



Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar
Irányítástechnika és Informatika Tanszék

Beágyazott rendszerek szoftvertechnológiája fejlesztői és felhasználói dokumentáció

Torpedó

Csapattagok:

Regőczy Tamás (regoczy.tamas@edu.bme.hu)

Balogh Dániel (balogh@edu.bme.hu)

Bellyei Boldizsár (bellyeib@edu.bme.hu)

Konzulens:

Naszály Gábor (naszaly.gabor@vik.bme.hu)

2021. május

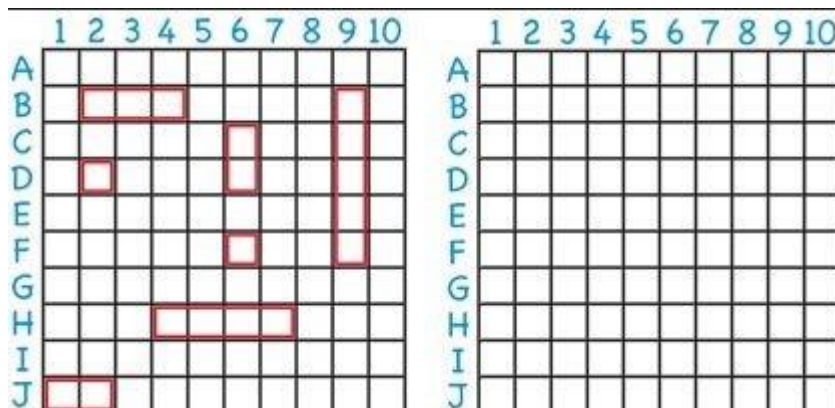
Tartalom

Játékszabályok	3
1. A játék neve: Torpedó	3
2. Előkészületek	3
3. A játék menete	3
4. A játék vége	3
5. Az elhelyezendő hajók és a pálya nagysága	4
A játék folyamata	5
6. Kezdőképernyő	5
7. Játék indítása	6
8. Előkészületek	7
9. A játék menete	8
10. A játék vége	9
Tervezés	10
1. Osztálydiagramm	10
2. Megvalósítandó osztályok definíciója	11
1.1 ShipPart	11
1.2 Ship	11
1.3 MainMenu	11
1.4 Comms	11
1.5 Grid	12
2.1 MyGrid	12
1.6 GridCreator	12
1.7 Main	12
2.2 Importált osztályok	12
A megvalósítás során felmerült nehézségek, tapasztalatok	13

Játékszabályok

1. A játék neve: Torpedó

A játékot egyszerre két játékos játssza, egymás ellen online formában (két külön számítógépről). A játékosok előtt két-két 10x10 cellából álló, négyzetalapú tábla helyezkedik el. Az egyik tábla (a továbbiakban „saját pálya”) a játékos fenségvizét mutatja, amelyeken a saját hajóit látja. A másik tábla (továbbiakban „ellenfél pálya”) az ellenfél játékos fenségvizét mutatja, de az ellenfél hajóit nem láthatja rajta (ld. 1. ábra). A játék során a két játékos felváltva egy-egy lövéssel próbálja eltalálni az ellenfél hajóit, a játék célja elsüllyeszteni az ellenfél összes hajóját.



1. ábra – Saját pálya és ellenfél pálya

2. Előkészületek

Mindkét játékos azonos számú és méretű hajókkal rendelkezik. A játék kezdetekor mindkét játékos elhelyezi a hajóit a saját pályáján: egy adott hajó vízszintesen és függőlegesen helyezhető el (a cellákon keresztbe nem). A hajók egymással szomszédos cellákon is elhelyezhetők, azonban egymáson nem (egy adott cellán legfeljebb egy hajó tartózkodhat). Miután mindkét játékos elkészült, megkezdődik a csata. A hajók ezek után már nem mozgathatóak.

3. A játék menete

A játékot a host kezdi. A játék körökből áll, a körben soron lévő játékos megjelöl (meglő) pontosan egy üres cellát az ellenfél pályáján, ekkor a következők történhetnek:

1. Ha a megjelölt cellán egy ellenséges hajó található, akkor az adott hajó találatot kapott.
2. Ha a megjelölt cellán nem található ellenséges hajó, akkor a lövés mellé ment.

Az a cella, amelyre már lőttek megjelölésre kerül (attól függően, hogy mellé vagy találat más-más jelöléssel): a támadó játékosnak az „ellenfél pályán”, míg a nem támadónak a „saját pályán”. A megjelölt cellákra többet nem lehet lőni. Egy hajó akkor süllyed el, ha az összes általa lefedett cella találatot kapott.

