EÖTVÖS LORÁND TUDOMÁNYEGYETEM

Informatikai Kar

Programozáselmélet és Szoftvertechnológia Tanszék

Shōbu más játékossal vagy intelligens ellenféllel

Témavezető:
dr. Gregorics Tibor
egyetemi docens

Szerző:
Dunai Tamás
programtervező informatikus BSc

Budapest, 2021

Tartalomjegyzék

1. Bevezetés

1.1. Feladat ismertetése

A Shōbu egy kétszemélyes, teljes információjú, véges, determinisztikus, zéró összegű játék. Manolis Vranas és Jamie Sajdak tervezésével készült, és a *Smirk & Laughter Games* kiadó mutatta be 2019-ben. Szintén 2019-ben *Golden Geek Awards* díjára jelölték *Legjobb két személyes játék* kategóriában. [1]

A szakdolgozat célja egy Shōbu-program implementálása egy vagy két résztvevő számára. A játékot így játszhatjuk egy másik játékos, vagy a számítógép ellen. A program biztosítja, hogy mindkét fél szabályos lépéseket hajthasson végre, valamint észleli, ha az egyik játékos megnyerte a játékot. Van lehetőség időlimit beállítására is, valamint a játékállás elmentésére, betöltésére.

Két résztvevő esetén a játékosok egymás ellen játszhatnak egy gépen, vagy hálózaton keresztül. Lokális játék esetén a felhasználó a játékot elmentheti, és a lépéseit visszavonhatja, hálózati játék esetén felajánlhat és elfogadhat döntetlent. Egy résztvevő esetén a játékos egy mesterséges intelligencia ellen küzd, ami a beállított nehézségi szinttől függően választ lépéseket a játékos ellen. A játék ekkor csak az egyik fél győzelmével érhet véget, döntetlennel nem. Ebben a játékmódban a felhasználó oldalt válthat játék közben.

1.2. Témaválasztás indoklása

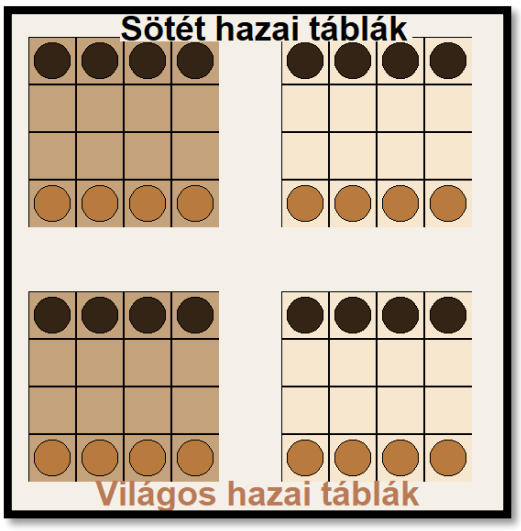
A Shōbu a régi stratégiai játékok hangulatát kelti, akár több ezer évvel ezelőtt is könnyen játszható lett volna néhány kavics segítségével. Nagy előnye, hogy fiatal korának köszönhetően még nem sokan foglalkoztak vele. Nem léteznek hosszú könyvek telve erős stratégiákkal vagy akár egy nyerő- vagy nem vesztő stratégiával. Nincsenek bevált heurisztikák, amiket a fejlesztők már évek óta használnának.

A fejlesztés így visszahozza az ókori játékok, mint például a sakk felfedezésének stratégiai kihívásait anélkül, hogy meglévő megoldásokból kellene építkeznie, a felhasználónak pedig újszerű élményt nyújt anélkül, hogy túl messzire kellene merészkednie a hagyományos, valóban régi játékoktól.

2. Felhasználói leírás

2.1. A Shōbu szabályai

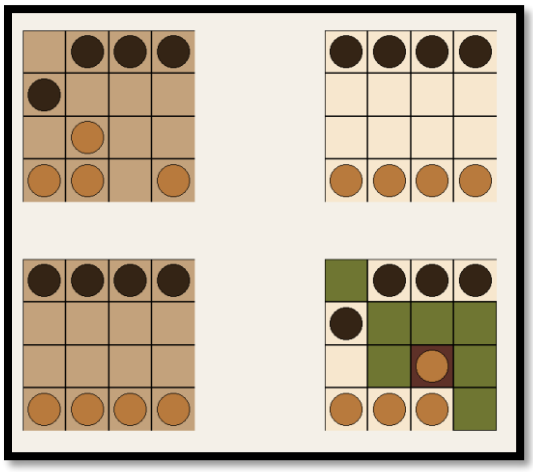
A Shōbu egy kétszemélyes stratégiai játék, amit négy táblán 32 kaviccsal játszanak. A táblák közül kettő világos, kettő sötét, a kavicsok közül pedig 16 világos és 16 sötét aszerint, hogy melyik játékoshoz tartoznak. A játékos akkor nyerheti meg a játékot, ha az egyik tábláról sikeresen leszorítja a másik játékos összes kavicsát, vagy lépésképtelenné teszi. Időlimit alkalmazása esetén a játékos akkor is győzhet, ha az ellenfele kifut az időből.



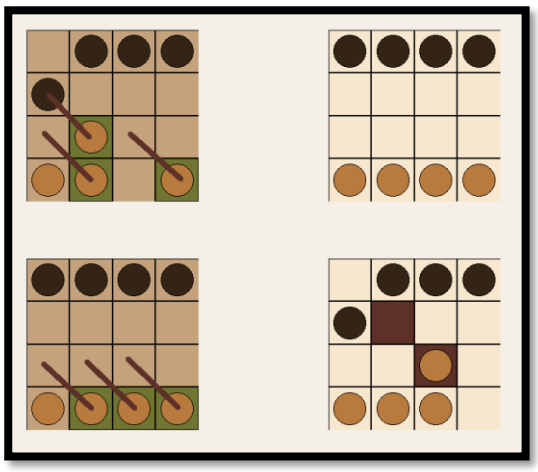
1. ábra: Kezdeti pozíció

A játék elején mindkét játékos kap két-két hazai táblát, egy sötétet és egy világosat, és az 1. ábra szerint elhelyezik rajta a bábuikat. Ezt felváltva lépnek, amíg az egyikük győzelmével véget nem ér a játék. Az első lépés mindig a fehér játékosé.

Egy lépés két szakaszból áll, egy passzív és egy agresszív lépésből. A passzív lépést a játékosnak a hazai táblái egyikén kell megtennie. Ilyenkor az egyik általa birtokolt kavicsot helyezheti odébb egy vagy két mezőnyivel bármelyik irányba egyenesen, vagy átlósan. Ez a lépés csak akkor szabályos, ha egy bábu sem keresztezi az útját, és a lépés végén is a táblán marad.



2. ábra: Lehetséges passzív lépések a kiválasztott kavicshoz



3. ábra: Lehetséges agresszív lépések a kiválasztott passzív lépéshez

Ezt követően az agresszív lépést az egyik ellenkező színű táblán kell megtennie (ha a passzív világos színű táblán volt, az agresszívnek sötéten kell történnie, és fordítva), ám itt már mindegy, hogy a saját, vagy az ellenfele hazai tábláját választja. Az agresszív lépés irányának és nagyságának meg kell egyeznie a passzív lépésével. A kiválasztott kavics ebben az esetben eltolhatja az ellenfél kavicsait, de csak akkor, ha az eltolni kívánt kavics mögött nincs másik kavics. A játékos nem jogosult a saját kavicsainak az odébb lökésére. A játékos saját kavicsának a táblán kell maradnia az agresszív lépés végén, az általa eltolt kavics viszont lekerülhet a tábláról. Az így letolt kavicsok kikerülnek a játékból.

Fontos megjegyezni továbbá, hogy egy passzív lépés csak akkor lehet szabályos, ha létezik hozzá megfelelő agresszív lépés is.

Az 2. és 3. ábrák szemléltetik a világos játékos lépéslehetőségeit. Érdemes megfigyelni, hogy habár a kiválasztott passzív kavics a 2. ábrán képes lenne kettőt balra lépni, ez nem volna szabályos passzív lépés, mivel nem létezik a sötét táblákon olyan kavics, amely képes lenne ugyan ezt a lépést agresszív lépésként megtenni. [2]

2.2. Célközönség

Az alkalmazás elsősorban a kétszemélyes társasjátékok kedvelőinek készült. Akik ismerik és szeretik a Shōbu-t, könnyen megbarátkozhatnak az alkalmazással, de bárkinek remek szórakozást nyújthat, aki nem idegenkedik a régi (vagy legalább régi hangulatú) táblajátékoktól, mint például a sakktól.

A program elsősorban otthoni felhasználásra készült. Jól alkalmazható, amikor a felhasználó egy gyors mentális kihívásra vágyik, vagy egy ismerősével szeretné összemérni stratégiai készségeit.

2.3. Rendszerkövetelmények

2.3.1. Minimális rendszerkövetelmények

- **RAM:** 256 MB
- **Processzor:** 500 MHz
- **Operációs rendszer:** Windows 7
- OpenGL ES 2.0 támogatás

(A Qt 5.11-ben írt alkalmazások becsülhető minimális rendszerkövetelménye) [3]

2.3.2. Ajánlott rendszerkövetelmények

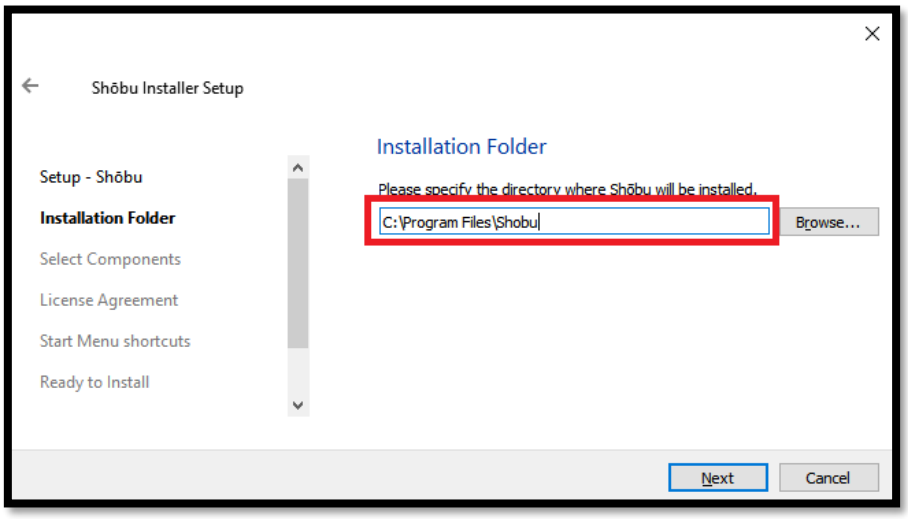
- **RAM:** 8 GB vagy több
- **Processzor:** Intel® Core™ i5-2400 3.10 GHz vagy jobb
- **Operációs rendszer:** Windows 10
- Hálózati hozzáférés (csak online játék esetén)

(A fejlesztéskor használt gép adatai)

2.4. Telepítés

Kövesse a következő lépéseket a telepítés érdekében:

1. Kattintsunk duplán a **ShobuInstaller.exe**, vagy **ShobuServerInstaller.exe** fájlra! Ez meg fogja nyitni az adott szoftver telepítőjét.
2. Kattintsunk a **Next** gombra!
3. Válasszuk ki a mappát, ahová telepíteni szeretnénk az alkalmazást, majd kattintsunk a **Next** gombra!



4. ábra: A mappa kiválasztása telepítés közben

4. Kattintsunk ismét a **Next** gombra!
5. A licensz szerződés alapos átolvasása után kattintsunk az felirat melletti jelölőnégyzetre, amennyiben egyetértünk a szerződés tartalmával, majd kattintsunk a **Next** gombra!

6. Kattintsunk ismét a **Next** gombra!
7. Kattintsunk az **Install** gombra!
8. Várjuk meg, amíg a telepítés befejeződik! Ez több percig is eltarthat.
9. Kattintsunk a **Finish** gombra! A telepítés ezzel befejeződik.
10. Az alkalmazást **Shobu.exe** vagy **ShobuServer.exe** fájlra duplán kattintva indíthatjuk el, amit a korábban kiválasztott mappában találhatunk meg

2.5. Az alkalmazás használata

A szerver elindításához kattintsunk duplán a **ShobuServer.exe**. Ekkor megnyílik egy konzol ablak. Amennyiben a „Connected” felirat jelent meg, a szerver sikeresen elindult. Ha a „Not connected” felirat jelenik meg, a szerver nem tudott elindulni.

Az alkalmazás elindításához kattintsunk duplán a **Shobu.exe** fájlra. Ekkor az alkalmazás el fog indulni, és elsőként a főmenü fog megjelenni.

2.5.1. Főmenü

A főmenüben négy gomb található, a „**New Game**”, a „**Load Game**”, a „**Multiplayer**” és az „**Exit**”.

5. ábra: A főmenü gombjai

Új, egyszemélyes játékot a **New Game** gombra kattintva indíthatunk. Ekkor felugrik egy ablak, ami a játék beállításait fogja bekérni. A játék beállításairól a dokumentáción belül később találunk részletesebb leírást a **2.5.3 Beállítások ablak** résznél. Miután kiválasztottuk a kívánt beállításokat, az új egyszemélyes játék elindul. Ebben a játékmódban a felhasználó a számítógép ellen fog játszani.

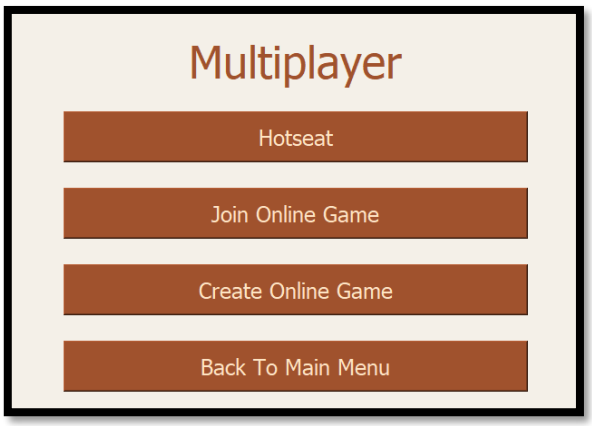
A **Load Game** gombra kattintva felugrik egy ablak, amin keresztül kiválaszthatunk egy korábban elmentet játékot. A játék kiválasztásának folyamatáról a dokumentáción belül később találunk részletesebb leírást a **2.5.3 Beállítások ablak** résznél.

A **Multiplayer** gombra kattintva átkerülünk a többjátékos menüre. A többjátékos menüről bővebben olvashatunk a **2.5.2 Többjátékos menü** résznél.

Az **Exit** gombra kattintva kiléphetünk az alkalmazásból. Hasonlóképpen bármikor kiléphetünk az alkalmazásból az alkalmazás ablakának jobb felső sarkában lévő **X** gombra kattintva. Ennek kinézete az operációs rendszer beállításaitól függően változhat.

2.5.2. Többjátékos menü

A többjátékos menü a főmenühöz hasonlóan négy gombot tartalmaz. Ezek a következők: „**Hotseat**”, a „**Join Online Game**”, a „**Create Online Game**” és az „**Back To Main Menu**”.



