Programozás alapjai 3
Házi feladat dokumentáció

Nánási Dániel
C1NRWP

2020. november 9.

1. Megvalósított játék lényege

A játék egy hajócsata szimuláción alapul, egy játékos játsza a gép ellen. Mindkét félnek van egy-egy véges nyílt vize (10x10-es pálya), ahova a hajóikat pozícionálhatják. Kezdésnek mindkét játékos lerakhat valamennyi különböző méretű hajót, mindkét játékos ugyanazokat (egy 2, egy 3, egy négy és egy 5 méretűt) a pályán úgy hogy a pálya szélétől és a másik hajótól legalább 1 távolságra van, lehet vertikálisan és horizántálisan is. Utána körökre osztva tüzelhetnek egymásra, értesülnek a találatról és a süllyesztésről (akkor süllyedt, ha a hajó összes részét, ha 5 hosszú, akkor mind az 5 koordinátáját, tüzelés érte). Az nyer, aki hamarabb elsüllyeszti a másik játékos flottáját.

2. Use-case

2.1. Főmenü

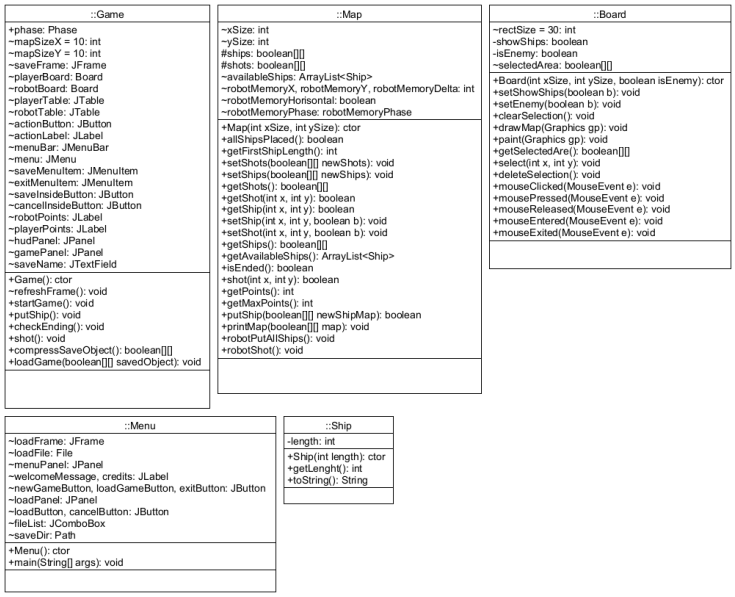
A program indításánál grafikus egy 3 opciós grafikus menü jelenik meg:

- Új játék indítása
  Új ablak jön fel a játék megkezdésével.
- Játék betöltése
  Új ablak jön fel, ahol a savedgames mappa tartalmából lehet egyet választani vagy visszalépni a főmenübe.
- Kilépés Kilép a programból

2.2. In-game GUI

A játékmenet alatt egy ablak látszódik amely 3 konténerre bontható fel:

- Menü
  - M entés gomb: felhoz egy új ablakot, ahol a játémnetés nevét tudjuk megadni.
  - K ilépés gomb: kilép a játékból.
- Robot pályája: Az ellenfél térképe ahol ki tudunk jelölni (pirossal) és a kijelölés után az akciógombbal tudunk tüzelni a térképen kijelölt pontra.
- Akciógomb: Ennek a funkciója változik, ezzel indítjuk a játékot, ezzel rakjuk le a hajókat, ezzel tüzelünk és ezzel léphetünk ki a végén.
- Akció label: Az aktuális feladatot/információt közli a játékossal, pl. talált-e a lövés, milyen hajókat lehet még lehelyezni vagy szabályos volt-e a hajó lehelyezése.



1. ábra. Osztályok

- Pontozó labelek: a játékos és a robot pontjait jelzi ki.
- Játékos táblája: Hajók lerakásához kijelölünk négyzeteket, majd lerakjuk az akciógombbal.

A lövéseket mindkét táblán X-el látjuk lefedve, a hajókat pedig sárga kitöltéssel. Az ellenfél tábláján csak a játék megnyerése/elvesztése után láthatóak a hajók.