4. A játék vége

A játék akkor ér véget, ha valamelyik játékos összes hajója elsüllyedt. Ekkor az a játékos nyer, akinek még van el nem süllyedt hajója.

5. Az elhelyezendő hajók és a pálya nagysága

A pálya nagysága jellemzően 100 cella (10x10). A pálya sorai betűkkel, oszlopai számokkal vannak azonosítva, így minden cella egyértelműen megadható egy betű és egy szám kombinációjával. A pálya nagysága implementáció függő.

Az egyes hajótípusok darabszáma szintén implementáció függő. 4 féle hajó szerepelhet a játékban. Egy általános elosztást 100 cella esetén az alábbi táblázat szemléltet (ld. 1. Táblázat).

A hajók száma és mérete a programkódban konstans, de tetszőlegesen változtatható: a játék az elvártak szerint ugyanúgy működik.

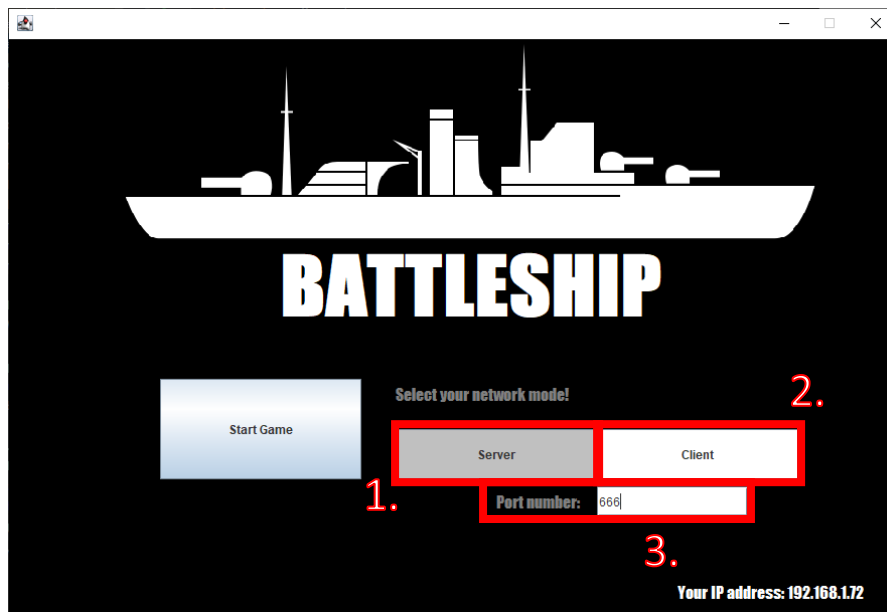
100 cella	
<i>Anyahajó (4)</i>	1 db
<i>Csatahajó (3)</i>	1 db
<i>Tengeralattjáró (2)</i>	1 db
<i>Járőr hajó (1)</i>	1 db
<i>Összes hajó</i>	4 db
<i>Összes elfoglalt cella</i>	10
<i>Elfoglalt cellák aránya</i>	10%

