*	fut a szimuláció
	WHEN	Fájl menüből a "Jelenlegi szimuláció újraindítása" menüpont kiválasztása
	THEN	szimuláció újraindítása ugyanabból a fájlból
Új szimuláció	*GIVEN*	fut a szimuláció
	WHEN	Fájl menüből az "Új szimuláció" menüpont kiválasztása
	THEN	új szimuláció elindítása a felhasználó által megadott fájlból
Súgó megnyitása	*GIVEN*	fut a szimuláció
	WHEN	"Súgó" menüpont kiválasztása
	THEN	megnyílik a súgó, amely leírja a program működését

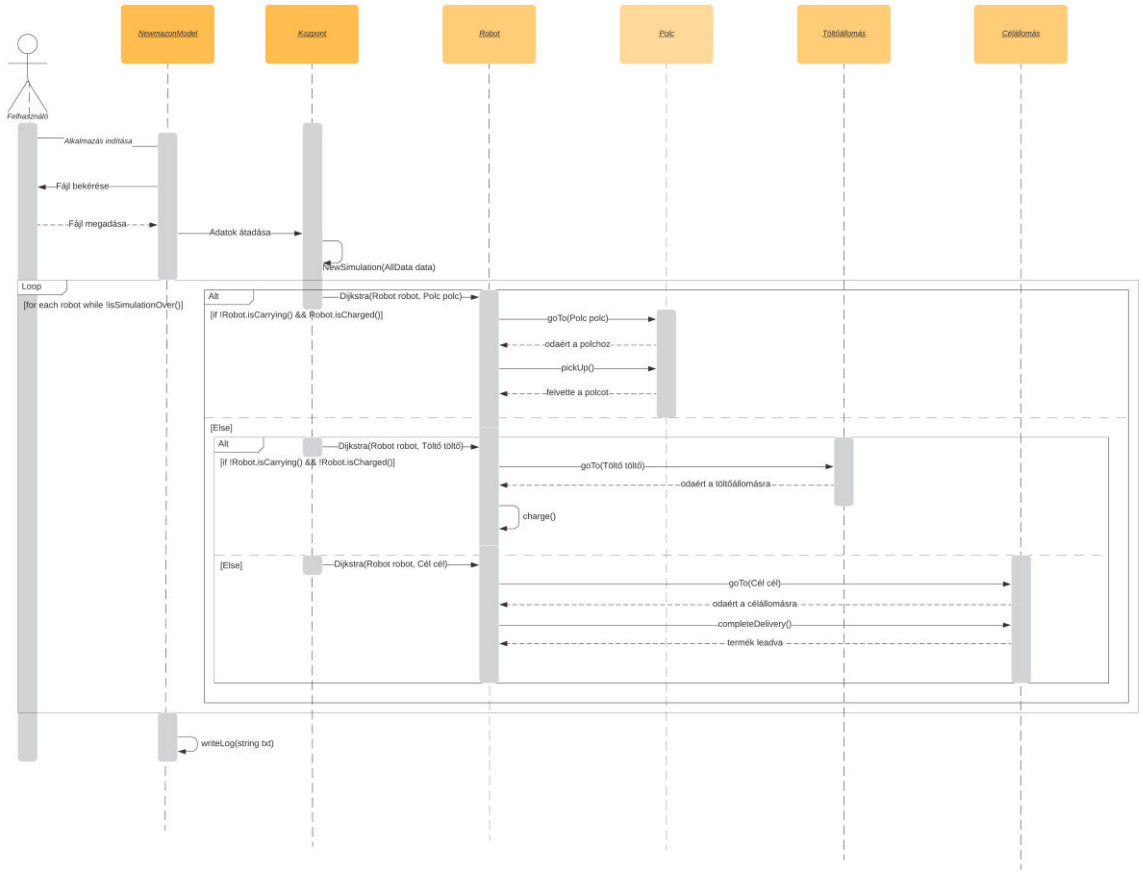
Központi esetek	Leírás	
Robot feltöltése	*GIVEN*	fut a szimuláció
	WHEN	egy robot töltöttsége bizonyos (kb. 25-30%) szint alá esik, és éppen nincs nála polc
	THEN	a központ elküldi a robotot a legközelebbi szabad töltőállomásra, ahol ezután az feltöltődik
Robot útkeresése (nem célállomás)	*GIVEN*	fut a szimuláció
	WHEN	van szabad robot
	THEN	a központ útkereső algoritmussal kiszámolja a legkedvezőbb utat az úticélhoz (lehet polc, töltőállomás) és ez alapján odaküldi a robotot (akár polcokon keresztül is)
Polc felvétele	*GIVEN*	fut a szimuláció
	WHEN	robot áll egy olyan polc alatt, amin még vannak termékek
	THEN	a központ utasítja a robotot, hogy vegye fel a polcot
Robot útkeresése (célállomás)	*GIVEN*	fut a szimuláció
	WHEN	egy robot éppen felvett egy polcot
	THEN	a központ útkereső algoritmussal kiszámolja a legkedvezőbb utat a célállomáshoz, és ez alapján odaküldi a robotot (nem mehet polcokon keresztül)
Elakadás kezelése	*GIVEN*	fut a szimuláció
	WHEN	két robot egymás útját állja
	THEN	a központ újraszámolja az útkereső algoritmusát az egyik robotra, hogy kikerülje a másikat

