Esemény vezérelt alkalmazások beadandó dokumentáció

Késztette:

Hallgató: Gadácsi Brendon
Neptun kód: F2ZU17
E-mail: gadacsi.b@gmail.com

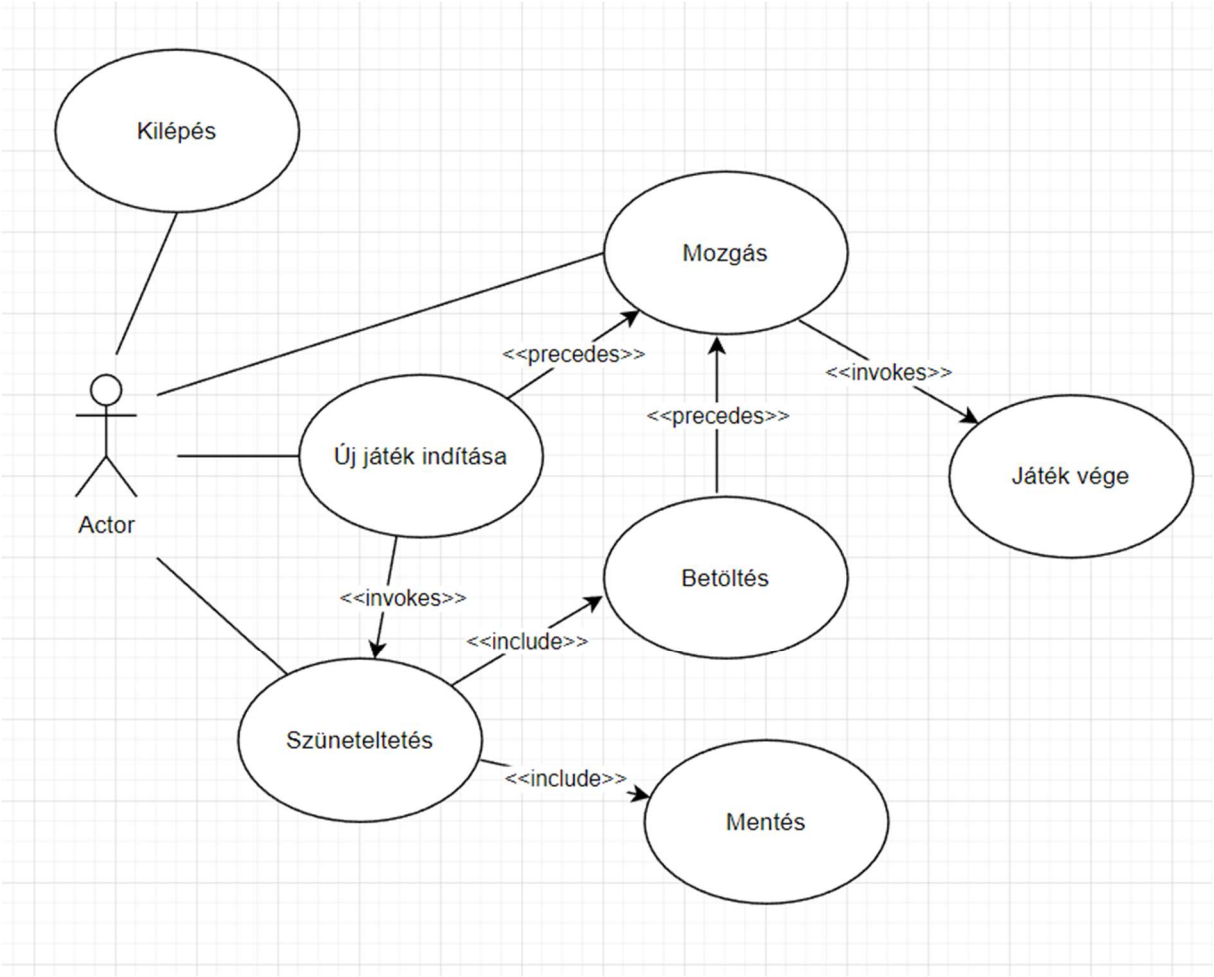
Feladat:

Aknamező Készítsünk programot a következő játékra. A játékban egy tengeralattjárót kell irányítanunk a képernyőn (balra, jobbra, fel, illetve le), amely felett ellenséges hajók köröznek, és folyamatosan aknákat dobnak a tengerbe. Az aknáknak három típusa van (könnyű, közepes, nehéz), amely meghatározza, hogy milyen gyorsan süllyednek a vízben (minél nehezebb, annál gyorsabban). Az aknákat véletlenszerűen dobják a tengerbe, ám mivel a hajóskapitányok egyre türelmetlenebbek, egyre gyorsabban kerül egyre több akna a vízbe. A játékos célja az, hogy minél tovább elkerülje az aknákat. A játék addig tart, ameddig a tengeralattjárót el nem találta egy akna. A program biztosítson lehetőséget új játék kezdésére, valamint játék szüneteltetésére (ekkor nem telik az idő, és nem mozog semmi a játékban). Ismerje fel, ha vége a játéknak, és jelenítse meg, mennyi volt a játékidő. Ezen felül szüneteltetés alatt legyen lehetőség a játék elmentésére, valamint betöltésére.

Elemzés:

- Legyen egy irányítható tengeralattjárónk, legyenek hajóink, amik aknákat dobálnak a vízbe. A játék indításkor automatikusan elindul, a hajók véletlenszerű pozícióval, a tengeralattjáró pedig középen.
- A feladatot WPF grafikus felülettel valósítjuk meg.
- Az ablakon elhelyezünk egy menüt a következő menüpontokkal: New game, Load Game, Save Game. A load és a save gomb alapértelmezetten disabled állapotban van. Az ablak alján megjelenítünk egy státuszsort, amely az eltelt időt mutatja.
- A játéktér a window középső területe, amelynek a tetején az előre legenerált hajók mozognak. Ez a játékos számára elérhetetlen. A tengeralattjáró a hajók alatt, és a státuszbar közötti területen tud mozogni. A hajók erre a területre dobják az aknáikat. Irányítani a w,a,s,d gombok segítségével lehet.
- A játékot a space gomb segítségével lehet szünetelteni. Ilyenkor megáll az időszámlálás, és a load és save gombok enabled állapotba kapcsolnak. Ilyenkor van lehetőség a játékállást menteni, illetve betölteni. A mentést és a betöltést dialógus segítségével végezzük el. A betöltést esetén azonnal a betöltött állapot rajzolódik ki a windowon szüneteltetett állapotban. A file neveket a felhasználó adja meg.
- Új játékot kezdeni bármikor van lehetőség. Ilyenkor eltűnnek az aknák, újragenerálódnak a hajók és a tengeralattjáró is visszakerül középre.
- Amikor vége a játéknak, azaz a hajót eltalálja egy akna, azt egy felugró ablak jelzi, rajta a játék idejével. Bezárva azonnal egy új játék kezdődik.

Esemény vezérelt alkalmazások beadandó dokumentáció



1. ábra Felhasználói esetek

Tervezés:

Programszerkezet:

A programot négyrétegű architektúrában valósítjuk meg. A megjelenítés a View, a nézet logikája a ViewModel, a modell a Model, míg a perzisztencia a Persistence névtérben helyezkedik el.