6. ábra: Többjátékos menü gombjai

A **Hotseat** gombra kattintva felugrik a beállítások ablak. A játék beállításairól a dokumentáción belül később találunk részletesebb leírást a **2.5.3 Beállítások ablak** résznél. Miután kiválasztottuk a kívánt beállításokat, az új kétszemélyes játék elindul. Ebben a játékmódban mind a fehér, mind a fekete bábukat a felhasználó irányítja. Ideális esetben két külön felhasználó játszat egy számítógépen, de akár egy felhasználó is játszhat saját maga ellen.

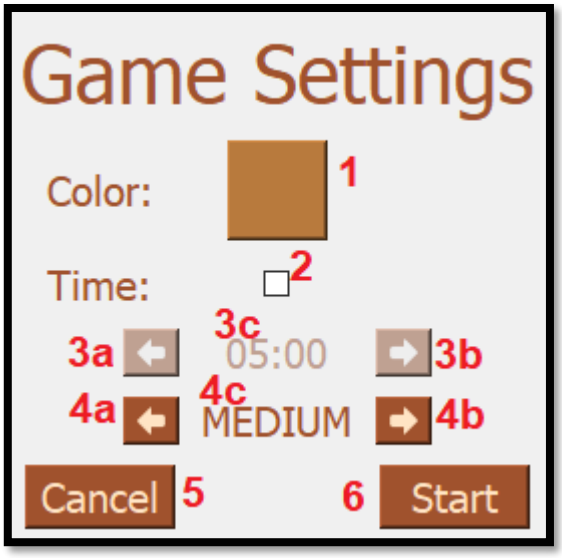
A **Join Online Game** gombra kattintva csatlakozhatunk online játékokhoz. Ezt megelőzően csatlakoznunk kell egy szerverhez, majd ki kell választanunk a játékot, amelyhez csatlakozni szeretnénk. Ennek menetéről a **2.5.7 Szerverhez csatlakozó ablak** és **2.5.6 Online játékhoz csatlakozó ablak** részeknél bővebben olvashatunk a dokumentációban.

A **Create Online Game** gombra kattintva készíthetünk új online játékot. Ehhez először csatlakoznunk kell a szerverhez, majd meg kell adnunk a játékunk beállításait. Ennek menetéről a **2.5.7 Szerverhez csatlakozó ablak** és **2.5.3 Beállítások ablak** részeknél bővebben olvashatunk a dokumentációban. Amennyiben sikeresen létrehoztuk a játékot, a váróterembe kerülünk, amíg nem csatlakozik a játékhoz az ellenfelünk. A váróteremről részletesebben a **2.5.7 Váróterem** résznél olvashatunk.

A **Back To Main Menu** gombra kattintva visszatérhetünk a főmenüre. Erről bővebben a **2.5.1 Főmenü** résznél olvashattunk.

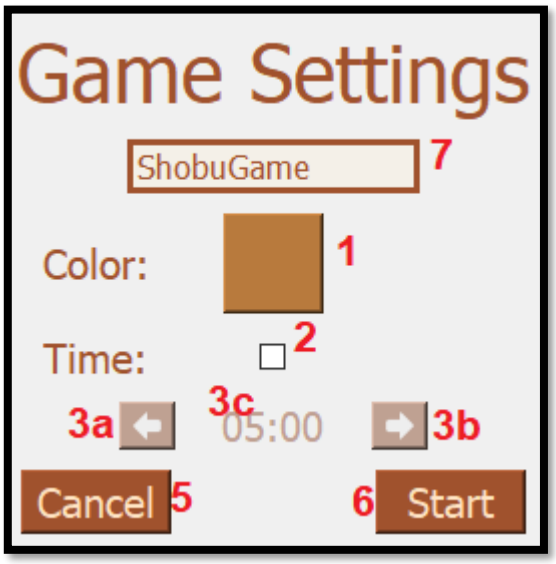
2.5.3. Beállítások ablak

A beállítások ablak a kiválasztott játékállástól függően változhat. Az adatbeviteli eszközök nevei a 7., 8. és 9. ábrán láthatóak.



7. ábra: Egyszemélyes játék beállításai

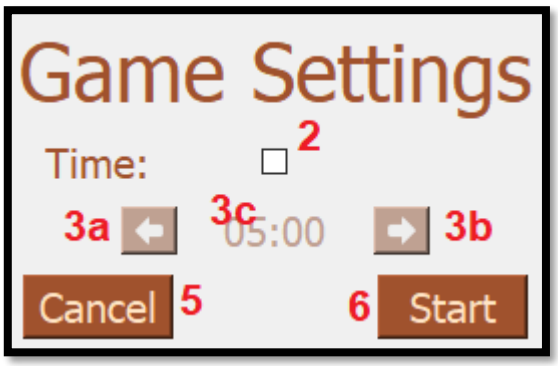
- **1** Színváltó gomb
- **2** Időlimit jelölődoboz
- **3a** Időlimit csökkentő gomb
- **3b** Időlimit növelő gomb
- **3c** Időlimit kijelző
- **4a** Nehézség csökkentő gomb
- **4b** Nehézség növelő gomb
- **4c** Nehézség kijelző
- **5** Cancel gomb
- **6** Start gomb
- **7** Játék neve



8. ábra: Online játék beállításai

Amennyiben nem kétszemélyes helyi játékra készülünk, a színváltó gombra (**1**) kattintva változtathatjuk meg a saját színünket. Amennyiben világosbarna színű, a kattintás hatására sötétbarnára vált, ha pedig eredetileg sötétbarna volt, világosbarnára vált.

Az időlimit jelölődobozzal (**2**) állíthatjuk be, hogy szeretnénk-e időlimitet a játékba, vagy sem. Ha kipipáljuk, az időlimit növelő (**3b**) és csökkentő (**3a**) gombok elérhetővé válnak, az aktuális időlimitet pedig közöttük (**3c**) láthatjuk. Amennyiben a jelölődoboz nincs kipipálva, a kijelzett időlimit nem fog életbe lépni a játék indításakor.



9. ábra: Kétszemélyes helyi játék beállításai

Az időlimit csökkentő gomb (**3a**) lenyomásával az időlimitet fél perccel csökkentjük. Az időlimit alsó határa 30 másodperc, ebben az esetben az időlimit csökkentő gomb elérhetetlenné válik, amíg nem növeljük az időlimitet. Az időlimit növelő gomb (**3b**) fél perccel növeli az aktuális időlimitet. Ennek felső határa nincsen.

Egyszemélyes játék esetén a játék nehézségét a nehézség növelő (**4b**) és csökkentő (**4a**) gombokkal állíthatjuk be, az aktuális nehézséget pedig közöttük (**4c**) láthatjuk. Három nehézségi szint áll rendelkezésünkre („EASY”, „MEDIUM”, „HARD”). Amennyiben az aktuális nehézség „EASY”, a nehézségi szint csökkentő gomb elérhetetlenné válik, amíg a nehézségi szintet nem növeljük, ha pedig az aktuális nehézségi szint „HARD”, a nehézségi szint növelő gomb válik elérhetetlenné, amíg nem csökkentjük a nehézségi szintet. Az „EASY” beállítás a legegyszerűbben megverhető ellenfelet biztosítja számunkra, a „MEDIUM” egy közepes erősségűt, a „HARD” pedig a legnehezebbet.

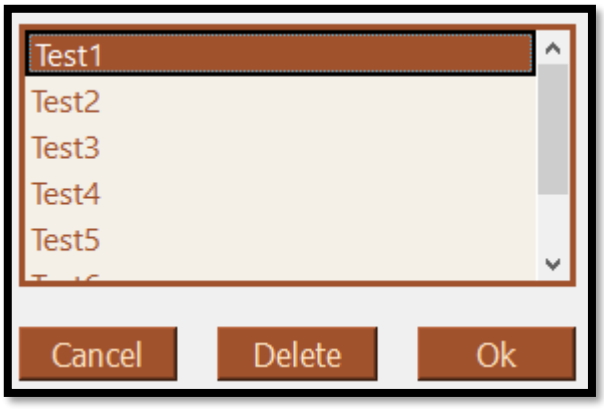
A Cancel (**5**) gomb lenyomásával elvethetjük a beállításainkat. Ebben az esetben a beállításainkkal nem kezdünk új játékot, vagy ha már van meglévő játékunk, azt nem befolyásolja. Amennyiben online játékot szerettünk volna létrehozni, ez a gomb lecsatlakoztat minket a szerverről.

A Start (**6**) gombra kattintva életbe léptetjük a beállítások ablakunkon feltüntetett aktuális beállításokat. Új játék esetén elkezdődik a játék a beállításainkkal, online játék esetén létrejön az online játék, és a váróterembe kerülünk, ha pedig már játékban vagyunk, az új beállításaink felülírják az előzőt.

Online játék készítése esetén a Játék nevére (**7**) kattintva átírhatjuk azt. A játék neve minimum 1, maximum 15 karakterből állhat, és nem egyezhet meg egy másik online játék nevével sem. Ez a név lesz látható a többi játékos számára, amikor online játékhoz próbálnak csatlakozni.

2.5.4. Játék betöltő ablak

A játék betöltő ablakot használhatjuk a korábban elmentett játékállásaink helyreállítására, valamint ezek letörlésére. A betölthető játékok neveit egy listában találjuk az ablak három gombja („Cancel”, „Delete” és „Ok”) felett. Az aktuális

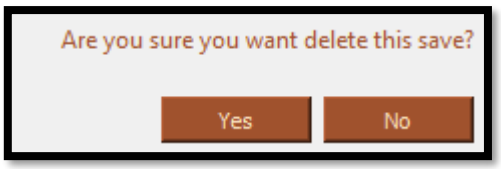


10. ábra: Játék betöltő ablak

kiválasztott játékot a játék nevére kattintva változtathatjuk meg. Kezdetben mindig a legelső játék a kiválasztott játék, amennyiben létezik.

A **Cancel** gombra kattintva elvethetjük a játék betöltését. Ez bezárja az ablakot, és nem tölt be egy játékot sem.

A **Delete** gombra kattintva letörölhetjük az aktuálisan kiválasztott játékot. Ezt követően egy megerősítő ablak fog felugrani, melynek az **Yes** gombjára kattintva megerősíthetjük törlési szándékunkat, **No** gombjára kattintva pedig visszavonhatjuk. Amennyiben nincs kiválasztott játék, a **Delete** gomb semmit nem csinál.

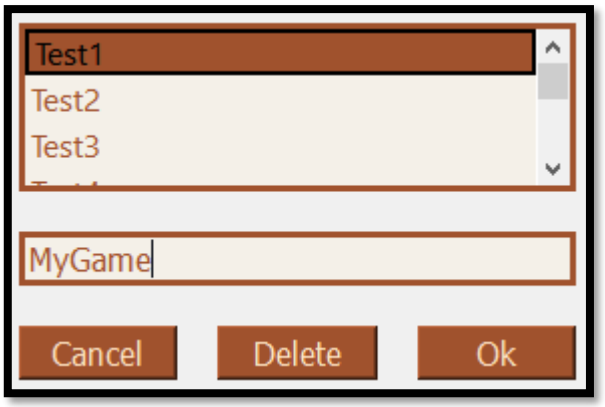


11. ábra: Mentés törlésének megerősítése

Az **Ok** gombra kattintva betölthetjük a kiválasztott játékállást. helyesen kiválasztott mentés esetén ez helyreállítja a kiválasztott állást, és a mentés játékmódjától függően a megfelelő játékba visz minket. Amennyiben nincs kiválasztott játék, az **Ok** gomb a **Cancel** gombhoz hasonlóan bezárja az ablakot, és nem tölt be játékot.

2.5.5. Játék elmentő ablak

A játék elmentő ablak segítségével menthetjük el az aktuális játékállásunkat. Erre csak a helyi játékok esetén van lehetőségünk, online játéknál nem. A játék elmentő ablak ugyan úgy néz ki, mint a játék betöltő ablak, annyi különbséggel, hogy az elérhető játékok listája és az alul található gombok között van egy szövegbeviteli mező is. Ebbe a szövegbeviteli mezőbe írhatjuk be a nevet, amivel később megtalálhatjuk az elmentett játékállásunkat.



12. ábra: Játék elmentő ablak

A lista elemeire kattintva a kiválasztott játék neve kerül be a beviteli mezőbe. Jóváhagyás esetén ekkor a kiválasztott elemet felülírjuk az új játékállásunkkal.

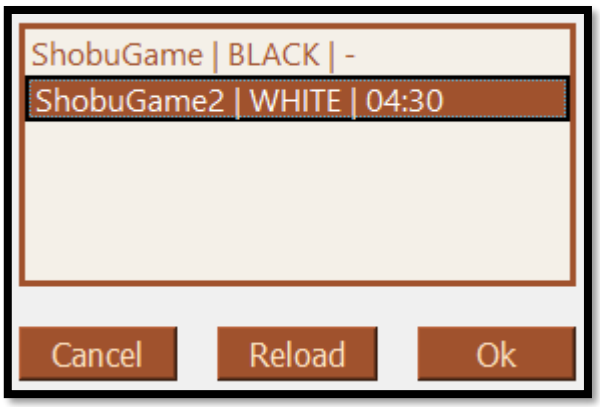
A **Cancel** gombra kattintva elvethetjük a játék elmentését. Ez bezárja az ablakot, és nem menti el az aktuális játékállást.

A **Delete** gomb pont ugyan úgy működik, mint a játék betöltő ablaknál. Erről részletesebben a **2.5.4 Játék betöltő ablak** résznél olvashattunk.

Az **Ok** gombra kattintva elmenthetjük az aktuális játékállást azon a néven, ami éppen a szövegbeviteli mezőben van. Amennyiben ez a név megegyezik egy korábbi mentés nevével, az felülírásra kerül, és a korábbi mentést elveszítjük. Sikeres mentés esetén egy értesítő ablak tájékoztat minket a mentés sikerességéről. Amennyiben a szövegbeviteli mező üres, az **Ok** gomb a **Cancel** gombhoz hasonlóan bezárja az ablakot, és nem menti el a játékállást.

2.5.6. Online játék kiválasztó ablak

Az online játék kiválasztó ablak segítségével csatlakozhatunk egy már meglévő online játékhoz. Az elérhető játékok neveit és főbb beállításait egy listában találjuk az ablak három gombja („Cancel”, „Reload” és „Ok”) felett. Az aktuális kiválasztott játékot a játék nevére kattintva változtathatjuk meg. Kezdetben mindig a legelső játék a kiválasztott játék, amennyiben létezik. Ha nincs elérhető online játék, erről egy felugró ablak értesít minket.

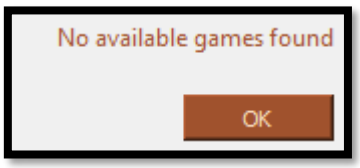


13. ábra: Online játék kiválasztó ablak

A játékok listájának egy eleme három tagból áll, „|” karakterekkel elválasztva. A legelső tag a játék neve, ennek egyedinek kell lennie. A második a szín, amit a felhasználó megkap, amennyiben csatlakozik a játékhoz („WHITE” esetén világos, „BLACK” esetén sötét). A harmadik az időlimit percekben és másodpercekben. Amennyiben az időlimit helyén a „-” karaktert látjuk, az azt jelenti, hogy az adott játékhoz nem tartozik időlimit, így mindkét játékos korlátlan ideig gondolkozhat minden lépése előtt.