1. Táblázat – Egy általános felosztása a hajóknak

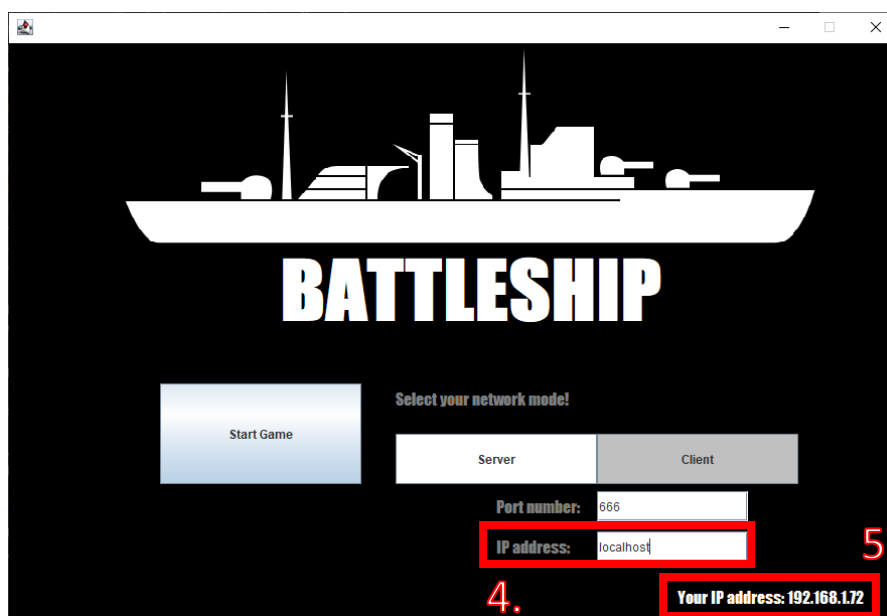
A játék folyamata

6. Kezdőképernyő

A játék megnyitása után az első ablakon választhatja ki a felhasználó, hogy a gazdája (*Server (1)*) (továbbiakban host) akar lenni egy meccsnek, vagy más felhasználó által indított csatában akar részt venni (*Client (2)*). Ehhez rendre a *Server (1)* vagy *Client (2)* gombokra kell kattintani. Minden esetben meg kell adni a *Port number (3)* azaz *port számot*, amelyen keresztül a két ellenfél applikációi megtalálják egymást.

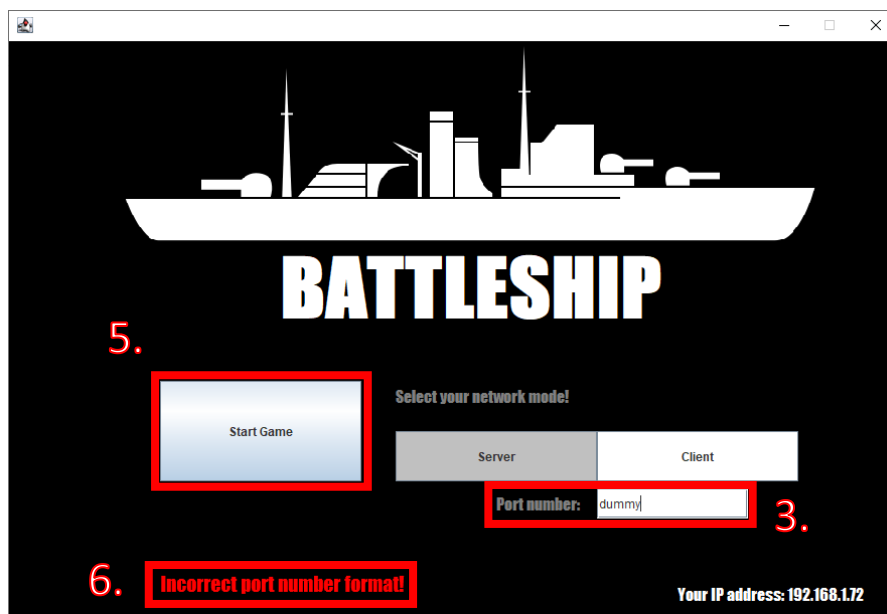


A *Client (2)* esetében megjelenik egy új mező, az *IP address (4)*, melyben a host IP címét kell megadni. Ez a string lehet egy IPv4 cím, vagy például a „localhost” alias. Alap értéke a helyi hálózat. Minden játékos a saját IP címét *your IP address (5)* néven feltüntetve, a jobb alsó sarokban találja meg.



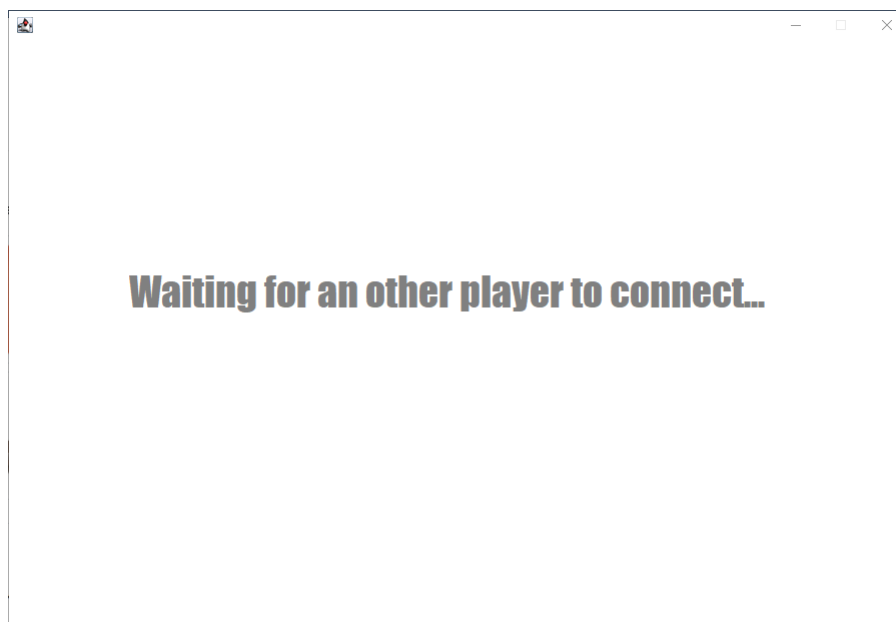
7. Játék indítása

A meccskeresést értelemszerűen annak gazdája kezdeményezheti a *Start Game* (5) gombra kattintva. Amennyiben nem megfelelően adtuk meg a port számot (3), a játék figyelmeztet (6) erre.