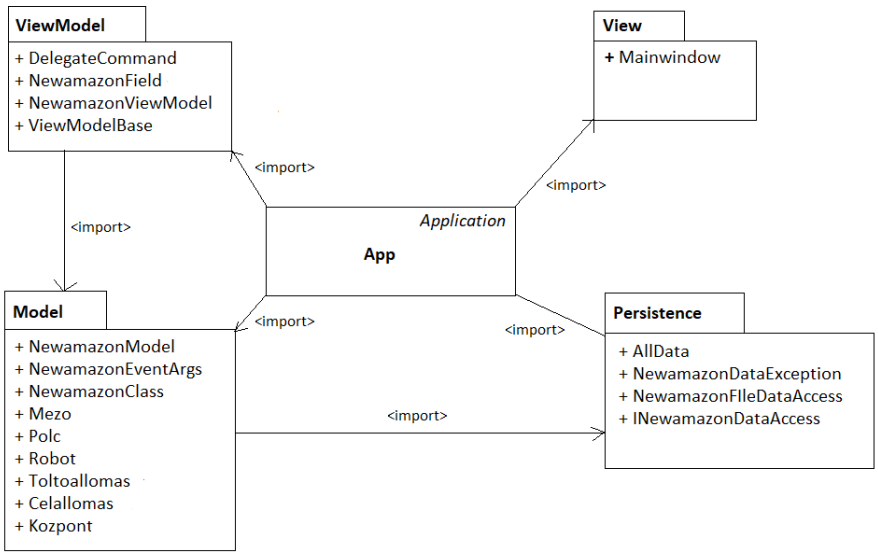
Programműködés áttekintése szekvenciával:

A program indításkor lehetőséget biztosít arra, hogy a felhasználó előre megadott fájlokból tölthesse be a szimulációt. Miután ez megtörtént, megjelenik a raktár, és a benne lévő elemek (robotok, polcok, falak, töltő- és leadási állomások) a fájlban megadott elrendezés szerint. Ezután a felhasználónak nincsen beleszólása a program működésébe. (maximum kilépni, új szimulációt indítani, vagy a már betöltött szimulációt tudja újraindítani)

A robotok a központ irányítása alapján elviszik a polcokat a megfelelő célállomásra, ezután visszateszik őket a helyükre. Ha egy bizonyos szint alá esik a töltöttségi szintjük, és éppen nincsen náluk állvány, akkor a központ elküldi őket a legközelebbi töltőállomásra.

Mindez addig megy, amíg el nem fogynak a polcokról a termékek. Ekkor a program egy log fájlt készít, amelyben megtalálható, hogy hány lépésig tartott a teljes feladat végrehajtása, minden egyes robotra, hogy összesen mennyi energiát fogyasztott, valamint az, hogy összesen mennyi energia kellett a feladat végrehajtásához. Ezután a felhasználó – ugyanúgy, mint eddig – kiléphet, új szimulációt indíthat, vagy a jelenlegit indíthatja újra.