A **Cancel** gombra kattintva elvethetjük a csatlakozási szándékunkat. Ez bezárja az ablakot és lecsatlakozik a szerverről.

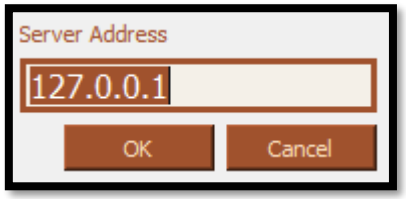
A **Reload** gombra kattintva frissíthetjük az elérhető játékok listáját. A lista elemei ekkor lecserélődnek, és a kiválasztott listaelem ismét a legelső lesz, amennyiben létezik. Ha nincsenek elérhető játékok a szerveren, erről egy felugró ablak értesít minket.



14. ábra: Értesítő, amikor nincs elérhető játék

Az **Ok** gombra kattintva csatlakozhatunk a kiválasztott online játékhoz. Erre kattintva a váróterembe kerülünk, amíg a szerver vissza nem jelez. Ha a szerver nincs megterhelve, elképzelhető, hogy a váróterem olyan rövid ideig jelenik csak meg, hogy észre sem vesszük. Amennyiben az online játék még mindig elérhető, ezt követően megkezdjük a játékot. Ha már nem elérhető, erről egy felugró ablak értesít minket, és visszatérünk a többjátékos menühöz. Erről részletesebben majd a **2.6 Rendszerüzenetek** résznél olvashatunk.

2.5.7. Szerverhez csatlakozó ablak



15. ábra: Szerverhez csatlakozó ablak

A szerverhez csatlakozó ablak akkor jelenik meg, ha a többjátékos menüben rákattintunk a **Join Online Game** vagy a **Create Online Game** gombok egyikére. Erről részletesen a **2.5.2 Többjátékos menü** résznél olvashattunk. Az ablak segítségével választhatjuk ki a szervert, amihez csatlakozni szeretnénk. Ezt az ablakon lévő szövegbeviteli mező segítségével tehetjük meg. Alapértelmezetten itt a „127.0.0.1” címet találjuk, ez abban az esetben fog működni, ha a szerver ugyan azon a számítógépen fut, mint a Shōbu alkalmazás. Amennyiben a szerver nem a mi számítógépünkön fut, ezt lecserélhetjük a szerver helyes címével.

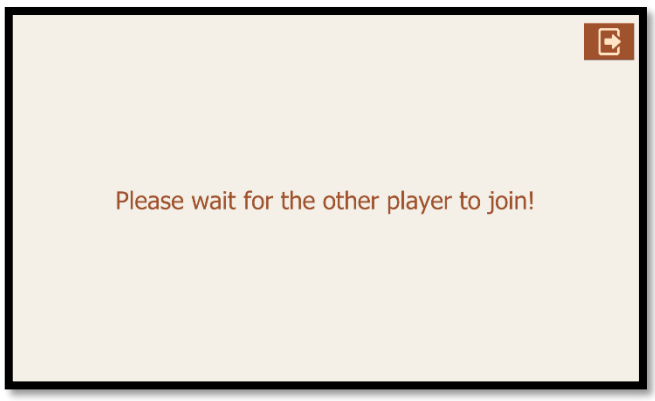
A **Cancel** gombra kattintva elvethetjük a csatlakozási szándékunkat. Ekkor a szerverhez csatlakozó ablak bezárul.

Az **Ok** gombra kattintva megkíséreljük a csatlakozást a szerverhez, amelynek a címe a szövegbeviteli mezőben van. Siker esetén ezt követően tovább lépünk az Online játék kiválasztó ablakhoz (erről bővebben a **2.5.6 Online játék kiválasztó ablak** résznél olvashattunk), vagy a Beállítások ablakhoz (lásd **2.5.3 Beállítások ablak**) attól függően, hogy milyen művelet közben ugrott fel a Szerverhez csatlakozó ablak. Amennyiben nem tudunk csatlakozni a kiválasztott szerverhez, erről értesítést kapunk.

2.5.7. Váróterem

A váróterembe akkor kerülünk, amikor már létrehoztunk egy játékot, és a másik játékosra várunk, vagy amikor csatlakoztunk egy játékhoz és a szerver visszajelzésére várunk.

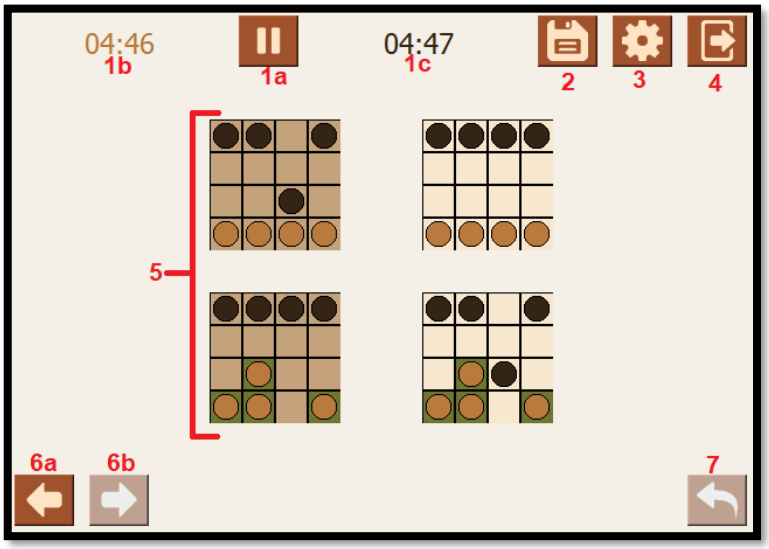
A váróteremben egyetlen gomb van a jobb felső sarokban. Erre kattintva kilépünk a váróteremből, és visszatérünk a főmenühöz (lásd **2.5.1 Főmenü**). Ezzel lecsatlakozunk a szerverről, és véget vetünk az aktuális játéknak, amire éppen várakoztunk.



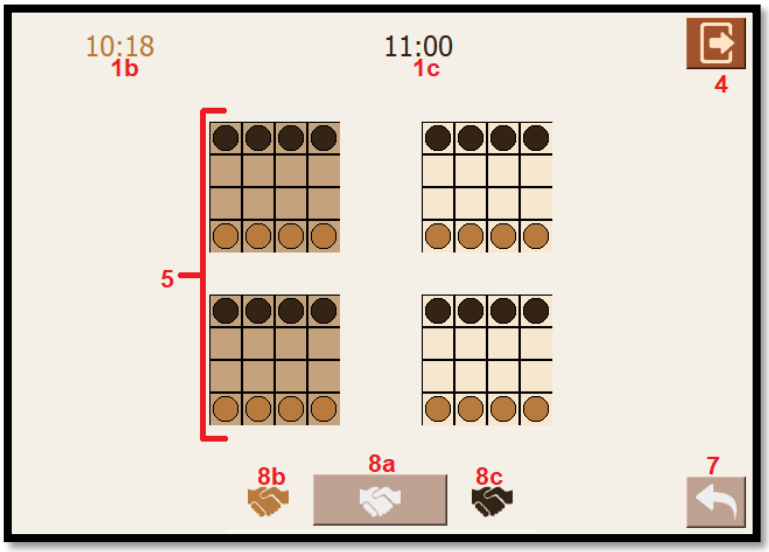
16. ábra: Váróterem

2.5.8. Játékablak

A játék az alkalmazás legfontosabb része. Itt mérhetjük össze stratégiai tudásunkat a géppel, egy másik játékossal, akár helyben, vagy a hálózaton keresztül. A játékmódtól függően itt különböző gombok segítségével navigálhatunk, amiket a következő felsorolásban, és a hozzájuk tartozó 17. és 18. ábrán láthatunk.



17. ábra: Játékablak (helyi)



18. ábra: Játékablak (online)

- **1a** Idő gomb
- **1b** Világos idő kijelző
- **1c** Sötét idő kijelző
- **2** Mentés gomb
- **3** Beaállítások gomb
- **4** Játék elhagyó gomb
- **5** Játéktér

- **6a** Visszavonás gomb
- **6b** Lépés ismétlő gomb
- **7** Lépés újrakezdő gomb
- **8a** Döntetlen ajánló gomb
- **8b** Világos döntetlen ajánlat kijelző
- **8c** Sötét döntetlen ajánlat kijelző

Az Idő gomb (**1a**) szolgál az idő mérésének megállítására és újraindítására. Amennyiben éppen áll az idő, rákattintva újraindíthatjuk azt, vagy ha éppen mérjük az időt, a kattintással megállítjuk a mérést. A gomb mindig az aktuális művelethez igazítja a rajta lévő ikont. Ha a kattintással megállítanánk az idő mérését, akkor két téglalap van rajta, ha pedig elindítanánk, egy háromszög. Ez a gomb csak akkor jelenik meg a játékablakon, ha az aktuális játékhoz be van állítva időlimit, és helyi játékot játszunk. Online játék esetén nincs mód az idő megállítására.

Az idő kijelzők (**1b** és **1c**) az adott színhez tartozó játékos hátralévő idejét mutatják. Amennyiben éppen mérjük az időt, a soron következő játékos ideje másodpercentként eggyel csökken. A kijelzők csak akkor jelennek meg a játékablakon, ha az aktuális játékhoz be van állítva időlimit. Amennyiben gépi ellenfél ellen játszunk, a kijelzők közül csak a játékos színéhez tartozó időlimit látható. A gépi ellenfél soha nem fogyhat ki az időből.

A mentés gombra (**2**) kattintva felugrik a játék elmentő ablak. (Részletesebben erről a **2.5.5 Játék elmentő ablak** résznél olvashattunk.) A mentés gomb csak helyi játékoknál jelenik meg a játékablakon, online játékok elmentésére nincs lehetőségünk.

A beállítások gombra (**3**) kattintva felugrik a beállítások ablak (lásd **2.5.3 Beállítások ablak**). A felugró ablak eredetileg az aktuális beállításokat fogja mutatni. Amennyiben ennek segítségével megváltoztatjuk az aktuális játék beállításait, a játékablak tartalma az új beállításokhoz igazodva megváltozhat. A beállítások gomb csak helyi játékoknál jelenik meg a játékablakon, online játékok beállításainak megváltoztatására nincs lehetőségünk a játék megkezdése után.

A játék elhagyó gombra (**4**) kattintva megszakíthatjuk a folyamatban lévő játékot, és visszatérhetünk a főmenüre. Ehhez a program először megerősítést kér. Online játék esetén a

gombot lenyomva megszakítjuk a szerverrel a kapcsolatot, és az ellenfelünk automatikusan megnyeri a játékot.

A játéktéren (**5**) láthatjuk az aktuális állást, és ennek segítségével léphetünk a saját körünkben. Ennek menetéről részletesebben a **2.5.9 Játéktér** résznél fogunk olvasni.

A visszavonás gombra (**6a**) kattintva léphetünk vissza egy korábbi játékállásra. A csak akkor elérhető, amikor létezik olyan játékállás, amire visszaléphetünk. Mindig egy olyan játékállásra léphetünk csak vissza, ahol a soron következő játékos a felhasználó irányítása alatt áll. (Egyjátékos esetén a gépi ellenfél körét átugorjuk, két játékos esetén csak egyet lépünk vissza.) A visszavonás gomb csak helyi játékoknál jelenik meg, online játék esetén nem vonhatjuk vissza a lépéseinket.

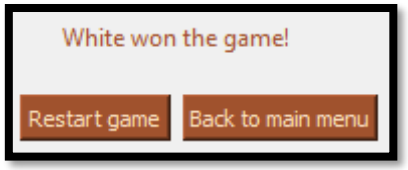
A lépés ismétlő gombra (**6b**) kattintva állíthatjuk vissza azokat a játékállásokat, amiket korábban visszavontunk. A gomb csak akkor elérhető, ha létezik olyan játékállás, amit korábban visszavontunk. Amennyiben egy játékállás visszavonása után szabályos lépést teszünk, a visszavont állapot(ok) helyreállításának lehetősége elveszik, és a gomb elérhetetlenné válik. Mindig egy olyan játékállásra léphetünk csak vissza, ahol a soron következő játékos a felhasználó irányítása alatt áll. (Egyjátékos esetén a gépi ellenfél körét átugorjuk, két játékos esetén csak egyet lépünk előre.) A lépés ismétlő gomb csak helyi játékoknál jelenik meg, online játék esetén nem vonhatjuk vissza a lépéseinket.

A lépés újrakezdő gombra (**7**) kattintva újrakezdhetjük a folyamatban lévő lépésünket. A gomb csak akkor elérhető, ha létezik egy folyamatban lévő lépésünk. A gomb lenyomását követően, a játéktér frissül, és újrakezdhetjük a lépésünket a passzív kavics kiválasztásával.

A döntetlen ajánló gombbal (**8a**) ajánlhatunk döntetlent az ellenfelünknek. A gomb csak online játékoknál jelenik meg, helyi játék esetén a játék idő előtti elhagyása számít döntetlennek. Az ajánlatról a szerver értesíteni fogja az ellenfelünket. Miután döntetlent ajánlottunk, a gomb elérhetetlenné válik, amíg valamelyik játékos szabályos lépést nem tesz, ezzel elutasítva minden függőben lévő ajánlatot. Amennyiben mindkét játékosnak van párhuzamosan döntetlen ajánlata, a játék döntetlennel véget ér.

A döntetlen ajánlat kijelzők (**8b** és **8c**) jelzik a játékosok számára a függőben lévő döntetlen ajánlatokat. Amennyiben valamelyik játékosnak van ilyen ajánlata, az ő színéhez tartozó kézfogás ikon megjelenik a játékablakon. Az ajánlat elutasítása esetén ez az ikon eltűnik. Ha mindkét ikon egyszerre megjelenik, a játék döntetlennel véget ér.

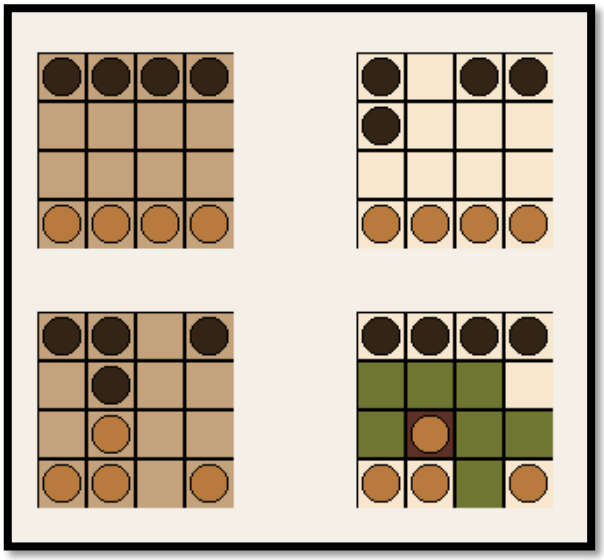
Amennyiben a játékot az egyik fél megnyerte, a program erről értesítést küld egy felugró ablak segítségével. Helyi játék esetén innen kétféleképpen léphetünk tovább. A **Restart Game** gomb lenyomásával új játékot kezdhetünk a meglévő beállításinkkal, a **Back to main menu** gombra kattintva pedig visszatérhetünk a főmenüre. Online játék esetén az értesítő ablak csak az **Ok** gombot tartalmazza, amit lenyomva visszatérhetünk a többjátékos menüre.