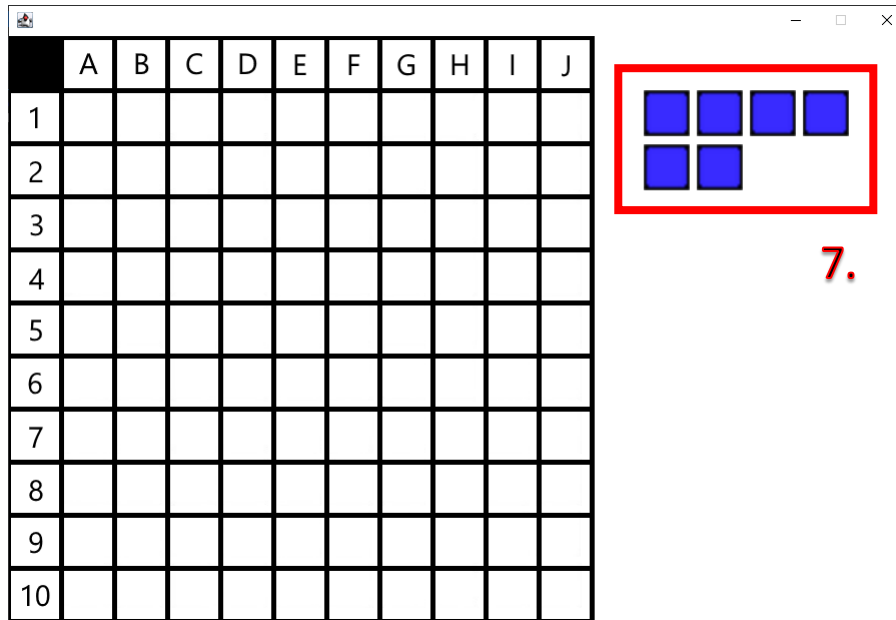
A játék indításával (5) a résztvevők megkezdhetik az előkészületeket a csatára.

Az előkészületek után, a vendég addig nem tud csatlakozni, amíg a játékgazda el nem kezdi a meccskeresést. Ezalatt a host előtt egy töltőképernyő jelenik meg.

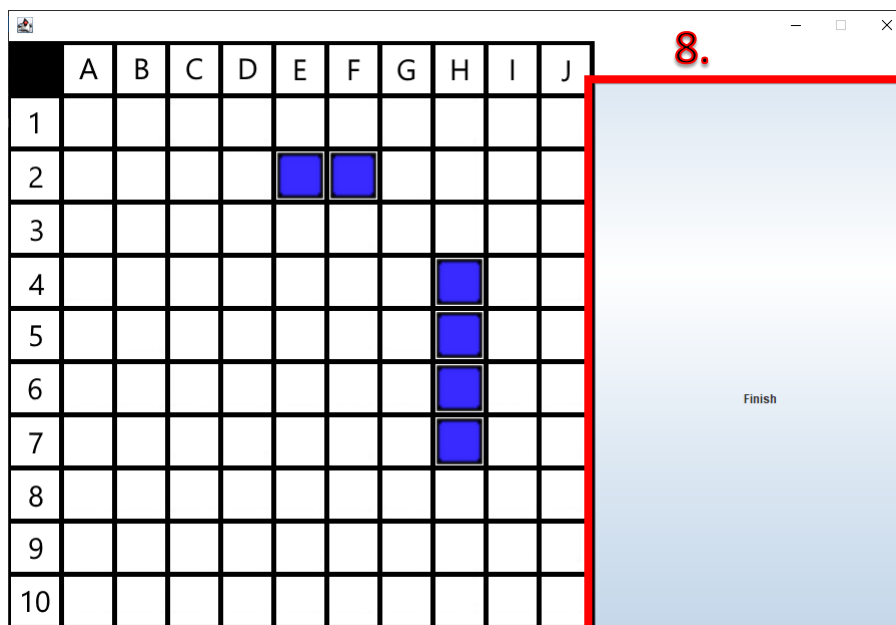


8. Előkészületek

A játékosok felhelyezik *csatahajóikat* (7) a térképükre, melyek a jobb oldali sávban találhatók meg. Ezt Drag and Dropp eljárással tehetik, azaz kattintva, tartva és húzva majd elengedve az objektumokat. A vertikális és horizontális irányú elhelyezés a jobb egérgomb kattintásával változtatható.