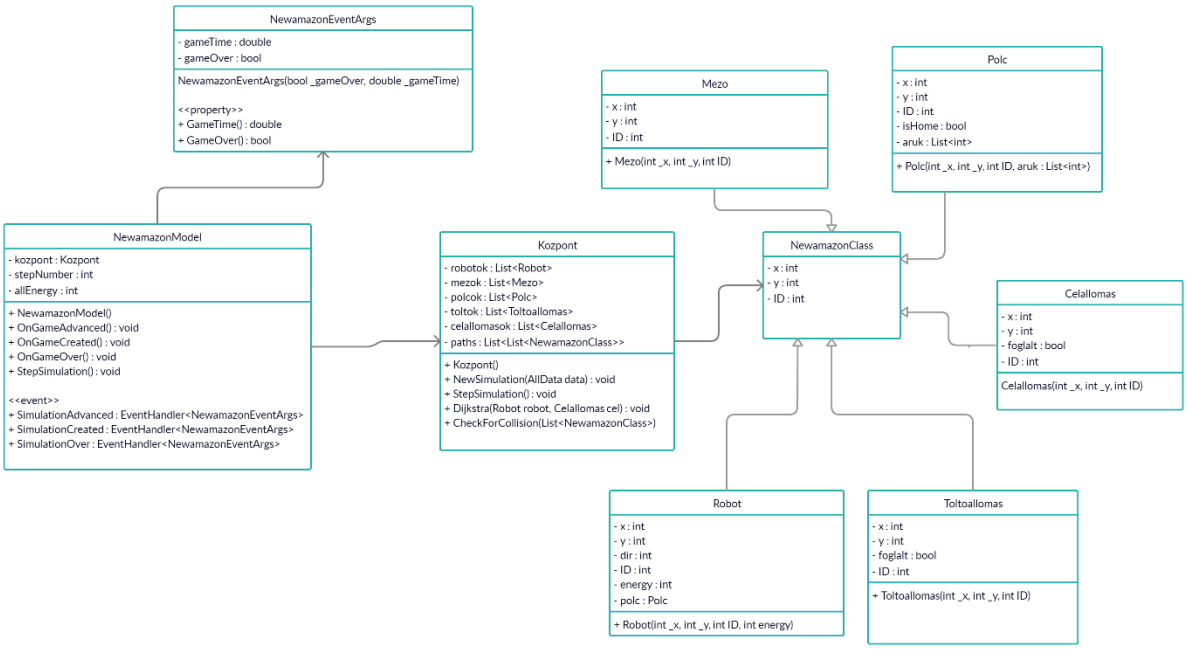

NewmazonModel
Kozpont
Robot
Polc
Töltőállomás
Célállomás
Felhasználó
Alkalmazás indítása
Fájl bekérése
Fájl megadása
Adatok átadása
NewSimulation(AllData data)
Loop
[for each robot while !isSimulationOver()]
Alt
[if !Robot.isCarrying() && Robot.isCharged()]
Dijkstra(Robot robot, Polc polc)
goTo(Polc polc)
odaért a polchoz
pickUp()
felvette a polcot
[Else]
Alt
[if !Robot.isCarrying() && !Robot.isCharged()]
Dijkstra(Robot robot, Töltő töltő)
goTo(Töltő töltő)
odaért a töltőállomásra
charge()
[Else]
Dijkstra(Robot robot, Cél cél)
goTo(Cél cél)
odaért a célállomásra
completeDelivery()
termék leadva
writeLog(string txt)