19. ábra: Helyi játék vége

2.5.9. Játéktér

A játéktéren láthatjuk az aktuális állást, és ennek segítségével léphetünk a saját körünkben. A játéktér négy darab 4x4-es táblából áll, ebből a két bal oldali sötétebb, a két jobboldali világosabb. A fent elhelyezkedő táblák a sötét játékos hazai táblái, a lent lévők pedig a világos játékoséi. A kezdeti felállásban a 4-4 világos kavics a táblák alján, míg a 4-4 sötét kavics a táblák tetején helyezkedik el.



20. ábra: Folyamatban lévő játék. A passzív kavics már ki lett választva, így a zöld mezők a lehetséges passzív célállomásokat jelölik

Egy lépést három kattintással lehet megtenni. Először ki kell választanunk a passzív lépésünkhöz tartozó kavicsunkat az egyik hazai táblánkon, utána rá kell kattintanunk a mezőre, ahová lépni szeretnénk vele, majd végül ki kell választanunk a kavicsot, amivel az agresszív lépést szeretnénk megtenni.

Egy-egy kattintásnak csak akkor lesz hatása, ha a Shōbu szabályai szerint egy elfogadható lépés részét képezheti. A szabályokról bővebben a **2.1 A Shōbu szabályai** fejezetben olvashattunk.

A táblákon a már kiválasztott mezők vöröses barna színűek, a szabályosan kiválasztható mezők pedig zöldek.

Az agresszív lépés kavicsának kiválasztása után a program frissíti az aktuális játékállást a lépésünk szerint, és a következő játékos köre következik. Gépi ellenfél esetén a gép pár másodpercen belül lép egyet, kétszemélyes helyi játék esetén ismét a felhasználóra (vagy

vélhetőleg egy másik felhasználóra) hárul a lépés felelőssége, online játék esetén pedig a szerver értesítést kap a felhasználó lépéséről.

2.6. Rendszerüzenetek

A játék időnként felugró ablakok segítségével fog kommunikálni velünk. Néhányról már említést tettünk a **2.5 Az alkalmazás használata** fejezetben, itt viszont részletesebben kitérünk minden üzenetre, és azok jelentésére.

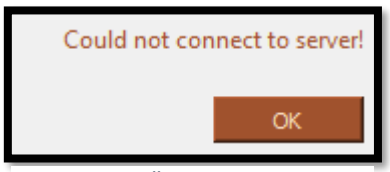
2.6.1. Szerver indítása

Amennyiben a szerver megnyitáskor a „Connected” üzenetet jeleníti meg a konzolon, a szerver sikeresen elindult, és nincs további teendőnk.

Ha viszont a „Not connected” üzenetet látjuk, a szerver nem indult el megfelelően. Ez esetben győződjünk meg arról, hogy a tűzfalunk beállításai nem gátolják a szoftver működését. Ha a tűzfal beállításai megfelelőek, ellenőrizzük le, hogy az 1234-es port szabad, és egy másik alkalmazás sem használja, és szabadítsuk fel, amennyiben foglalt.

2.6.2. Csatlakozás a szerverhez

Előfordulhat, hogy a csatlakozási kísérletünk (lásd **2.5.7 Szerverhez csatlakozó ablak**) nem jár sikerrel. Ilyenkor pár másodperc szünet után erről egy felugró ablak tájékoztat, „Could not connect to server!” üzenettel.



21. ábra: Üzenet a sikertelen csatlakozásról

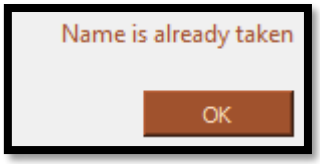
Ennek a hibának az elhárítása érdekében legelőször győződjünk meg a beírt cím helyességéről. Amennyiben a cím biztosan jó, ellenőrizzük le a hálózati kapcsolatunkat. Ha ott is mindent rendben találunk, győződjünk meg arról, hogy a tűzfal nem akadályozza a program helyes működését. Abban az esetben, ha eddig minden helyesnek látszik, vegyük fel a kapcsolatot a szerver üzemeltetőjével.

Fontos: Csak olyan szerverhez csatlakozzunk, amiben maradéktalanul megbízunk!

2.6.3. Név foglalt

Ha olyan névvel próbálunk online játékot létrehozni, ami már létezik a szerveren, a játék létrehozása sikertelen lesz. A program tovább enged minket a váróterembe, de amint a szerver

jelzi a problémát, egy felugró ablakkal tájékoztat a játék létrehozásának sikertelenségéről. Ezt bezárva visszatérünk a többjátékos menübe.

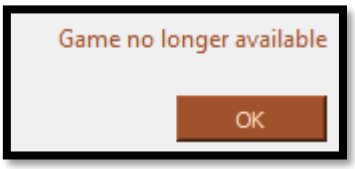


22. ábra: Üzenet, ha a választott játéknevünk már foglalt

Amennyiben azt az üzenetet kapjuk, próbáljunk meg más néven játékot létrehozni. Ajánlatos lehet olyan nevet választani, ami nem valószínű, hogy másoknak is eszébe juthat.

2.6.4. Játék már nem elérhető

Előfordulhat, hogy a játék, amihez csatlakozni próbálunk, idő közben elérhetetlenné vált. Ilyenkor a program tovább enged minket a váróterembe, de amint a szerver jelzi a problémát, egy felugró ablakkal tájékoztat a csatlakozás sikertelenségéről. Ezt bezárva visszatérünk a többjátékos menübe.



23. ábra: Üzenet a játék elérhetetlenségéről

Ennek elkerülése érdekében érdemes a lehető legrövidebbre csökkenteni a játékok közötti válogatás idejét. Amennyiben sok idő telt el az elérhető játékok kilistázása után, ajánlatos lehet frissíteni a listát, mielőtt csatlakozni próbálunk.

2.6.5. Kapcsolat megszakadt a szerverrel

Amennyiben a szerver és a számítógépünk közötti kommunikációban hiba lép fel, a szerver bontja a kapcsoltot, és erről egy felugró ablak segítségével értesít minket. Amennyiben játékban voltunk, a játék véget ér, és valószínűleg az ellenfelünk megnyeri a játékot (amennyiben ő nem veszítette el a kapcsolatot).

Ha ezzel az üzenettel találkozunk, ellenőrizzük le a hálózati kapcsolatunk stabilitását. Ha mindent rendben találtunk, próbáljunk meg újra csatlakozni. Amennyiben ez nem sikerül, elképzelhető, hogy a szerver leállt. Ekkor vegyük fel a kapcsolatot a szerver üzemeltetőjével!

2.6.6. Idő előtti győzelem

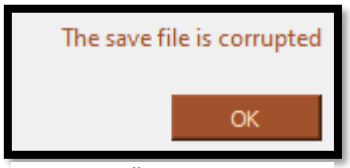
Előfordulhat, hogy a játékot váratlanul megnyerjük annak ellenére, hogy nem győztük le az ellenfelünket, és nem is futott ki az időből. Ez akkor történik, ha az ellenfelünk valamilyen oknál fogva idő előtt elhagyta a játékot. Ilyenkor a szerver nekünk ítéli a győzelmet, amiről egy felugró ablak által értesülünk. Az ablakot bezárva visszatérünk a többjátékos menübe.

2.6.7. Sikeres mentés ablak

Amennyiben sikeresen elmentünk egy játékot, ezt egy felugró ablak jelzi nekünk a „Save successful” üzenettel. Amennyiben ezt az üzenetet látjuk, nincs további teendőnk, az **Ok** gombra kattintva bezárhatjuk azt. Ha a mentést követően nem látjuk ezt az üzenetet, győződjünk meg arról, hogy tényleg elneveztük-e a mentésünket. Név nélküli játékot nem menthetünk el. Ha a „Save failed” üzenetet látjuk, az a mentés sikertelenségéről árulkodik. Ebben az esetben próbáljuk meg újra elmenteni az aktuális állapotot, újraindítani az alkalmazást, majd végsősoron újratelepíteni az alkalmazást.

2.6.7. Sikertelen betöltés

Ha a mentés fájl valamilyen oknál fogva megsérült, és nem betölthető, arról hibaüzenetet kapunk. Ebben az esetben sajnos nincs módunk az elmentett játék betöltésére, így a mentés törlése a javasolt eljárás.



24. ábra: Üzenet a sikertelen betöltésről

3. Fejlesztői leírás

3.1. Elemzés

3.1.1. Feladat leírás

A szakdolgozat célja egy Shōbu program megírása. Ennek megvalósítása érdekében a legfontosabb a szabályok implementálása. A programnak képesnek kell lennie az aktuális játékállás frissítésére egy szabályos lépéssel, valamint észre kell vennie, ha a játék véget ért.

A mesterséges intelligenciának a programban képesnek kell lennie az összes szabályos lépés felsorolására, valamint a lépések közül ki kell tudnia választani egyet valamilyen szempont szerint. A lépések felsorolása felhasználható a játékos segítésére is.

A felhasználó játékmódtól függően be kell, hogy tudja állítani a saját színét, az időlimitet és annak hosszát, a gépi ellenfél nehézségi szintjét, valamint a játék nevét. Lokális játék esetén ezeket játék közben is változtathatóvá kell tenni.

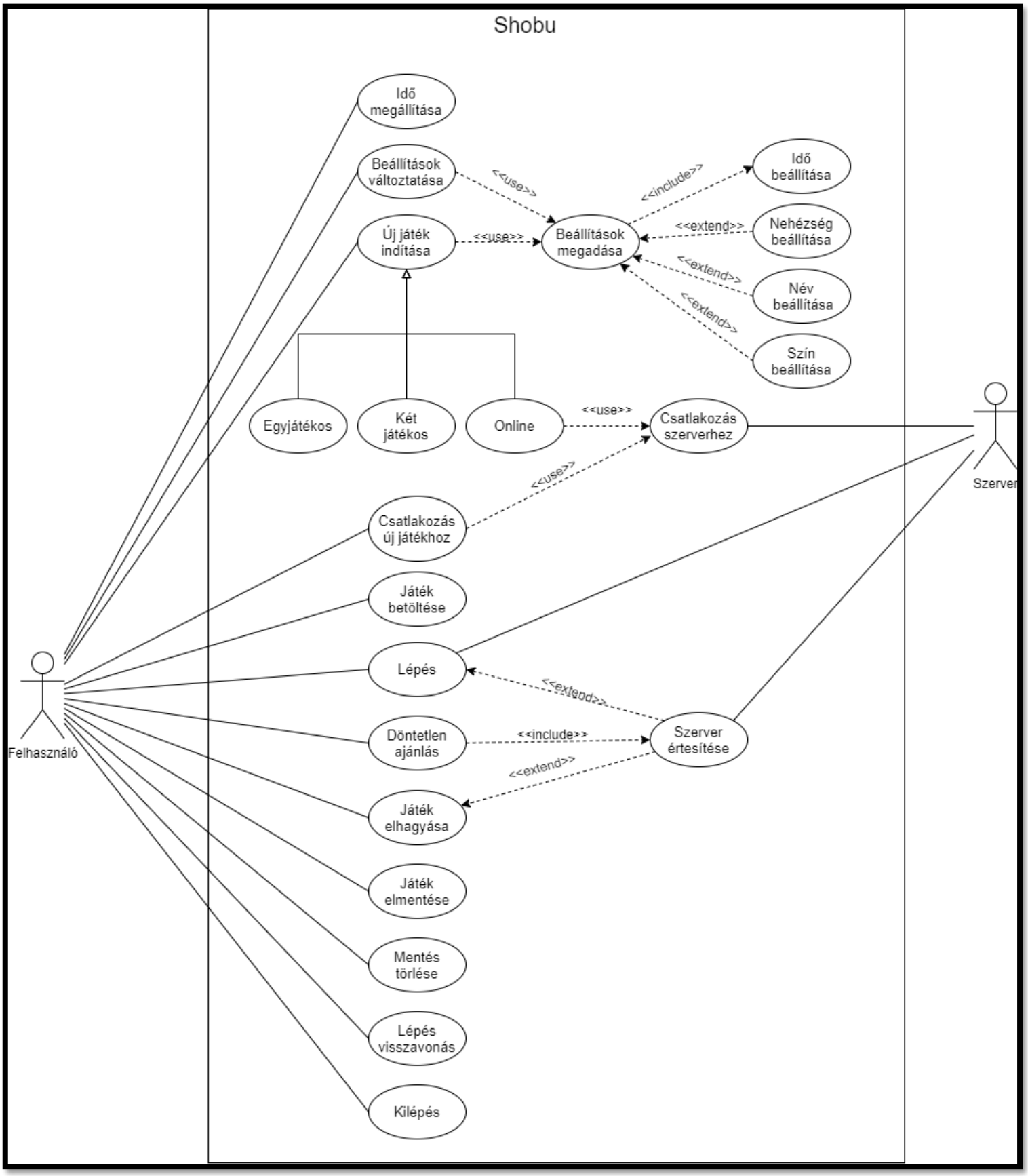
Helyi játék esetén biztosítani kell a lépések visszavonhatóságát is, valamint lehetőséget kell adni a játék elmentésére és betöltésére is.

A hálózati játék megvalósítása érdekében létre kell hozni egy szervert is, amihez a felhasználó csatlakozhat. Itt létrehozhat új játékokat vagy csatlakozhat egy már létrehozott játékhoz. Két játékos esetén a szervernek továbbítania kell a két játékos lépéseit. Biztosítania kell továbbá a döntetlenben való kiegyezés lehetőségét, valamint az egyik játékos idő előtti távozása esetén értesítenie kell a megmaradt játékost korai győzelméről, és le kell zárnia a játékot.

3.1.2. Funkcionális leírás

3.1.2.1. Használati esetdiagram

A feladat leírása alapján a 25. ábra használati esetdiagramján látható funkciókat kell megvalósítanunk.



25. ábra: Használati esetdiagram

3.1.2.2. Felhasználói történetek

Az esetdiagramnál részletesebben a következő, felhasználói történeteket tartalmazó táblázattal adhatjuk meg.