Miután ezzel megvannak, a *Finish* (8) gombra kattintva kezdhetik meg a meccskeresést.






9. A játék menete




A kezdő játékos a host.

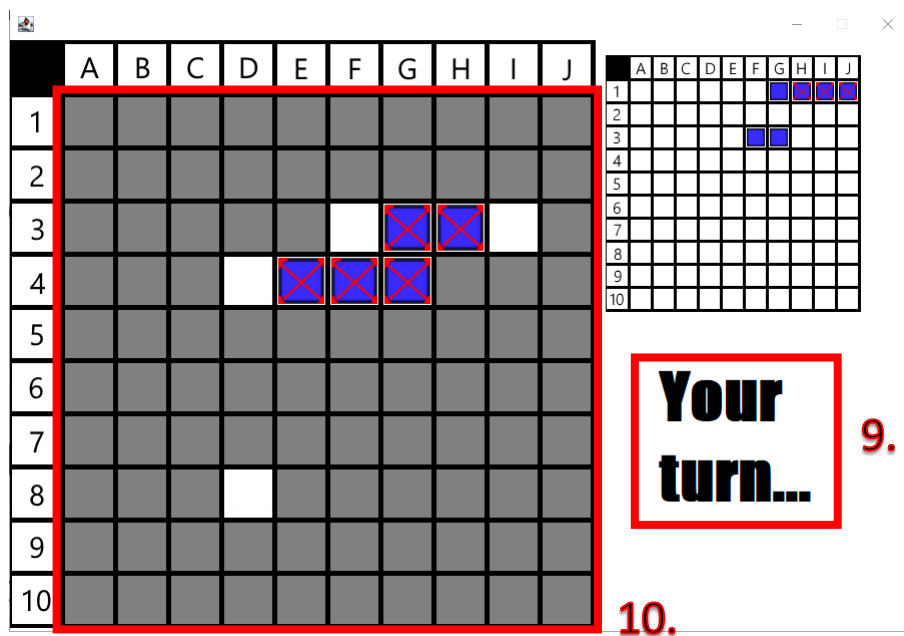
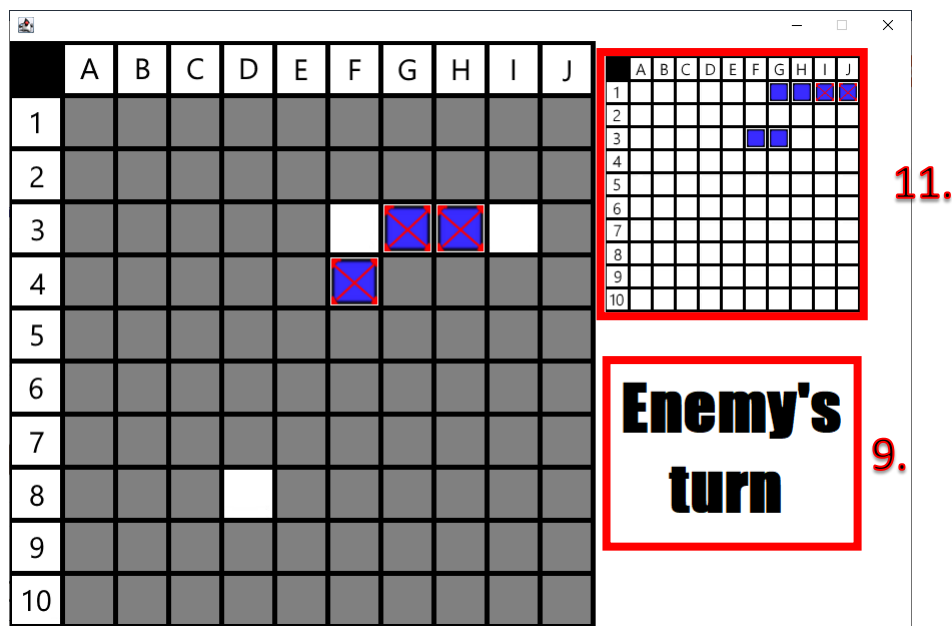
Egy *indikátor (9)* a jobb oldalon mutatja, hogy éppen ki a soron következő játékos.

Mindenki a saját körében tippel egy mezőt az *ellenfél térképén (10)*.

- Az ismeretlen mezők sötét szürkével, 
- az üres mezők fehérrel, 
- az eltalált hajórészek vörösen áthúzott kékkel jelöltek. 