Perzisztencia:

- Az interfészt szöveges fájl alapú adatkezelésre a Fileman osztály valósítja meg, ami megvalósítja az IFileman interfacet. A fájlkezelés során fellépő hibákat a DataAccessException kivétel jelzi.
- A program az adatokat egyszerű szöveges (.txt) fájlokban tudja eltárolni. Ezeket az adatokat szüneteltetés során lehet betölteni, illetve ilyenkor biztosít lehetőséget mentésre is a játék.
- A fájlban egy-egy sor, egy objektumot reprezentál. Az első karaktere mondja meg, hogy az milyen típusú objektum, a sor többi adata pedig az adattagok értékei. A sor hossza változó attól függően, hogy milyen objektumot reprezentál. Az adatok ';'-vel vannak elválasztva egymástól.

Esemény vezérelt alkalmazások beadandó dokumentáció

Modell:

- A modell több különböző osztályból áll, mindegyik a saját logikájáért felel. Ezeket a MainModel osztály fogja össze, aki a viewmodellel való kommunikáció nagyrészét kezeli. Ő adja tovább az adatokat a különböző objektumoknak, amik alapján legenerálódhatnak. Az objektumokat publikus adattagként tárolja el, amiből több is lehet, azokat listában. Ő kezdeményezi az aknák létrehozását. Az objektumokat újra tudja generálni és megállítani a mozgásukat. Ő végzi el az ellenörzést, hogy a játkost eltalálta e már egy akna. Ezen felül még a hajók erősítéséért is ő felel (Minden hajóra meghívja).
- A SubmarineModel osztály a játékos mozgását reprezentálja.
- A ShipModel osztály a hajókat kezeli. Mindegyik hajó a saját mozgásáért és az aknák ledobásáért felel. A hajó erősítésének a lényegi részét ő végzi el.
- A MineModel osztály az aknákat reprezentálja. Ő a saját mozgáért felel, és jelez, hogyha a játékos számára már nemlátható helyre ér, hogy el lehessen tüntetni.