Be- és kilépés:

	Felhasználói eset	Leírás	
1	Alkalmazás indítása	GIVEN:	Az alkalmazás telepítve van
		WHEN:	Alkalmazás indítása
		THEN:	Alkalmazás elindul, főmenü megjelenik
2	Kilépés	GIVEN:	Alkalmazás fut
		WHEN:	Kilépési szándék
		THEN:	Alkalmazás befejezése

Főmenü:

	Felhasználói eset	Leírás	
3	Új játék indítása	GIVEN:	Főmenü
		WHEN:	„New Game” gomb lenyomása
		THEN:	Játék beállítások ablak felugrik, Alapbeállítások: „világos szín, közepes nehézség, időlimit nincs
4	Játék betöltése	GIVEN:	Főmenü
		WHEN:	„Load Game” gomb lenyomása
		THEN:	Betöltés ablak felugrik, megjeleníti a betölthető játékok listáját, első ki van választva, amennyiben létezik.
5	Többjátékos játék kezdeményezése	GIVEN:	Főmenü
		WHEN:	„Multiplayer” gomb lenyomása
		THEN:	Főmenü eltűnik, többjátékos menü megjelenik
6	Kilépés (menüből)	GIVEN:	Főmenü
		WHEN:	„Exit” gomb lenyomása
		THEN:	Alkalmazás befejezése

Egyjátékos beállítások:

	Felhasználói eset	Leírás	
7	Új játék nehézségi fokozat emelése	GIVEN:	Főmenü, játék beállítások ablak, a kiválasztott nehézség nem a legnehezebb
		WHEN:	Nehézségi fokozat növelő gomb lenyomása
		THEN:	A kiválasztott nehézség eggyel emelkedik. A nehézség állító gombok elérhetősége frissül
8	Új játék nehézségi fokozat csökkentése	GIVEN:	Főmenü, játék beállítások ablak, a kiválasztott nehézség nem a legkönnyebb
		WHEN:	Nehézségi fokozat csökkentő gomb lenyomása
		THEN:	A kiválasztott nehézség eggyel csökken. A nehézség állító gombok elérhetősége frissül
9	Új játék időlimit bekapcsolása	GIVEN:	Főmenü, játék beállítások ablak, időlimit ki van kapcsolva
		WHEN:	„Time” doboz megjelölése
		THEN:	Időlimit és gombjai elérhetővé válnak.
10	Új játék időlimit kikapcsolása	GIVEN:	Főmenü, játék beállítások ablak, időlimit be van kapcsolva
		WHEN:	„Time” doboz jelölésének eltávolítása
		THEN:	Időlimit és gombjai elérhetetlenné válnak
11	Új játék időlimit növelés	GIVEN:	Főmenü, játék beállítások ablak, időlimit be van kapcsolva, kiválasztott időlimit nem a legmagasabb
		WHEN:	Időlimit növelő gomb lenyomása
		THEN:	A kiválasztott időlimit eggyel hosszabb kategóriába lép. Az időlimit gombjainak elérhetősége frissül
12	Új játék időlimit csökkentése	GIVEN:	Főmenü, játék beállítások ablak, időlimit be van kapcsolva, kiválasztott időlimit nem a legalacsonyabb
		WHEN:	Időlimit csökkentő gomb lenyomása
		THEN:	A kiválasztott időlimit eggyel rövidebb kategóriába lép. Az időlimit gombjainak elérhetősége frissül
13	Új játék játékos szín megváltoztatása	GIVEN:	Főmenü, játék beállítások ablak,
		WHEN:	Színváltó gomb lenyomása
		THEN:	A kiválasztott szín megváltozik (sötétről világosra, vagy világosról sötétre)
14	Új játék létrehozásának felfüggesztése	GIVEN:	Főmenü, játék beállítások ablak
		WHEN:	„Cancel” gomb lenyomása
		THEN:	Játék beállítások ablak eltűnik (nem indul új játék)

15	Új játék létrehozása	GIVEN:	Főmenü, játék beállítások ablak
		WHEN:	„Start” gomb lenyomása
		THEN:	Főmenü és játék beállítások ablak eltűnnek, a játék megkezdődik a megadott beállításokkal

Betöltés ablak:

	Felhasználói eset	Leírás	
16	Mentett játék kiválasztása	GIVEN:	Főmenü, betöltés ablak, elérhető nem-kiválasztott mentés
		WHEN:	Mentett játékra kattintás
		THEN:	A kattintással kiválasztott mentés lesz az új kiválasztott játék
17	Játék betöltésének elvetése	GIVEN:	Főmenü, betöltés ablak
		WHEN:	„Cancel” gomb lenyomása
		THEN:	Betöltés ablak bezáródik (nem töltődik be mentett játék)
18	Mentett játék indítása	GIVEN:	Főmenü, betöltés ablak, kiválasztott mentés
		WHEN:	„Ok” gomb lenyomása
		THEN:	Betöltés ablak eltűnik, a kiválasztott játék betöltődik
19	Mentett játék törlése	GIVEN:	Főmenü, betöltés ablak, kiválasztott mentés
		WHEN:	„Delete” gomb lenyomása, törlési szándék megerősítése
		THEN:	A kiválasztott mentés törlésre kerül, a mentések listája frissül, a legelső elem kiválasztódik, amennyiben létezik

Többjátékos menü:

	Felhasználói eset	Leírás	
20	Helyi kétszemélyes játék indítása	GIVEN:	Többjátékos menü
		WHEN:	„Hotseat” gomb lenyomása
		THEN:	Játék beállítások ablak felugrik. Alapbeállítások: időlimit nincs
21	Csatlakozás online játékhoz kezdeményezés	GIVEN:	Többjátékos menü
		WHEN:	„Join Online Game” gomb lenyomása
		THEN:	Elérhető online játékok ablak felugrik, elérhető online játékokat megjeleníti, első ki van választva. Ha nincs elérhető online játék, a játékos erről üzenetet kap.

22	Saját online játék kezdeményezése	GIVEN:	Többjátékos menü
		WHEN:	„Create Online Game” gomb lenyomása
		THEN:	Játék beállítások ablak felugrik, Alapbeállítások: „ShobuGame” név, világos szín, időlimit nincs
23	Visszatérés a főmenüre	GIVEN:	Többjátékos menü
		WHEN:	„Back To Main Menu” gomb lenyomása
		THEN:	Többjátékos menü eltűnik, főmenü megjelenik

Online játékok ablak:

	Felhasználói eset	Leírás	
24	Online játék kiválasztása	GIVEN:	Többjátékos menü, online játékok ablak, elérhető nem-kiválasztott online játék
		WHEN:	Online játékra kattintás
		THEN:	A kattintással kiválasztott online játék lesz az új kiválasztott játék
25	Online játékok frissítése	GIVEN:	Többjátékos menü, online játékok ablak
		WHEN:	„Reload” gomb lenyomása
		THEN:	Online játékok frissülnek, első ki van választva. Ha nincs elérhető online játék, a játékos erről üzenetet kap.
26	Csatlakozás online játékhoz	GIVEN:	Többjátékos menü, online játékok ablak, van kiválasztott online játék
		WHEN:	„Ok” gomb lenyomása
		THEN:	Online játékok ablak eltűnik, a váróterem megjelenik. Amennyiben az online játék már elérhetetlen, a felhasználó erről értesítést kap, és visszatér a többjátékos menübe. Ha elérhető, a szerver visszajelzése után megkezdődik a játék
27	Online játék kiválasztásának felfüggesztése	GIVEN:	Többjátékos menü, online játékok ablak
		WHEN:	„Cancel” gomb lenyomása
		THEN:	Elérhető online játékok ablak eltűnik

Online játék beállítások ablak:

	Felhasználói eset	Leírás	
28	Új játék névváltoztatás	GIVEN:	Többjátékos menü, játék beállítások ablak
		WHEN:	Szövegdoboz szerkesztése billentyűzetről
		THEN:	A beállított név megváltozik

29	Online játék időlimit bekapcsolása	GIVEN:	Többjátékos menü, játék beállítások ablak, időlimit ki van kapcsolva
		WHEN:	„Time” doboz megjelölése
		THEN:	Időlimit és gombjai elérhetővé válnak
30	Online játék időlimit kikapcsolása	GIVEN:	Többjátékos menü, játék beállítások ablak, időlimit be van kapcsolva
		WHEN:	„Time” doboz jelölésének eltávolítása
		THEN:	Időlimit és gombjai elérhetetlenné válnak.
31	Online játék időlimit növelés	GIVEN:	Többjátékos menü, játék beállítások ablak, időlimit be van kapcsolva
		WHEN:	Időlimit növelő gomb lenyomása
		THEN:	A kiválasztott időlimit eggyel hosszabb kategóriába lép. Az időlimit gombjainak elérhetősége frissül.
32	Online játék időlimit csökkentése	GIVEN:	Többjátékos menü, játék beállítások ablak, időlimit be van kapcsolva, kiválasztott időlimit nem a legalacsonyabb
		WHEN:	Időlimit csökkentő gomb lenyomása
		THEN:	A kiválasztott időlimit eggyel rövidebb kategóriába lép. Az időlimit gombjainak elérhetősége frissül
33	Online játék játékos szín kiválasztása	GIVEN:	Saját online játék menü
		WHEN:	Színváltó gomb lenyomása
		THEN:	A kiválasztott szín megváltozik (sötétről világosra, vagy világosról sötétre)
34	Online játék létrehozásának felfüggesztése	GIVEN:	Többjátékos menü, játék beállítások ablak
		WHEN:	„Cancel” gomb lenyomása
		THEN:	Játék beállítások ablak eltűnik
35	Online játék létrehozása	GIVEN:	Többjátékos menü, játék beállítások ablak
		WHEN:	„Ok” gomb lenyomása
		THEN:	Játék beállítások ablak eltűnik, váróterem megjelenik

Váróterem:

	Felhasználói eset	Leírás	
36	Online ellenfélre való várakozás felfüggesztése	GIVEN:	Váróterem
		WHEN:	Távozó gomb lenyomása
		THEN:	Váróterem eltűnik, főmenü megjelenik

37	Online játék megkezdése	GIVEN:	Váróterem
		WHEN:	Szerver jelzi a játék kezdetét
		THEN:	Váróterem eltűnik, a játék megkezdődik

A játék:

	Felhasználói eset	Leírás	
38	Passzív lépés: kavics kiválasztás	GIVEN:	Folyamatban lévő játék, a felhasználó a soron következő játékos, még nincs megkezdve lépés
		WHEN:	Saját kavicsra kattintás az egyik hazai táblán
		THEN:	A kavics kiválasztásra kerül, a mozgatásával elérhető lépések megjelölődnek
39	Passzív lépés: kavics mozgatása	GIVEN:	Folyamatban lévő játék, a felhasználó a soron következő játékos, van kiválasztott kavics a passzív lépéshez
		WHEN:	A kiválasztott kavics által egy elérhető mezőre kattintás
		THEN:	A mező kiválasztásra kerül, az agresszív lépéshez elérhető kavicsok megjelölődnek
40	Agresszív lépés: kavics kiválasztás	GIVEN:	Folyamatban lévő játék, a felhasználó a soron következő játékos, van passzív lépés
		WHEN:	Egy (a passzív lépés alapján) elérhető kavicsra kattintás
		THEN:	A kiválasztott kavicsok a megfelelő mezőre kerülnek. Ha az agresszív kavics útja keresztezi az ellenfél egyik kavicsát, azt odébb tolja, vagy lelöki a tábláról. Amennyiben a játék nem ér véget, az ellenfél sorra kerül.
41	Lépés megszakítása	GIVEN:	Folyamatban lévő játék, a felhasználó a soron következő játékos, lépés folyamatban van
		WHEN:	Megszakító gomb lenyomása
		THEN:	A lépés korábbi beállításai eltörlődnek
42	Játék vége	GIVEN:	Folyamatban lévő játék
		WHEN:	Egy lépést követően az egyik táblán nem marad az egyik játékosnak kavicsa, a soron következő játékosnak nincs lépése, vagy lejár az idő
		THEN:	Játék vége ablak felugrik, megjeleníti a győztest
43	Játék elhagyása	GIVEN:	Folyamatban lévő játék
		WHEN:	Távozó gomb lenyomása
		THEN:	A játék véget ér, program visszatér a főmenübe

Csak online játék esetén:

	Felhasználói eset	Leírás	
44	Döntetlen felajánlása	GIVEN:	Folyamatban lévő online játék
		WHEN:	Döntetlen gomb lenyomása
		THEN:	Az ellenfél megkapja az ajánlatot
45	Döntetlen elfogadása	GIVEN:	Folyamatban lévő online játék, az ellenfél döntetlent ajánlott
		WHEN:	Döntetlen gomb lenyomása
		THEN:	A játék döntetlennel véget ér
46	Döntetlen elutasítása	GIVEN:	Folyamatban lévő online játék, döntetlent ajánlat
		WHEN:	Szabályos lépés
		THEN:	Az ajánlat eltűnik mindkét játékosnál

Csak offline játék esetén:

	Felhasználói eset	Leírás	
47	Idő megállítása	GIVEN:	Folyamatban lévő offline játék, óra aktív
		WHEN:	Óra melletti idő gomb lenyomása
		THEN:	Az idő mérése felfüggesztésre kerül, az idő gomb ikonja megváltozik
48	Idő újraindítása	GIVEN:	Folyamatban lévő offline játék, óra aktív
		WHEN:	Óra melletti idő újraindító gomb lenyomása
		THEN:	Az idő mérése folytatódik, az idő gomb ikonja megváltozik
49	Játék mentése	GIVEN:	Folyamatban lévő offline játék
		WHEN:	Mentés gomb lenyomása
		THEN:	A mentés ablak felugrik
50	Lépés visszavonása	GIVEN:	Folyamatban lévő egyszemélyes játék, elérhető korábbi játékállapot
		WHEN:	Visszavonás gomb lenyomása
		THEN:	A játék visszaáll az eggyel korábbi állapotra, amikor a felhasználó volt soron
51	Visszavonás visszavonása	GIVEN:	Folyamatban lévő egyszemélyes játék, elérhető korábban visszavont játékállapot
		WHEN:	Lépés ismétlő gomb lenyomása
		THEN:	A játék visszaáll a korábban visszavont állapotra, amikor a felhasználó volt soron

52	Beállítások módosítása	GIVEN:	Folyamatban lévő egyszemélyes játék
		WHEN:	Beállítások gomb lenyomása
		THEN:	A játék közbeni beállítások ablak felugrik

Csak egyszemélyes játék esetén:

	Felhasználói eset	Leírás	
53	Ellenfél lép	GIVEN:	Folyamatban lévő egyszemélyes játék, nehézségi szint be van állítva
		WHEN:	A soron következő játékos a számítógép
		THEN:	A számítógép a nehézségi szintnek megfelelően lép egyet a felhasználó ellen

Beállítások módosítása játék közben:

	Felhasználói eset	Leírás	
54	Aktív játék nehézségi fokozat emelése	GIVEN:	Folyamatban lévő egyszemélyes játék, játék beállítások ablak, a kiválasztott nehézség nem a legnehezebb
		WHEN:	Nehézségi fokozat növelő gomb lenyomása
		THEN:	A kiválasztott nehézség eggyel emelkedik. A nehézség állító gombok elérhetősége frissül
55	Aktív játék nehézségi fokozat csökkentése	GIVEN:	Folyamatban lévő egyszemélyes játék, játék beállítások ablak, a kiválasztott nehézség nem a legkönnyebb
		WHEN:	Nehézségi fokozat csökkentő gomb lenyomása. A nehézség állító gombok elérhetősége frissül
		THEN:	A kiválasztott nehézség eggyel csökken
56	Aktív játék időlimit bekapcsolása	GIVEN:	Folyamatban lévő helyi játék, játék beállítások ablak, időlimit ki van kapcsolva
		WHEN:	„Time” doboz megjelölése
		THEN:	Időlimit elérhetővé válik. Az időlimit állító gombok elérhetősége frissül
57	Aktív játék időlimit kikapcsolása	GIVEN:	Folyamatban lévő helyi játék, játék beállítások ablak, időlimit be van kapcsolva
		WHEN:	„Time” doboz jelölésének eltávolítása
		THEN:	Időlimit elérhetetlenné válik. Az időlimit állító gombok elérhetősége frissül
58	Aktív játék időlimit növelés	GIVEN:	Folyamatban lévő helyi játék, játék beállítások ablak, időlimit be van kapcsolva
		WHEN:	Időlimit növelő gomb lenyomása
		THEN:	A kiválasztott időlimit eggyel hosszabb kategóriába lép. Az időlimit állító gombok elérhetősége frissül