Miközben végig látható a *saját térképünk (11)* és hajóink állapota.

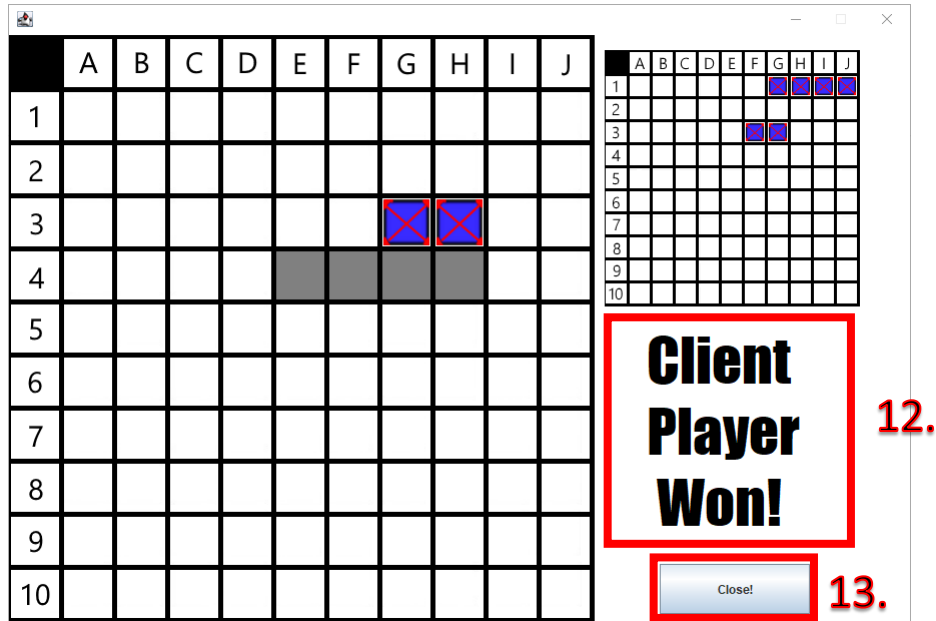
- A sértetlen hajórészek kékkel, 
- az üres mezők fehérrel, 
- az eltalált hajórész vörösen áthúzott kékkel jelöltek. 



10. A játék vége

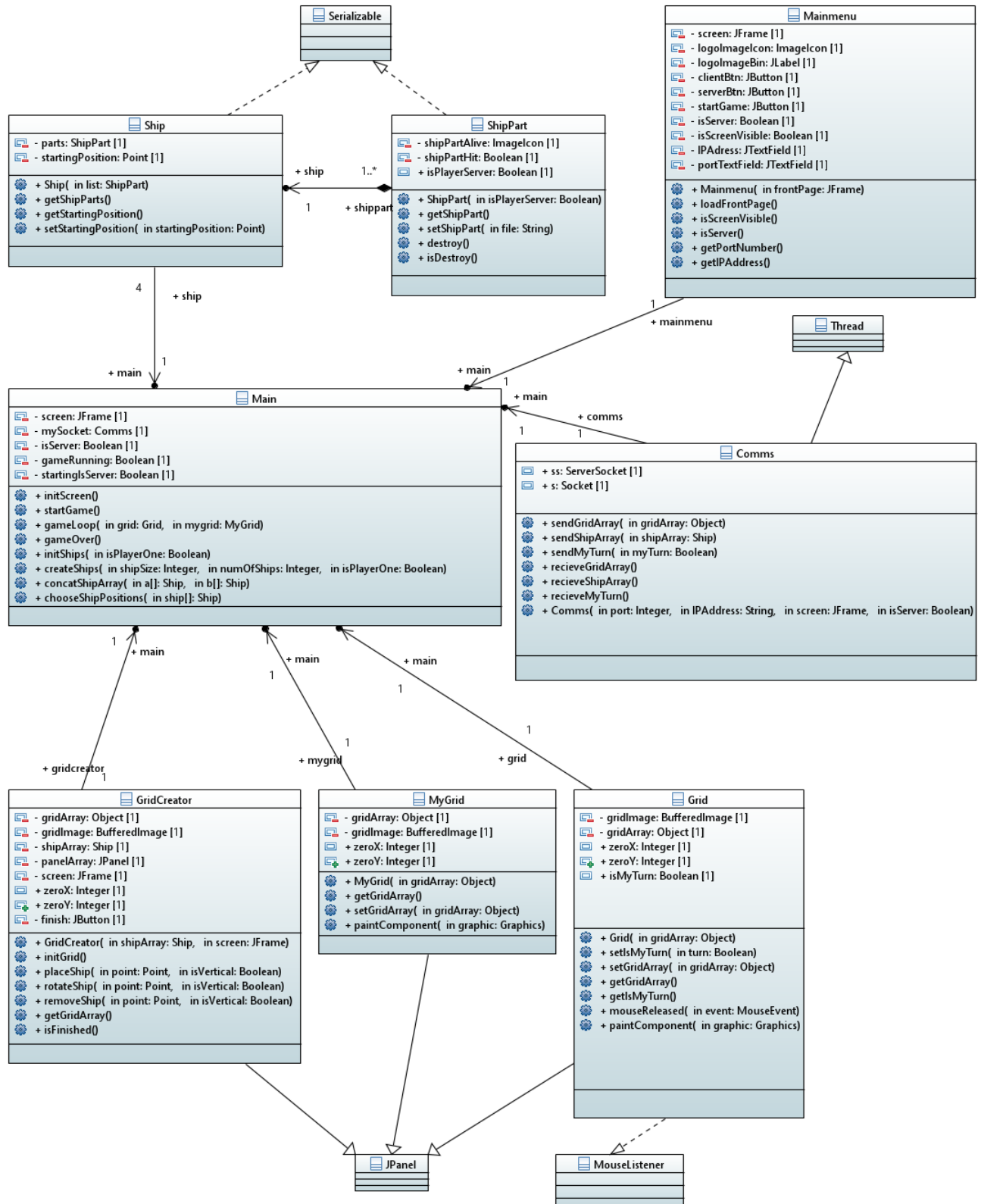
Ha az egyik játékos sikeresen elsüllyesztette a másik összes hajóját, a térképek lelepleződnek, és a *győztes játékos neve (12)* mindkét oldalon megjelenik. A szürkével jelölt mezők a nem eltalált hajórészek.