Osztályok áttekintése

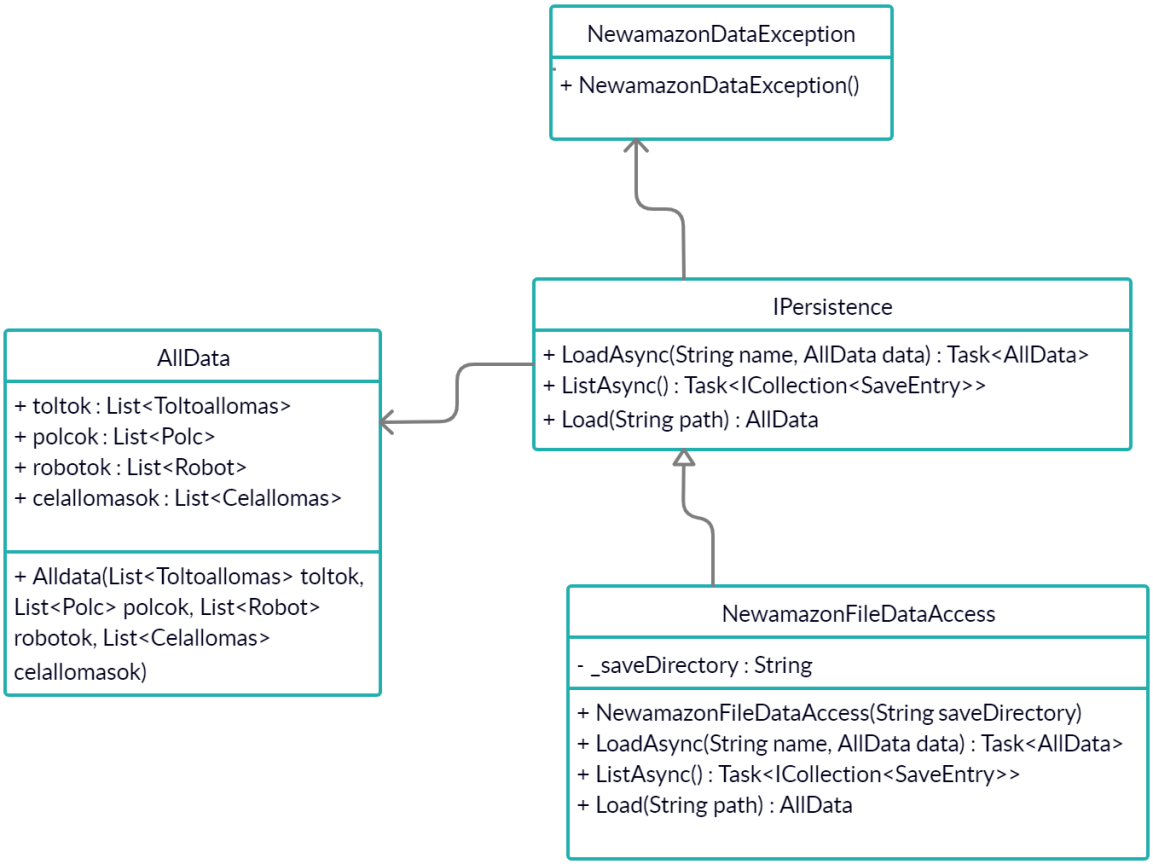
Model:

- NewamazonModel.cs
- NewamazonEventArgs.cs
- NewamazonClass.cs
- Mezo.cs
- Polc.cs
- Robot.cs
- Toltoallomas.cs
- Celallomas.cs
- Kozpont.cs



Persistance:

- AllData.cs
- NewamazonDataException.cs
- NewamazonFileDataAccess.cs
- INewamazonDataAccess.cs