59	Aktív játék időlimit csökkentése	GIVEN:	Folyamatban lévő helyi játék, játék beállítások ablak, időlimit be van kapcsolva, kiválasztott időlimit nem a legalacsonyabb
		WHEN:	Időlimit csökkentő gomb lenyomása
		THEN:	A kiválasztott időlimit eggyel rövidebb kategóriába lép. Az időlimit állító gombok elérhetősége frissül
60	Aktív játék játékos szín megváltoztatása	GIVEN:	Folyamatban lévő egyszemélyes játék, játék beállítások ablak
		WHEN:	Színváltó gomb lenyomása
		THEN:	A kiválasztott szín megváltozik (sötétről világosra, vagy világosról sötétre)
61	Aktív játék létrehozásának felfüggesztése	GIVEN:	Folyamatban lévő helyi játék, játék beállítások ablak
		WHEN:	„Cancel” gomb lenyomása
		THEN:	Játék beállítások ablak eltűnik (, nem változnak a beállítások)
62	Aktív játék beállításainak frissítése	GIVEN:	Folyamatban lévő helyi játék, játék beállítások ablak
		WHEN:	„Start” gomb lenyomása
		THEN:	Játék beállítások ablak eltűnik, a játék beállításai megváltoznak

Mentés ablak:

	Felhasználói eset	Leírás	
63	Mentett játék kiválasztása	GIVEN:	Játék folyamatban, mentés ablak, elérhető nem-kiválasztott mentés
		WHEN:	Mentett játékra kattintás
		THEN:	A kiválasztott mentés neve lesz a kiválasztott név
64	Játék mentésének elvetése	GIVEN:	Játék folyamatban, mentés ablak
		WHEN:	„Cancel” gomb lenyomása
		THEN:	Mentés ablak bezáródik (nem készül új mentés)
65	Játék mentése	GIVEN:	Játék folyamatban, mentés ablak, kiválasztott név
		WHEN:	„Ok” gomb lenyomása
		THEN:	Mentés ablak eltűnik, játékállás elmentődik a kiválasztott néven

66	Mentett játék törlése	GIVEN:	Játék folyamatban, mentés ablak, kiválasztott mentés
		WHEN:	„Delete” gomb lenyomása, törlési szándék megerősítése
		THEN:	A kiválasztott mentés törlésre kerül, a mentések listája frissül, a legelső elem kiválasztódik, amennyiben létezik
67	Mentés nevének állítása	GIVEN:	Játék folyamatban, mentés ablak
		WHEN:	Szövegdoboz szerkesztése billentyűzetről
		THEN:	Az új mentés neve megváltozik

3.1.3. Nem-funkcionális leírás

A programnak a következő nem-funkcionális követelményeknek kell eleget tennie:

- **Hatékonyság**: Gépi ellenfél esetén a számítógép pár másodperc alatt adjon válaszlépést a játékosnak!
- **Biztonság**: A szerver távolítsa el a szabálytalan lépésekkel próbálkozó játékosokat!
- **Hordozhatóság**: A programnak legyen egy telepítője a Windows operációs rendszerrel rendelkező számítógépek számára!
- **Megbízhatóság**: A program ne legyen érzékeny a hibás felhasználói bemenetre! Hagyja őket figyelmen kívül, vagy jelezze hibaüzenettel a problémát!

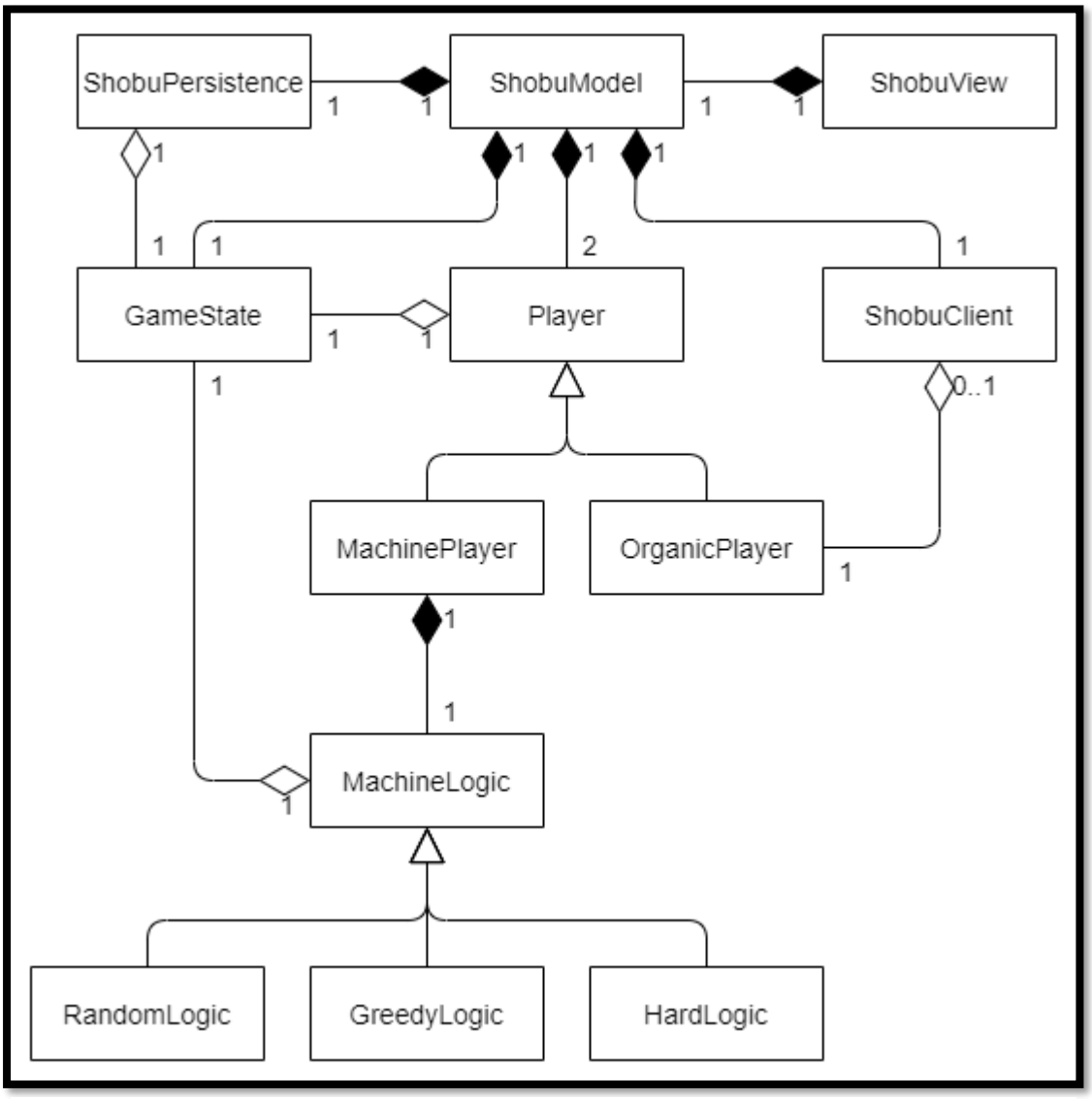
3.2. Tervezés

3.2.1. Szoftver architektúra

A program a modell-nézet-perzisztencia architektúrát követi, így külön osztály gondoskodik a játéklogika működéséért (ShobuModel), a megjelenésért és felhasználóval való kapcsolattartásért (ShobuView), valamint a korábbi állapotok és mentések kezeléséért (ShobuPersistence). Ezen felül online játék esetén egy külön vezérlő osztály felel a szerverrel való kapcsolattartásért (ShobuClient). Említésre méltó még a szerver is, amely összeköti az egyes játékosokat, bár azt már önálló programnak is értelmezhetjük.

3.2.2. Modell

A modell osztályainak kapcsolatát a 26. ábrán lévő UML diagramon láthatjuk. A program méretére való tekintettel az olvashatóság érdekében az egyes osztályokat csak később tárgyaljuk részletesebben.



26. ábra: A modell osztályainak kapcsolata

3.2.2.1. ShobuModel

A ShobuModel feladata a játéklogika összefogása. Ez figyelmezteti a nézetet, ha frissíteni kell valamit a megjelenítésen, és ez tartja a kapcsolatot a játékos osztályokkal, a perzisztencia és vezérlő rétegekkel, és ez felel azért, hogy a beállított játékmód szerint folyjon a játék.

Fontosabb függvények:

- **changeTime():** Megállítja, vagy újraindítja az idő mérését.

- **newGame(GameSettings):** új játékot indít a megadott beállításokkal.
- **makeMove():** A felhasználó köre esetén meglépi a beállított lépést, a gépi játékos köre esetén megkezdi a lépés előtti visszaszámlálást. Az egyik játékos (moveMade() szignállal jelzett) lépésének hatására hívjuk meg.
- **resetMove():** Eltávolítja a folyamatban lévő lépés beállításait.
- **endGame():** Végleg leállítja az idő mérését az új játék megkezdéséig
- **changeSettings(GameSettings):** Játék közben megváltoztatja az aktuális beállításokat.
- **saveGame(QString):** Elmenti az adott játékállást a megadott néven
- **loadGame(QString):** Betölti a megadott néven elmentett játékállást, amennyiben lehetséges.
- **undoStep():** Visszaállít egy korábbi játékállást játékmódtól függően.
- **redoStep():** Visszaállít egy korábban visszavont játékállást játékmódtól függően.
- **hasUndo():** Jelzi, hogy létezik-e korábbi visszaállítható játékállás.
- **hasRedo():** Jelzi, hogy létezik-e korábban visszavont játékállás, amit vissza lehet állítani.

27. ábra: ShobuModel osztály

- **connectToServer(QString):** Megpróbál csatlakozni a megadott címen található szerverhez. A kérést továbbítja a _client osztálynak.
- **disconnectFromServer():** Lecsatlakozik a szerverről. A kérést továbbítja a _client osztálynak.
- **createOnlineGame(GameSettings):** Megkísérel készíteni egy online játékot a megadott beállításokkal. A kérést továbbítja a _client osztálynak.
- **joinOnlineGame(QString):** Megkísérel csatlakozni egy meglévő online játékhoz. A kérést továbbítja a _client osztálynak.
- **offerDraw():** Döntetlent ajánl az ellenfélnek online játék esetén. A kérést továbbítja a _client osztálynak.
- **initializeGame():** Megteszi a játék elindításához szükséges előkészületeket a meglévő beállításokkal.
- **getStepSize():** Visszaadja az aktuális játékmódhoz tartozó lépésvisszavonás méretét. Mindig olyan számot ad, hogy a visszavonást követően a felhasználó legyen a soron következő játékos.
- **applySettings():** A játékot az eltárolt beállításokhoz igazítja, és odaadja a kört a következő játékosnak.
- **applyMove():** Meglépi az eltárolt lépést, és értesíti a nézetet a változásról.
- **onGameStarted(GameSettings):** Online játékba kezd a megadott beállításokkal. A _client gameStarted() szignálja hatására fut le.
- **onDrawOffer():** Továbbítja az ajánlatot a nézetnek. A _client drawOffer() szignáljának hatására fut le.
- **tick():** Az idő múlásának hatásaiért felel. Időlimites játékban ez csökkenti a hátralévő időt. Egyszemélyes játék esetén ez felel a gépi játékos lépésének késleltetéséért. A _timer timeout() szignáljának hatására fut le.
- **stepGame():** Szignál, ami jelzi egy lépés végét. A nézet ennek hatására frissül, és szólítja fel a soron következő játékost a lépésre.
- **boardChange():** Szignál, ami jelzi, hogy valami megváltozott a játékálláson. A nézetnek ennek hatására frissülnie kell.

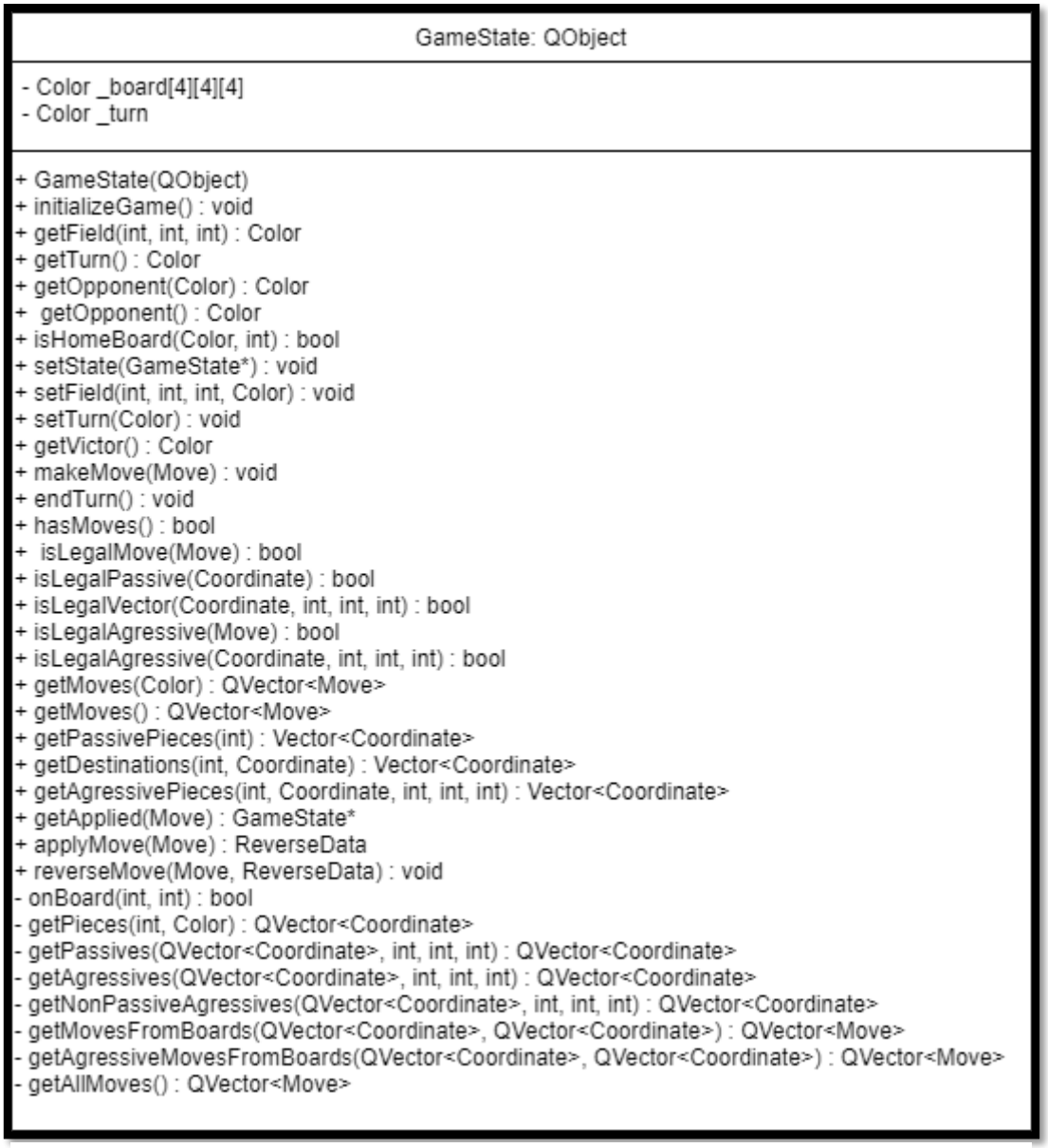
- **timeIsPassing():** Szignál, ami jelzi az idő múlását. Időlimites játék esetén a nézetnek ennek hatására frissítenie kell a hátralévő időt.
- **gotOnlineList(QVector<GameSettings>):** Szignál, ami jelzi, hogy a szerver elküldte az elérhető játékok listáját. A szignál tartalmazza a megérkezett listát. A _client gamesArrived() szignálja hatására küldjük ki.
- **gotServerMessage(QString):** Szignál, ami jelzi, hogy a szerver üzenetet küldött. Ez a szignál mindig a játék végét jelzi. A _client serverMessage() szignálja hatására küldjük ki.
- **onlineGameStarted(GameSettings):** Szignál, ami jelzi, hogy a szerver új játékot indított, amiben részt veszünk.