A *Close! (13)* gombra kattintva a játék bezárul.



Tervezés

1. Osztálydiagramm



2. Megvalósítandó osztályok definíciója

1.1 ShipPart

Az hajó egyes celláit reprezentáló osztály, minden a hajó által lefedett cellához egy objektum tartozik. Egyes objektumokhoz tartozó változók:

- Image: az adott cellához kapcsolódó képet tárolja
- Boolean: a cella állapotát tárolja (találatot kapott/nem kapott találatot)
- Boolean: a szerver-, kliens oldali játékoshoz tartozik-e

1.2 Ship

A Ship objektum ShipPart objektumokat tartalmaz, számuk a hajó hosszától függ. Egyes objektumokhoz tartozó változók:

- ShipPart[]: a hajóhoz tartozó ShipPart objektumokat tároló tömb
- Point: a hajó első cellájának a koordinátája (x,y)

A „ShipPart” objektumok egy úgynevezett „GridArray” változóban is el vannak tárolva, melynek típusa egy *Object[][]* java class, 2 dimenziós tömb. A *GridArray* mérete megegyezik a pálya méretével, azaz 10x10. Az egyes cellák tartalmazhatnak *int* típusú értéket, amennyiben az adott cellán nincsen hajó, illetve tartalmazhatnak *ShipPart* típusú objektumot.

Fontos megjegyezni, hogy a fejlesztés során úgy döntöttünk, hogy az összetartozó *ShipPart* objektumokat egy *Ship* objektum is tárolja, így tudjuk ellenőrizni, mely hajódarabok részei ugyanannak a hajónak. Mivel a TCP kommunikációt a fejlesztés végső fázisában adtuk hozzá a programhoz, utólag derült ki, hogy adatküldés során az objektumok közti kapcsolatok elvesznek. Ebből kifolyólag a *Ship* osztályt csak az inicializáló fázisban használjuk fel, a játék során már nem.

1.3 MainMenu

A játék indítását követően ez az osztály kerül meghívásra, funkciója a főmenü megjelenítése. A főképernyőn megjelenítésre kerül a játék logója, ezen kívül a felhasználó a Start Game gombra kattintva indíthatja el a játékot. A játék indítását megelőzően szintén ezen a képernyőn ki kell választania a játékosnak, hogy Szerver vagy Kliens gépről játszik. Utóbbi esetben a szerver gép IP címét is meg kell adnia a port szám alatt, egy arra kijelölt szövegdobozban.

1.4 Comms

A TCP kommunikációt megvalósító osztály, konstruktortól függően *ServerSocket* vagy *Socket* kommunikációs csatornát hoz létre. 3-3 függvénye van: *Ship[]*, *Object[][]*, *Boolean* változókat küldő és fogadó függvény. Inicializáláskor a *Main*-ben elküldésre kerülnek: a hajókat tartalmazó tömb, a pálya celláit tároló *Object* kétdimenziós tömb. Játék során a cellák kerülnek folytonosan elküldésre, illetve egy változó, amely jelzi, hogy tart-e még a küldő játékosnak a köre, vagy már lépett.