2.3. Játék kezdete

Ugyanannyi és ugyanakkora hajója van a játékosnak és a botnak. A bot ezeket automatikusan random leteszi magának. A játékos egyesével kijelöli azokat a négyzeteket, ahova a hajót szeretné letenni, majd rányom a lerak gombra. Ha a hajót megfelelően rakta le (van olyan mérretű hajója), akkor lerakta, egyébként figyelmezteti, hogy nincs ilyen hajója.

2.4. Játék

A játékmenet a hajók lerakása után körökre osztott. A játékos és a bot is tud lőni egyet-egyet. Értesülnek a találatról. A robot taktikája random, addig amíg el nem talált egy hajót, utána süllyedésig azt a hajót lövi körbe, keresvén a hajótesteket. Az nyer aki hamarabb elsüllyeszti minden hajóját az ellenségnek.

2.5. A játék vége

Ha minden hajórész találatot ért az egyik térképen, akkor megmutatja a robot térképét és kiírja, hogy a játékosnak mi lett a kimenetele a játékban.

2.6. Irányítás

A játékos kijelölni tud négyzeteket és a két pálya közötti akciógombra tud kattintani.

3. Megvalósítás

3.1. Változtatások a speicifikációhoz képest

- JTable to Graphics: A specifikációra review-ként azt a kérést kaptam, hogy ne JTable-el oldjam meg a táblát, hanem Graphics-al. Így az eredetileg AbstractTable modellből leszármoztatott Map osztályt kiegészítettem egy Board osztállyal, ami megvalósítja a Graphics megoldásait.
- InGame UI: A pontos elrendezés próbálgatás közben derült ki hogyan kényelmes/szép. A mentés és kilépés gombot utólag vittem fel szabványos menübe, mivel ez követelménye a házinak.

3.2. Menu osztály

Megvalósítja a játék belépési pontját a főmenüt, három gomb és két szöveg található rajta, a gombok:

- Új játék: Teljesen új játék, példányosítja a Game osztályt
- Játék betöltése: Új JFrame ugrik fel és a munkakönyvtárban található savedGames mappa tartalma jelenik meg JComboBox-ban és egy betöltés, illetve mégse gomb.. Kiválasztás után beolvassa a kiválasztott bináris adatállományt, majd ugyanúgy példányosítja a Game osztályt, ahova betölti a bináris adatokat, illetve beállítja a játék fázisát.
- Kilépés: Bezárja a programot.

3.3. Game osztály

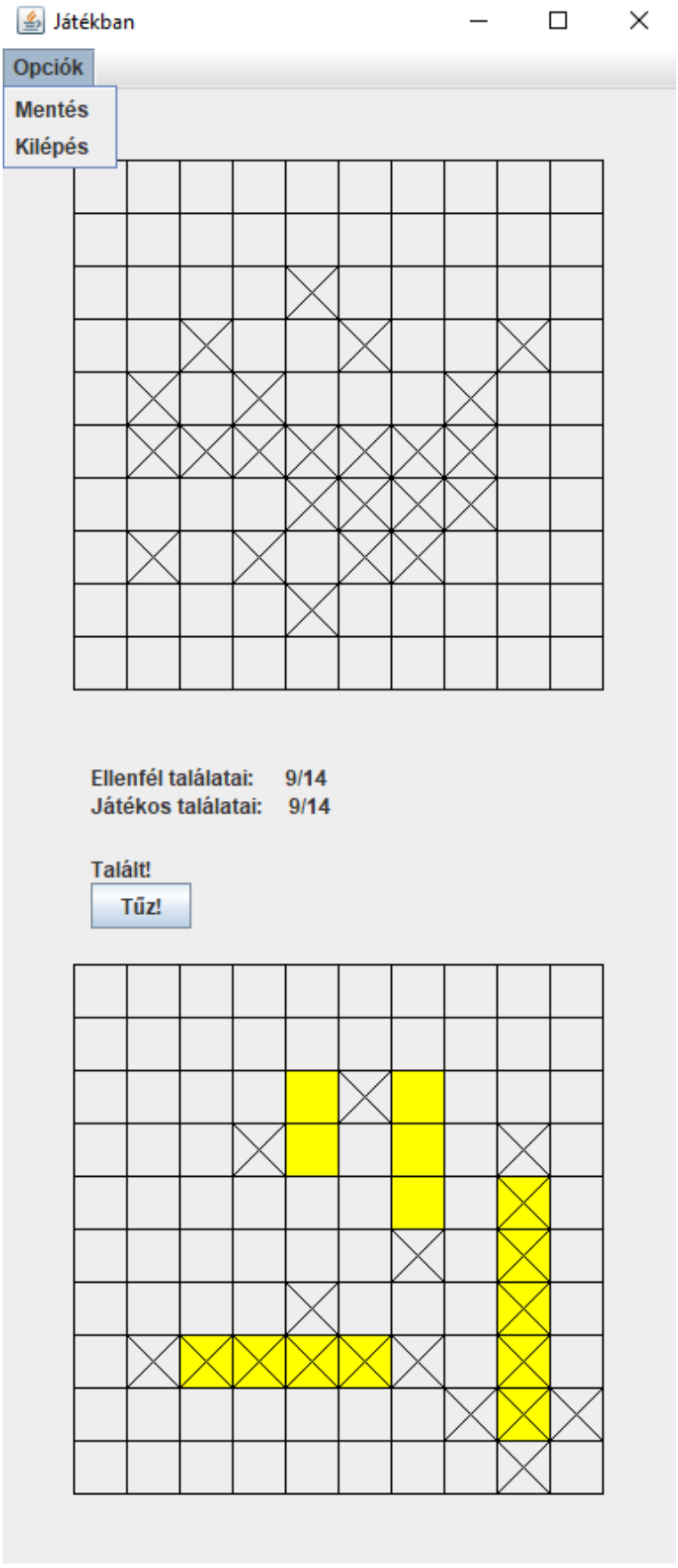
JFrame-ből származtatott a Game osztály is, tehát új játék indításnál feljön egy ablak a játéknak. Az ablak elemeit a 2.2. számú pontban ismertettem. Illetve grafikus megvalósítása a **??**. számú ábrán látható.