3.2.2.2. GameState

A GameState osztály tartalmazza az aktuális játékállást, és a játékszabályokat. Többek között a játékállás frissítésére, egy lépés szabályosságának meghatározására és a szabályos lépések felsorolására képes.

Fontosabb függvények:

- **initializeGame():** Felállítja a játék kezdeti állapotát. A fekete kavicsok föl, a fehérek le kerülnek, és a fehér a kezdőjátékos.
- **getOpponent(Color):** Visszaadja az adott szín ellentétét. A függvény statikus, így nem igényel példányosítást a meghívása.
- **getOpponent():** Visszaadja az aktuális játékos ellenfelét.
- **isHomeBoard(Color, int):** Meghatározza, hogy az adott színű játékosnak hazai táblája-e az adott indexű tábla. A függvény statikus.
- **getVictor():** Visszaadja a győztes játékos színét. Ha a játékot még az egyik fél sem nyerte meg, az EMPTY értéket adja vissza.
- **makeMove(Move):** Végrehajtja az adott lépést, amennyiben az szabályos.
- **endTurn():** Átadja a kört a következő játékosnak.
- **hasMoves():** Visszaadja, hogy az adott játékosnak vannak-e elérhető lépései.
- **isLegalMove(Move):** Meghatározza, hogy az adott lépés szabályos-e.



28. ábra: GameState osztály

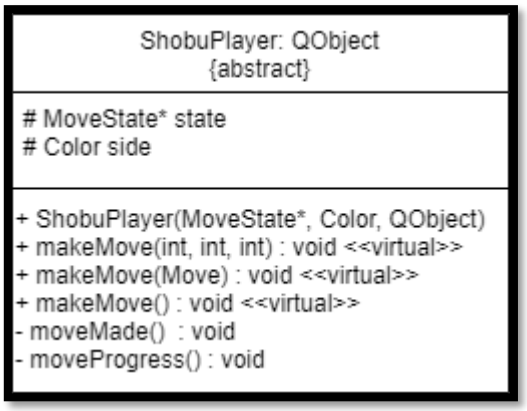
- **isLegalPassive(Coordinate):** Meghatározza, hogy az adott koordinátán szabályosan választható passzív kavics van-e.
- **isLegalVector(Coordinate, int, int, int):** Meghatározza, hogy a három számmal megadott vektor szabályosan választható vektor-e az adott koordinátán lévő passzív kavicshoz.
- **isLegalAgressive(Move):** Meghatározza, hogy az adott lépéshez szabályos agresszív kavics van-e kiválasztva.
- **isLegalAgressive(Coordinate, int, int, int):** Meghatározza, hogy a három számmal megadott vektorhoz szabályosan választható agresszív kavicsot tartalmaz-e az adott koordináta.

- **getMoves(Color):** Visszaadja az adott szín összes szabályos lépését.
- **getMoves():** Visszaadja a soron következő játékos összes szabályos lépését.
- **getPassivePieces(int):** Visszaadja az aktuális játékos számára szabályosan elérhető passzív kavicsok koordinátáit az adott indexű táblán.
- **getDestinations(int, Coordinate):** Visszaadja az adott koordinátán megtalálható kavicshoz elérhető célállomások koordinátáit (ahová egy szabályos lépéssel eljuthat).
- **getAgressivePieces(int, Coordinate, int, int, int):** Visszaadja az adott indexű táblán a megadott passzív lépéshez szabályos agresszív kavicsok koordinátáit.
- **getApplied(Move):** Visszaad egy új GameState osztályt, amit úgy kapunk meg, hogy a jelenlegin végrehajtjuk a megadott lépést.
- **applyMove(Move):** Meglépi a megadott lépést. Szabálytalan lépés esetén hibát dob.
- **reverseMove(Move, ReverseData):** Visszaállítja a megadott lépés előtti állapotot.
- **onBoard(int, int):** Ellenőrzi, hogy a megadott koordináták rajta lehetnek-e egy táblán.
- **getPieces(int, Color):** Visszaadja a megadott színű kavicsok koordinátáit az adott indexű táblán.
- **getPassives(QVector<Coordinate>, int, int, int):** Visszaadja a megadott koordináták közül azokat, amik szabályos passzív lépést tehetnek a megadott vektorral.
- **getAgressives(QVector<Coordinate>, int, int, int):** Visszaadja a megadott koordináták közül azokat, amik szabályos agresszív lépést tehetnek a megadott vektorral.
- **getNonPassiveAgressives(QVector<Coordinate>, int, int, int):** Visszaadja a megadott koordináták közül azokat, amik szabályos agresszív lépést tehetnek a megadott vektorral, de passzívat nem.
- **getMovesFromBoards(QVector<Coordinate>, QVector<Coordinate>):** Visszaadja a szabályos lépéseket, amit a két különböző tábláról származó kavicsokkal tehetünk.
- **getAgressiveMovesFromBoards(QVector<Coordinate>, QVector<Coordinate>):** Visszaad minden olyan szabályos lépéseket, amit a két különböző tábláról származó kavicsokkal tehetünk, és odébb tolnak egy ellenséges kavicsot.

- **getAllMoves():** Visszaad minden lehetséges lépést. Külön lépésnek számolja azokat a lépéseket is, amelyeknek eredménye megegyezik, de eltér, hogy melyik az agresszív, és melyik a passzív lépés.

3.2.2.3. ShobuPlayer

A ShobuPlayer osztály felel a lépések elkészítéséért. A ShobuModel osztállyal osztozik egy MoveState struktúrán, amibe a lépés aktuális állapotát tárolják. Amennyiben kész van a lépéssel, azt egy szignállal jelzi a modellnek. A ShobuPlayer absztrakt osztály, a lépés elkészítésének módja a gyerekeitől függően változik.



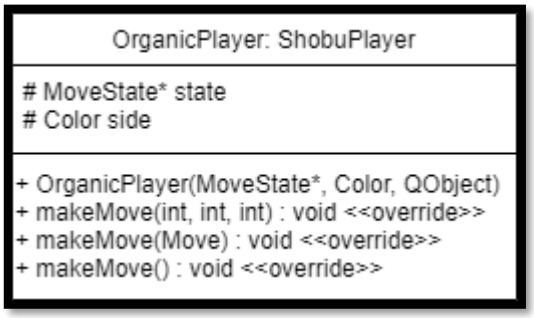
29. ábra: ShobuPlayer osztály

Fontosabb függvények:

- **makeMove():** A lépés elkészítéséért felelős függvény. Működése leszármazottól függően változik
- **makeMove(int, int, int):** A lépés elkészítéséért felelős függvény. Működése leszármazottól függően változik
- **makeMove(Move):** A lépés elkészítéséért felelős függvény. Működése leszármazottól függően változik
- **moveMade():** A szignál, ami jelzi a modell számára, hogy elkészült a lépés.
- **moveProgress():** A szignál, ami jelzi a modell számára, hogy megjelenítendő változás történt, de még nincs kész a lépés

3.2.2.4. OrganicPlayer

Az OrganicPlayer a ShobuPlayer azon leszármazottja, amely egy felhasználótól várja a lépés kiválasztását. Ez megtörténhet darabonként, vagy akár egy kész lépést is el tud fogadni.



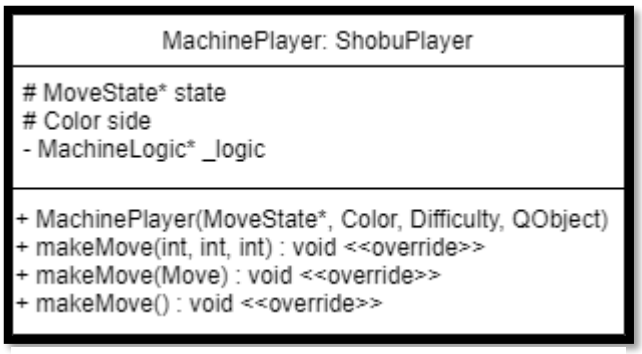
30. ábra: OrganicPlayer osztály

Fontosabb függvények:

- **makeMove(int, int, int):** Megpróbálja a három számot hozzáilleszteni az aktuális lépéshez mint koordinátát, vagy vektort. Ha elkészül a lépés, kibocsátja a moveMade() szignált, ha pedig nem, a moveProgress() szignált.
- **makeMove(Move):** Egy kész lépést kap. Azonnal jelzi a modell számára, hogy kész van a lépés.

3.2.2.5. MachinePlayer

A MachinePlayer a ShobuPlayer azon leszármazottja, amely önállóan alkotja meg a lépést. Ezt a MachineLogic tagja segítségével teszi meg, minek eredménye nehézségi szinttől változik.



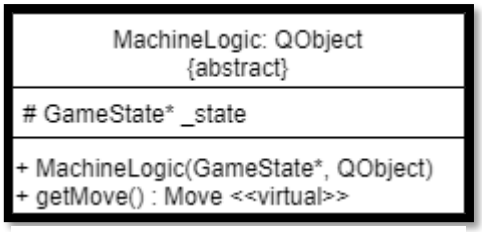
31. ábra: ShobuPlayer osztály

Fontosabb függvények:

- **makeMove():** Elkészíti a lépést. Továbbítja a kérést a MachineLogic osztálynak, majd miután megkapta az eredményt, a moveMade() szignállal jelzi, hogy kész a lépés.

3.2.2.6. MachineLogic

A MachineLogic felel a legjobb lépés kiválasztásáért. A választékot a GameState osztálytól kapja. A MachineLogic absztrakt osztály, így a legjobb lépés kiválasztásának módja a leszármazottjaitól függ.



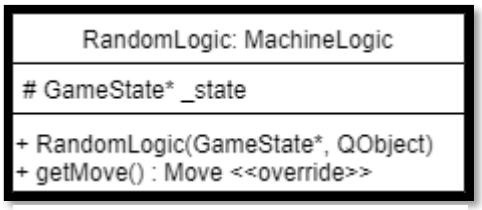
10. ábra: MachineLogic osztály

Fontosabb függvények:

- **getMove():** Visszaadja a legjobb lépést. A legjobb lépés kiválasztásának módja a leszármazottaktól függően változik.

3.2.2.7. RandomLogic

A RandomLogic a MachineLogic azon leszármazottja, amely semmilyen logikát nem alkalmaz a következő lépés megválasztására, csak véletlenszerűen választ az elérhető lépések közül. Ez egy könnyen megverhető gépi ellenfelet eredményez.



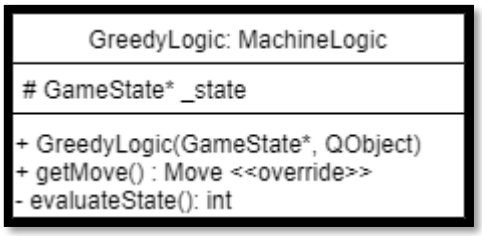
11. ábra: RandomLogic osztály

Fontosabb függvények:

- **getMove():** Véletlenszerűen visszaad egyet az elérhető lépések közül.

3.2.2.8. GreedyLogic

A GreedyLogic a MachineLogic azon leszármazottja, amely a lehető leggyorsabban le akarja ütni az ellenfele bábuit. Ha több választási lehetőség is van, ami a győzelemhez legközelebb álló táblán ütne, azok közül véletlenszerűen választ. A logika jó gépi ellenfél azoknak a játékosoknak, akik már ismerik a szabályokat, de még nincs sok tapasztalatuk a játékkal.



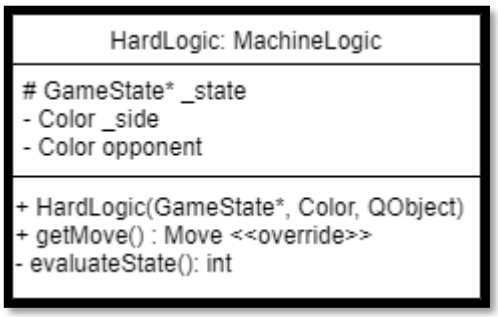
12. ábra: GreedyLogic osztály

Fontosabb függvények:

- **getMove():** Kiválasztja a legnagyobb pontértékkel rendelkező lépést.
- **evaluateState():** Pontértéket rendel az egyes lépések által létrehozott állásokhoz. Ehhez csak az egyes táblákon lévő kavicsok számát veszi figyelembe.

3.2.2.9. HardLogic

A HardLogic a MachineLogic azon leszármazottja, amely a már a kavicsok pozícióját és a saját védelmét is szem előtt tartja a legjobb lépés kiválasztásakor. A logika kihívást jelenthet akár tapasztaltabb játékosoknak is.



13. ábra: HardLogic osztály

Fontosabb függvények:

- **getMove():** Kiválasztja a legnagyobb pontértékkel rendelkező lépést.

- **evaluateState():** Pontértéket rendel az egyes lépések által létrehozott állásokhoz. Ehhez figyelembe veszi a kavicsok pozícióját is.