Nézet Modell

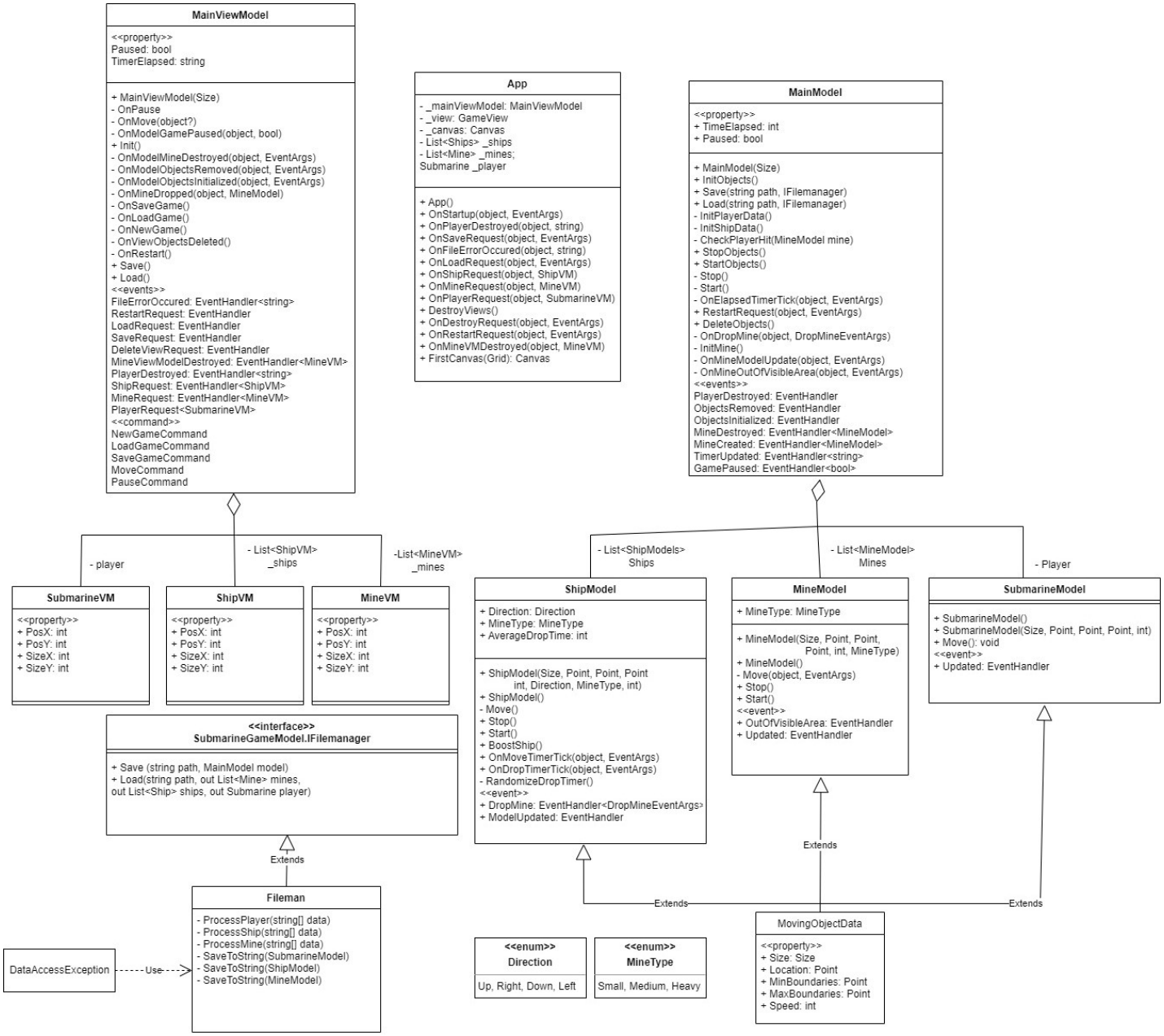
- A nézetmodell megvalósításához felhasználunk öt darab általános utasítást, valamint egy ős változásjelzőt
- A nézetmodell fő feladatait a MainViewModel osztály látja el. A többi, kisebb viewmodel a saját modeljével kapcsolatos feladatokat látja el, mint például a model mozgásának nyilvántartását. A MainViewModel lehetőséget biztosít új jták kezdésére, játék betöltésére, játék mentésére, illetve a játékos mozgatására és a szüneteltetésre. A parancsokhoz eseményeket kötünk, ami jelzi a parancs lefutását a vezérlőnek. A nézetmodell tárolja a mainmodel egy példányát, de csak információkat kér tőle, vagy esemény hatására az állapotának megváltoztatását kéri tőle.

Nézet

- Több megjelenítő van, a fő megjelenítő a GameWindow. Ez egy menusorból, egy canvasből és egy status barból áll. Az objektumok a canvasen rajzolódnak ki, ahol adatkötés segítségével jelenítjük meg az objektumok nézeteit, és pozicióit.

Környezet

- Az app osztály feladata a rétegek példányosítása és összekötése, a nézetmodell és a modell eseményeinek kezelése, ez által a felület szabályozása. A dinamikusan létrejövő objektumot létrehozását és törlését is ő végzi.



2. ábra Az alkalmazás osztálydiagrammja

Esemény vezérelt alkalmazások beadandó dokumentáció

Tesztelés:

A modell funkcionalitása egységtesztekkel lett ellenőrizve a Test projekt osztályaiban.
Az alábbi tesztesetek kerültek megvalósítása:

TestMainModel:

- TestCtor: Megfelelően létrehozza a szükséges objektumokat
- TestInit: Létrehozza a kezdő objektumokat
- TestDeleteObjects: Kitörli e a szükséges objektumokat
- TestLoad: Betöltés lefut e

TestShipModel:

- TestCtor: Megfelelő helyre generálja e az objektumot
- Boost: Boostolja e a hajót

TestSubmarineModel:

- TestCtor: Megfelelő helyre és megfelelő mérettel generálja e le
- TestMove:
    - Megfelelően mozog e a különböző irányokba.
    - Széleken való mozgás