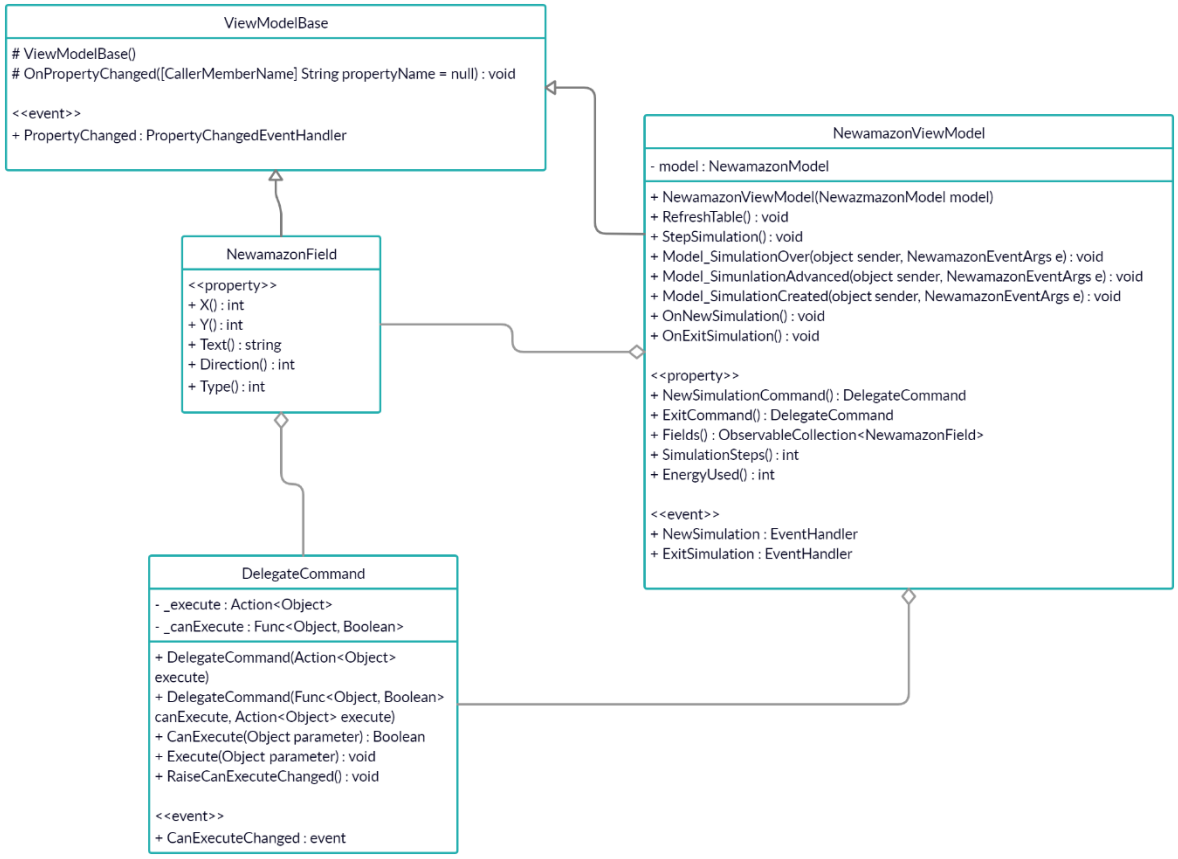


View:

- MainWindow.xaml.cs

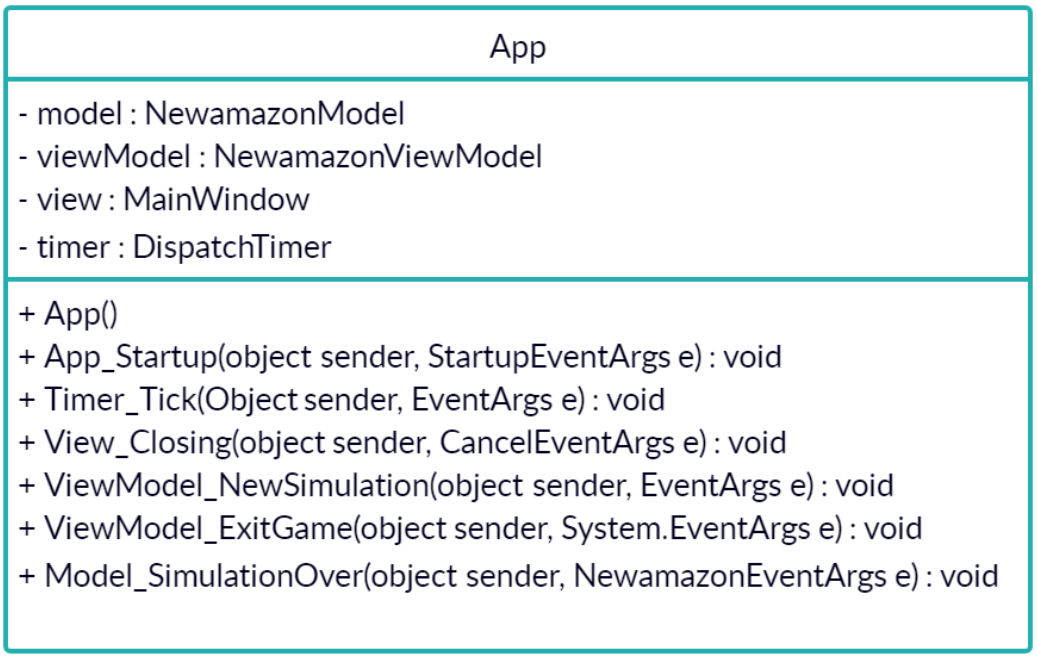
Viewmodel:

- DelegateCommand.cs
- NewamazonField.cs
- NewamazonViewModel.cs
- ViewModelBase.cs



App:

- App.xaml.cs



NewamazonTest:

- NewamazonModelTest.cs

Osztályok kifejtése

Model:

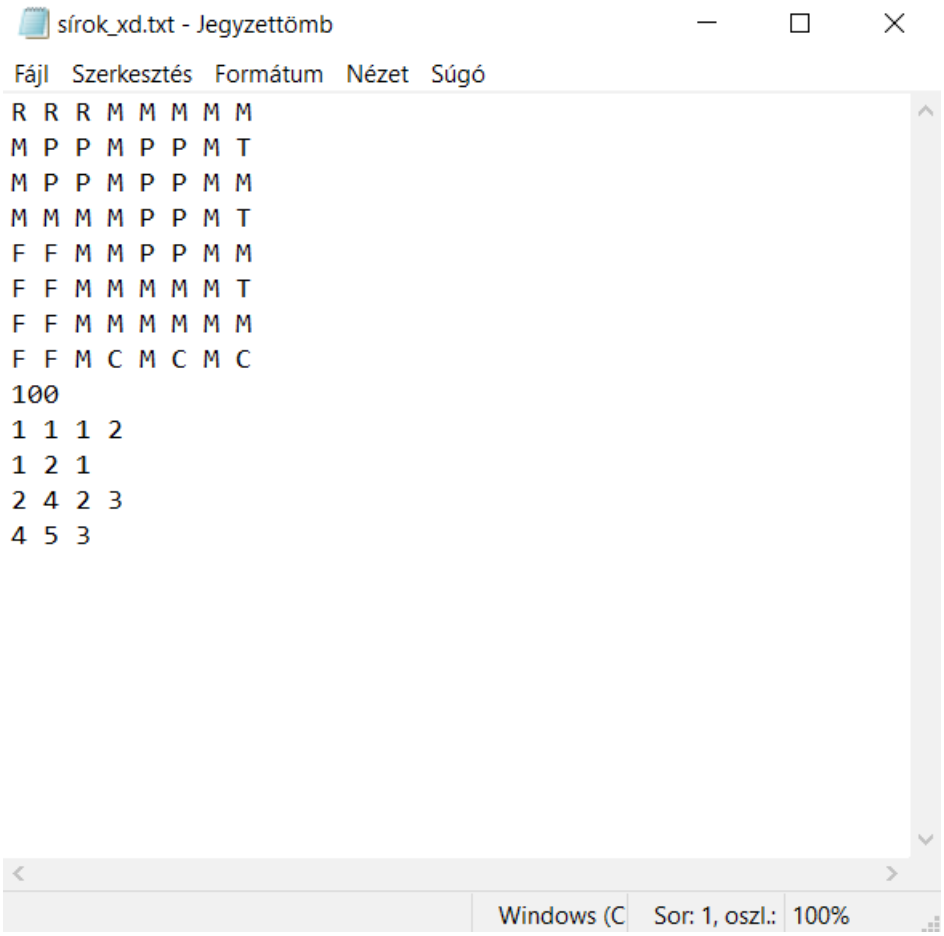
- **NewamazonModel.cs:**
  - A központot tartalmazó osztály. A nézetmodellel eventek segítségével kommunikál. Számolja a lépések számát, nézi, hogy az egyes robotok mennyi energiát fogyasztott, illetve az összes energia fogyasztást.
- **NewamazonEventArgs.cs:**
  - Az eventek példányosítására létrehozott osztály, az EventArgs osztályból származik.
- **NewamazonClass.cs:**
  - A mezők osztályainak a base osztálya. Tartalmazza többek között az adott mező koordinátáit és IDját.
- **Mezo.cs**
  - A mezők példányosítására létrehozott osztály, a NewamazonClass osztályból származik. Tartalmazza, hogy éppen van-e rajta robot.
- **Polc.cs:**
  - A polcok példányosítására létrehozott osztály, a NewamazonClass osztályból származik. Tartalmazza, hogy az adott polcnak éppen melyik célállomáshoz kell menni, hogy van-e