A kommunikáció folyamata: mindig a szerver (host) küldi el először az adatokat, a kliens pedig várakozik rá, hogy megkapja azokat. Sikeres adatküldés esetén cserélnek. Ha a fogadó fél adatot kapott, egy *Boolean* típusú „acknowledge” jelet küld vissza, amit ha a küldő fél megkap, már nem küldi tovább az adatot és tovább halad a program futása.

1.5 Grid

A játék pályáját kezelő osztály, mindig az ellenfél hajóinak megjelenítésére szolgál (ennek az objektumnak attribútuma a korábban említett, az ellenséghez tartozó *Object[][]* GridArray). A hozzá tartozó pálya grafika egy előre létrehozott kép. Az egyes cellákhoz tartozó inputot a MouseListener osztályból való származtatás segítségével definiáljuk, a grafikus felületet és a pálya elrendezését a JPanel felhasználásával kezeljük. Az osztályhoz tartoznak rögzített koordináták, képbuffer, és egy Boolean típus, hogy éppen melyik játékos van soron. Mivel az egérrel való kattintás felül van definiálva, az objektum adott függvénye minden kattintásra (amikor a játékoson van a sor) meghívásra kerül és lefut, itt kerül módosításra a pálya grafikája.

2.1 MyGrid

Az ellenfél által lőtt pályát (saját pálya) kezelő osztály, a Grid osztályban leírtakhoz hasonló működéssel. Ugyanakkor ezen a pályán interakció nem végezhető, ezért egér inputokat ez az osztály nem kezel. Itt a saját hajóinkat tároló GridArray változó található. Minden adatcsere után frissül.

1.6 GridCreator

A játék kezdetén kerül létrehozásra, amikor a játékos elhelyezi a pályáján az egyes hajóit. Az elhelyezendő hajók a pálya szélén helyezkednek el, ahonnan a csatamezőre húzhatóak (drag&drop). A pálya megjelenítése hasonló a Grid osztályéhoz, az inputok kiegészítésre kerülnek: a hajók forgatása jobb egérgombra történik, a hajók mozgatása pedig bal egérgomb nyomva tartásával.

1.7 Main

A program törzsét képező osztály, melyben a korábban definiált osztályok létrehozása és a hozzájuk tartozó függvények meghívására kerül sor. Az inicializáló lépéseket követően, egy ciklusba kerülünk, ahonnan a játék végét teljesítő feltétel (gameOver) visz ki. Ebben a ciklusban történik a kommunikációt lebonyolító függvények meghívása is.

2.2 Importált osztályok

A grafikus felület használatához, a képek tárolására és megjelenítésére, a pálya elrendezéséhez, az egér gombok kezelésére, a hálózati kommunikációhoz, illetve a tömbök és koordináták kezeléshez az alábbi osztályokat használjuk:

- java.swing: JFrame, JPanel, JLabel, ImageIcon, JButton, JTextField
- java.awt: BufferedImage, MouseListener, Image, Point
- java.io
- java.net
- java.lang

A megvalósítás során felmerült nehézségek, tapasztalatok

A mi implementációnkban a fejlesztés menete mögötti elgondolás az volt, hogy állítsunk össze egy működő offline torpedó játékot, amit majd később módosítunk, hogy lehessen TCP kommunikáción is játszani.

Ennek a megközelítésnek az előnye az volt, hogy sok időt szántunk magára a játékra és a grafikus felületre, így az elkészült programban sincsenek ezzel kapcsolatos problémák. A felhasználói felület megbízhatóan működik, a grafikus felület várakozáson felülre sikerült.

A megközelítés hátránya azonban az, hogy a tervezés során nem úgy alkottuk meg az osztályokat és az osztályok közti kapcsolatokat, hogy különösebben belegondoltunk volna abba, hogy bizonyos információkat a későbbiekben sorosítva fogunk elküldeni. A kommunikáció integrálása is nehezebb volt utólag, egy gyakorlatilag készre gyártott kódba. Ezért is jöttek elő a korábban említett problémák (lásd: 1.2 fejezet). A tanulság utólag az volt, hogy nagyobb körültekintéssel építhettük volna fel az osztályokat és részletesebben utána járhattunk volna a kommunikációnak, vagy párhuzamosan fejleszthettük volna a kód többi részével.

Összességében a program a specifikációban leírtak szerint működik és a játék játszható két különböző PC-ről is.