A Game osztály felelős a játék fázisainak kezelésért, a játkpályák frissítéséért, a user interakciók kezeléséért, az akciógombok és label-ek menedzseléséért. Az akciógomb és az akciólabel a képernyő közepén található és a feliratuk folyamatosan változik a játék fázisaitól függően. A játék fázisai:

- STARTING: A játék kezdete, a pályán kijelölhetünk, de nincs következménye és a Start gombbal tudunk kezedeni.
- PUTSHIP: Amikor hajókat rakunk le, addig tart, amíg a játékos le nem rakja az összes hajóját (a robot egyből megteszi ezt).
- FIRE: Az a szakasz, amikor a robot és a játékos felváltva tüzel egymásra.
- FINISHED: Akkor érünk át ebbe a szakaszba, amikor valamelyik pályán az összes hajó pozícióján van lövés is.

Főbb metódusai:

- Konstruktor: beállítja az ablakot, létrehozza a JButton, JMenu elemeket, példányosítja 2x a Board osztályt, amit szintén hozzáad az oldalhoz.
- refreshFrame(): frissíti a térképeket és a kijelzőket
- startGame: Start (akciógomb) lenyomására beállítja a játék indításához szükséges feliratokat és lerakatja a robot hajóit.
- putShip(): Lerak (akciógomb) hatására ellenőrizteti, hogy a játékos által kijelölt kockák lerakhatóak-e hajóként, ha igen, akkor lerakja és értesíti a játékost, hogy sikerült-e illetve milyen hajókat tud még lerakni.
- checkEnding() tüzelés után megvizsgálja, hogy nyert-e valaki



2. ábra. Játékmenet ablaka

- shot(): Tűz (akciógomb) lenyomásával, beolvassa a játékos kijelöléseit a robot tábláján és ha sikeres a lövés (1 négyzet van kijelölve), akkor a robotot is utasítja egy lövésre.
- compressSaveObject(): Mentésnél a mentendő paramétereket tömbösíti
- loadGame(): A tömbösített objektumot betölti játékállásnak.

3.4. Board osztály

A Map osztályt egészíti ki, megvalósítja a MouseListener interfészt. A Map osztály kiegészíti a JPanelt, így a Board lehelyezhető a Game osztály ablakában. Tehát egy panelt valósít meg, amire lehet kattintani.

Főbb metódusai:

- drawMap(): lerajzolja a négyzetrácsokat
- clearSelection(): letisztítja a panelt, újrarajzoltatja a négyzetrácsokat, újrarajzolja a hajókat és a lövéseket
- select(x,y): A figyelt egérkattintás hatására hívódik meg és a térképre projektált koordinátákat menti és jeleníti meg, mint kijelölés.
- deleteSelection(): törli a kijelölés mentett elemeit.
- mouseClicked(): Board-ra kattintés esetén hívódik meg, ellenőrzi, hogy térképen belül kattintottunk és projektálja a kattintás koordinátáját a 10x10-es térképre, majd meghívja a select metódust.