alatta robot és hogy hol a helye. Ha éppen nincs a helyén (egy robot viszi éppen), akkor mozog a robottal.

- **Robot.cs:**
  - A robotok példányosítására létrehozott osztály, a NewamazonClass osztályból származik. Tartalmazza az adott robot üzemanyagát, irányát és hogy van-e rajta szekrény.
- **Toltoallomas.cs:**
  - A töltőállomások példányosítására létrehozott osztály, a NewamazonClass osztályból származik. Tartalmazza, hogy foglalt-e vagy sem.
- **Celallomas.cs**
  - A célállomások példányosítására létrehozott osztály, a NewamazonClass osztályból származik. Tartalmazza, hogy foglalt-e vagy sem.
- **Kozpont.cs:**
  - A központ osztálya. Ez felel a robotok irányításáért, ő találja ki a robotok útját, illetve tárolja is azokat. A potenciális ütközéseket észreveszi, és igyekszik ilyenkor a lehető legkevesebb időveszteséggel megoldani azokat.
- **AllData.cs:**
  - A szimuláció fájlból való betöltéshez szükséges osztály, tartalmazza a szükséges kezdőadatokat (például a mezők kezdőpozícióit.)
- **NewamazonDataExcepion.cs:**
  - Az adatelérési kivétel példányosítására létrehozott osztály.
- **NewamazonFileDataAccess.cs:**
  - A szimuláció betöltéséért felelő osztály, az INewamazonDataAccess.cs osztályból származik.
- **INewamazonDataAccess.cs:**
  - Ebből az osztályból származik a NewamazonFileDataAccess.cs
- **MainWindow.xaml.cs:**
  - A szimuláció ablaka.
- **DelegateCommand.cs:**
  - Az általános parancs típusa.
- **NewamazonField.cs:**
  - Általános mező entitás típus. A játékmező számára biztosított osztály, amely eltárolja a pozíciót, szöveget, színt, irányt (robot esetén szükséges), stb. A mezőket egy felügyelt gyűjteménybe helyezzük a nézetmodellbe.

- **NewamazonViewModel.cs:**
  - A nézetmodell osztálya, a nézet tulajdonságait kezeli. Parancsokat biztosít az új játék kezdéséhez, játék betöltéséhez, mentéséhez, valamint a kilépéshez. A parancsokhoz események vannak kötve, amelyek a parancs lefutását jelzik a vezérlőnek. A nézetmodell tárolja a modell egy hivatkozását, de csupán információkat kér le tőle, illetve a játéknehézséget szabályozza. Direkt nem avatkozik a játék futtatásába.
- **ViewModelBase.cs:**
  - A nézetmodell base osztálya.
- **App.xaml.cs:**
  - 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. Továbbá az időzítők is itt futnak.
- **NewamazonModelTest.cs:**
  - Osztály a szimuláció helyességének tesztelésére.

Mintabemenet:



R: Robot

M: Üres mező

P: Polc

T: Töltőállomás

F: Fal

C: Célállomás

Ezután az első sorban a robotok kezdőüzemanyaga, majd ezután a megfelelő koordinátán lévő polcokon lévő áruk célállomása. (Pl: 2 4 2 3: 2. sor, 4. oszlop, 2es és 3as célállomás)