3.5. Map osztály

10x10-es pályán tárolja a hajók pozícióját adatait és a találatokat, kezeli a lövéseket és a hajólerakásokat.
Főbb metódusok:

- Konstruktor: inicializálja az üres térképet és elérhetővé teszi az előre definiált hajóméreteket lerakásra.
- isEnded(): Megnézi, hogy befejeződött-e a játék az adott pálya szerint, vagyis hogy ahol hajótest van, ott lövés is.
- shot(x,y): Egyszerű lövés.
- getPoints(): megszámolja a pályán megszerzett pontokat (ez az ellenfél pontja lesz).
- getMaxPoints(): visszatér a maximálisan megszerezhető pontszámmal.
- putShip(boolean[][] newShip): Egy 10x10-es boolean tömbböt vár, aminek a hajótest elemei igazak. Ellenőrzi, hogy az új hajó lerakható-e a pályára és ha igen, akkor le is rakja. Az alábbi feltételeket vizsgálja meg:

  - a pálya szélét nem érinti a hajó,

  - a hajótest elhelyezkedése alapján (vertikális/horizontális), megnézi, hogy csak egy oszlopban/sorban van hajótest,

  - a hajótest összefüggő,

  - a z új hajó összes hajóteste 1 pontos körzetében nincs még másikhajótest a pályán,

  - a pályán lerakható a kívánt méretű hajó.

- robotPutAllShips(): Lerakatja random az összes elérhető hajót. Kiválasztunk 3 random számot egy-egy koordinátát 1-9 között, illetve egy irányt 0-3 között és megpróbáljuk erre a kordinátára az adott irányba lerakni a hajót. Addig iterálunk a meglévő hajók között, amíg mindet sikerül lerakni.

- robotShot(): Lő egyet a robottal, egy egyszerű memóriát alkalmazva. 4 fáziban írható le a memória:

  - R andom lő a pályán olyan helyre ahova még nem lőtt, mindaddig amíg nem talált, ha talált elmenti a találat koordinátáit.

  - K örbelövi a találatot (bal, jobb, föl, le), addig amíg nem talál mégegyet és rögzíteni tudja, hogy a megtalált hajó vertikális vagy horizontális elhelyezkedésű.

  - A z elhelyezkedés alapján negatív irányba inkrementálisan tüzel, addig amíg van találat.

  - A z elhelyezkedés alapján pozitív irányba inkrementálisan tüzel, addig amíg van találat.

3.6. Ship

Hajó, hossza van, 2 és 5 között. Lehet lerakva és lerakatlanul

4. Mentés/Betöltés

A játék mentését a Game osztály, a betöltését pedig a Menu valósítja meg. Menteni akkor lehet, amikor már a játékos lerakta az összes hajóját. Mentésnél létrehozunk egy 10x40-es boolean[][] tömmböt, ahova projektáljuk a játékos lövéseit, a játékos hajóit, a robot lövéseit és a robot hajóit ebben a sorrendben. Ezt az boolean 2 dimenziós tömmböt mentjük el bináris fájlként a savedGames mappába a játékos által megadott néven. Ha nincs még savedGames mappa, akkor létrehozzuk. Betöltésnél beolvassuk a játékos által kiválasztott mentést és beolvasás után a 10x40-es tömbböt visszaprojektáljuk 4 db 10x10-es tömbre és betöltjük ezeket a két térképre, majd beállítjuk a játék fázisát.

5. Tesztek

Három osztályhoz készítettem unit tesztet, a Ship, a Map és a Game. A ship osztályban a konstruktor beálítását teszteltem. A Map osztályban a hajó lerakásaira írtam több esetet, kizárván az összes lehetséges hibás hajólerakási lehetőséget. Illetve a pontozó számlálóhoz és a sikeres hajólerakáshoz írtam két tesztet a Map osztályban. A Game osztályban a játék mentését és visszatöltését teszteltem úgy, hogy a két térképre létrehoztam véletlenszerű hajólerakást, 10-10 véletlenszerű lövést és ezt az állást tömörítettem egy olyan objektumra amit a program ment. Ezt visszatöltöttem, majd újra mentettem és megnéztem, hogy a két mentett objektum tartalma azonos-e.