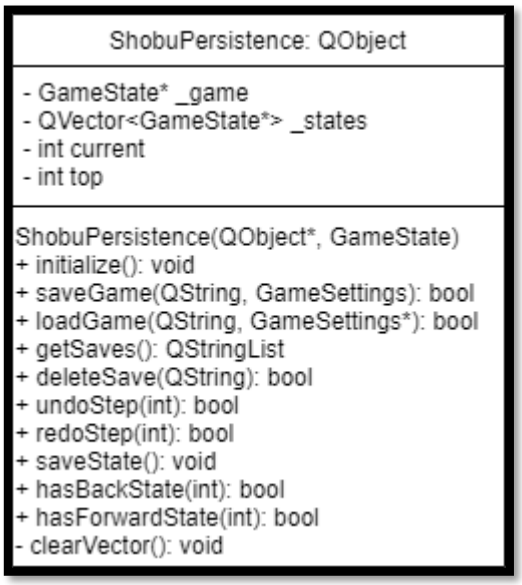
3.2.2.10. ShobuException

A ShobuException osztály jelzi, hogy a program működésében kivétel lépett fel. Ez a QException absztrakt osztály leszármazottja, így az onnan örökölt raise() függvénnyel jelzi a kivételes helyzet előállását. A kivétel természetéről további üzeneteket a setMessage(QString) és getMessage() függvényekkel továbbíthat.

3.2.3. Perzisztencia

3.2.3.1. ShobuPersistence

A ShobuPersistence felel mindenért, ami a korábbi állások visszaállításában szerepet játszhat. Ez gondoskodik a lépések visszavonásáról, a játékállás elmentéséről és betöltéséről, valamint a mentések lekérdezéséről és törléséről.



36. ábra: ShobuPersistence osztály

Fontosabb függvények:

- **initialize():** Letörli az eltárolt korábbi játékállásokat és elmenti a jelenlegit.
- **saveGame(QString, GameSettings):** Elmenti a jelenlegi játékállást a megadott néven.
- **loadGame(QString, GameSettings*):** Betölti a megadott néven elmentett játékállást.
- **getSaves():** Visszaadja az elmentett játékok listáját.
- **deleteSave(QString):** Letörli a megadott néven elmentett játékállást.
- **undoStep(int):** Visszalép egy korábbi játékállásra.
- **redoStep(int):** Visszalép egy korábban visszavont játékállásra.
- **saveState():** Elmenti az aktuális játékállást a korábbi játékállások listájába.
- **hasBackState(int):** Visszaadja, hogy létezik-e visszaállítható játékállás.

- **hasForwardState(int):** Visszaadja, hogy van-e visszaállítható, korábban visszavont játékállás.
- **clearVector():** Kiüríti a korábbi játékállások listáját.

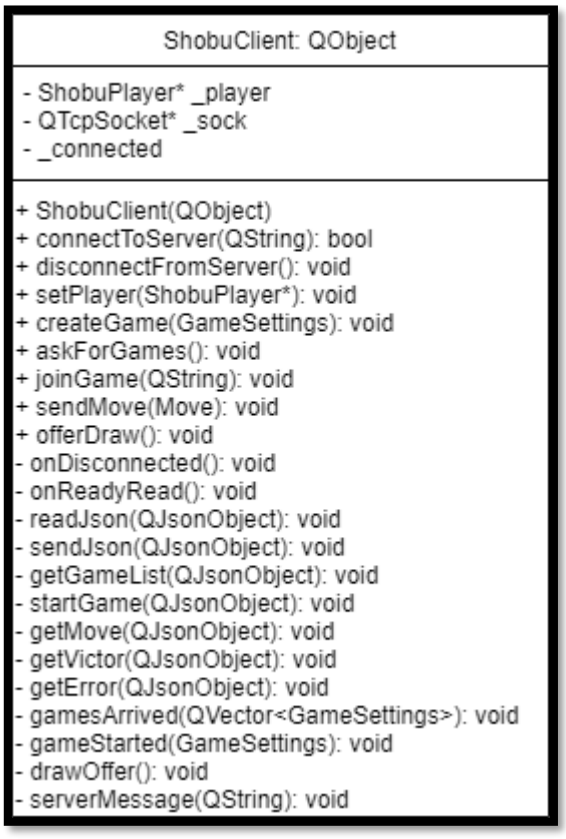
2.3.4. Vezérlő

2.3.4.1. ShobuClient

A ShobuClient osztály tartja a kapcsolatot a szerver és a modell között. Ez felelős a csatlakozásért, az üzenetek elküldéséért és fogadásáért, valamint a távoli játékos lépéseinek közvetítéséért.

Fontosabb függvények:

- **connectToServer(QString):** Csatlakozik a megadott címen található szerverhez. Visszaadja, hogy sikeres volt-e.
- **disconnectFromServer():** Lecsatlakozik a szerverről.
- **setPlayer(ShobuPlayer*):** Hozzárendeli a megadott játékos osztályt a klienshez. Ez a játékos osztály fogja végrehajtani a távoli felhasználó lépéseit.
- **createGame(GameSettings):** Megpróbál létrehozni egy online játékot a megadott beállításokkal.
- **askForGames():** Elkéri a szervertől az elérhető játékok listáját.
- **joinGame(QString):** Megpróbál csatlakozni egy online játékhoz.



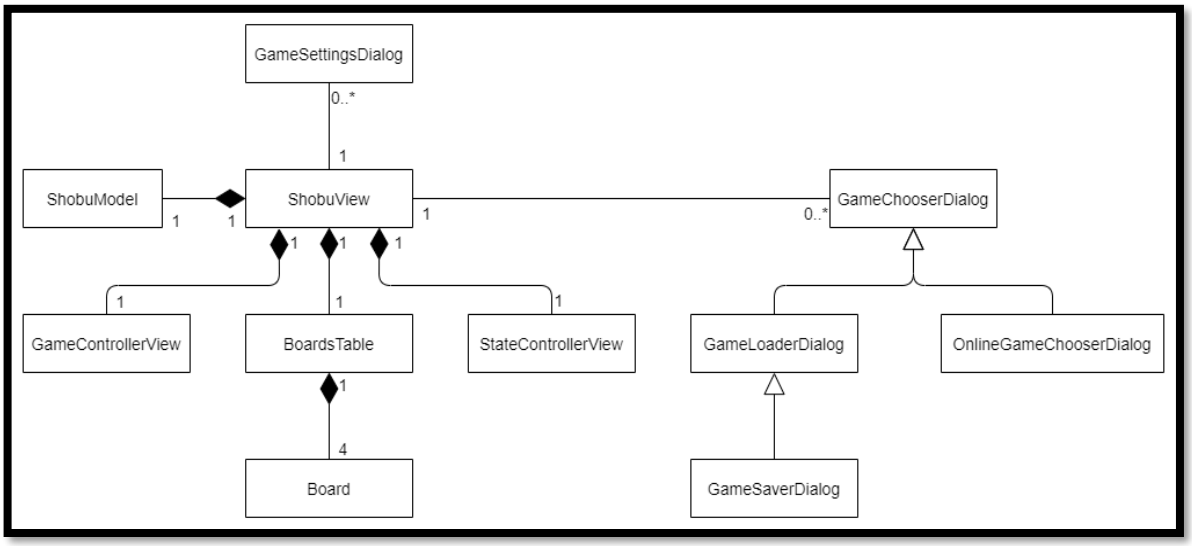
37. ábra: ShobuClient osztály

- **sendMove(Move):** Elküldi a helyi felhasználó lépését a szervernek.
- **offerDraw():** Elküldi a helyi felhasználó döntetlen-ajánlatát a szervernek.
- **onDisconnected():** Értesíti a modellt a lecsatlakozásról, amennyiben nem tőle jött a kérés

- **onReadyRead():** Json formátumúvá alakítja a szerver üzenetét. Új szerverüzenet érkezésének hatására fut le.
- **readJson(QJsonObject):** Kiolvassa a json-ból a szerver üzenetét, és az üzenet típusától függően kezeli azt.
- **sendJson(QJsonObject):** Elküldi a szervernek szánt üzenetet.
- **getGameList(QJsonObject):** Továbbítja a szerver által elküldött listát, ami az elérhető játékokat tartalmazza.
- **startGame(QJsonObject):** Továbbítja a szerver üzenetét, miszerint elkezdődött egy új online játék a felhasználó részvételével.
- **getMove(QJsonObject):** Közli a játékososztállyal a szerver által elküldött lépést.
- **getVictor(QJsonObject):** Kezeli, amikor a szerver elküldi a játék győztesét.
- **getError(QJsonObject):** Kiolvassa a szerver hibaüzenetét a json-ból, és továbbítja a modellnek.
- **gamesArrived(QVector<GameSettings>):** Szignál, ami továbbítja az elérhető játékok listáját.
- **gameStarted(GameSettings):** Szignál, ami továbbítja az újonnan elkezdődött játék beállításait.
- **drawOffer():** Szignál, ami jelzi, hogy az online ellenfél döntetlent ajánlott.
- **serverMessage(QString):** Szignál, ami továbbítja a szerver szöveges üzenetét.

3.2.5. Nézet

A nézet felelős a program megjelenéséért és a felhasználóval való kapcsolattartásért. A nézet osztályainak kapcsolatát a 38 ábrán lévő UML diagramon láthatjuk. A program méretére való tekintettel az olvashatóság érdekében az egyes osztályokat csak később tárgyaljuk részletesebben.



148. ábra: A nézet osztályainak kapcsolata

3.2.5.1. ShobuView

A ShobuView az alkalmazás fő ablaka. A fő ablaknak négy állapota van, a főmenü, a többjátékos menü, a játék és a váróterem. Ez az osztály teszi lehetővé a köztük lévő navigációt, és ezen keresztül folyik minden kommunikáció a felhasználó és a program között.

Fontosabb függvények:

- **resizeEvent(QResizeEvent*):** Az ablak átméretezése esetén meghívott függvény. Gondoskodik róla, hogy az ablak minden részének mérete megfelelően igazodjon az ablak méretéhez.
- **resizeMenu(QLabel*, QVector<QPushButton*>):** A menü elemek méretre igazításáért felelős függvény.
- **resizeGame():** A játékablak méretre igazításáért felelős függvény.
- **resizeWaitRoom():** A váróterem méretre igazításáért felelős függvény.

- **emptyLayout():** Az ablak kiürítéséért felelős függvény. Az új nézet megjelenítése előtt eltávolítja az előző nézet elemeit.
- **setGameLayout():** Megjeleníti a játékablak nézetet.
- **setMainMenuLayout():** Megjeleníti a főmenü nézetet.
- **setMultiMenuLayout():** Megjeleníti a többjátékos nézetet.
- **setWaitRoomLayout():** Megjeleníti a váróterem nézetet.
- **gameOver(Color):** Megjeleníti a győztest és elhagyja a játékablakot.
- **tick():** Frissíti a megjelenített időt.
- **onBoardChange():** Frissíti a játéktáblákat.
- **onStepGame():** Frissíti a játéktáblákat, lekérdezi, hogy véget ért-e a játék, majd ettől függően véget vet a játéknak, vagy jelez a soron következő játékosnak.

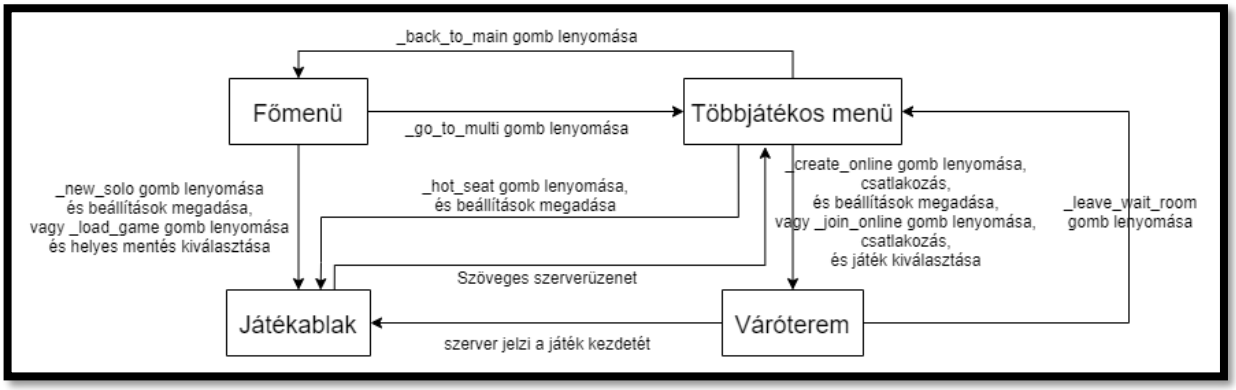
39. ábra: ShobuView osztály

- **newSoloClicked():** Felugrik a beállítások ablak, majd megerősítés után új egyszemélyes játékba kezd.
- **loadGameClicked():** Felugrik a betöltőablak, majd választás esetén új játékba kezd.
- **goToMultiClicked():** Átvált a többjátékos menü nézetre.
- **hotSeatClicked():** Felugrik a betöltőablak, majd választás esetén új kétszemélyes, helyi játékba kezd.

- **joinOnlineClicked():** Felugrik a szerverhez csatlakozó ablak. Sikeres csatlakozás esetén megjelenik a beállítások ablak, majd jóváhagyás után új játékot kér a szervertől, és átmegy a váróterem nézetbe.
- **createOnlineClicked():** Felugrik a szerverhez csatlakozó ablak. Sikeres csatlakozás esetén megjelenik az online játék kiválasztó ablak, majd jóváhagyás után kérvényezi a csatlakozást a szervertől, és átmegy a váróterem nézetbe.
- **backToMainClicked():** Visszatér a főmenü nézetre.
- **leaveWaitRoomClicked():** Elhagyja a várótermet, visszatér a főmenü nézetre.
- **onServerMessage(QString):** Megjeleníti a szerver hibaüzenetét, majd visszatér a többjátékos menü nézethez. A ShobuModel serverMessage() szignálja hatására fut le.

Navigáció a nézetek között:

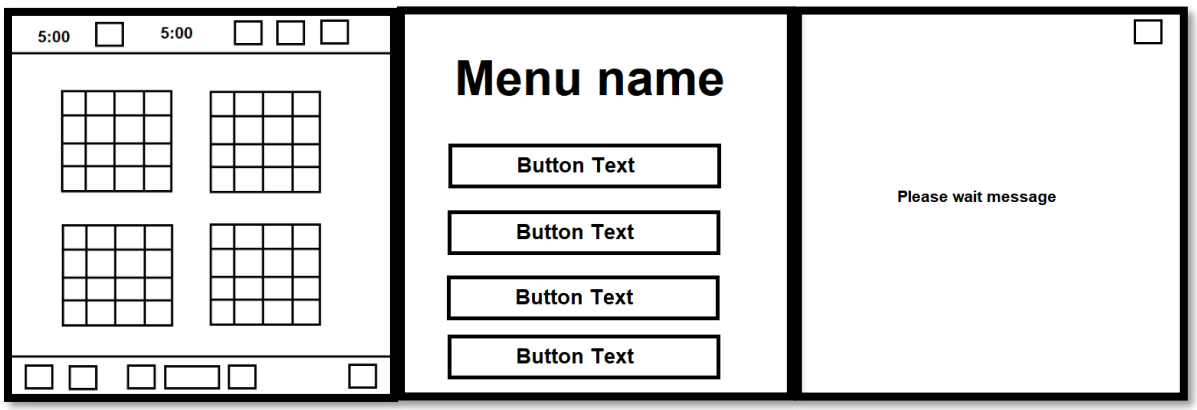
A felhasználó a különböző nézetek között a 40. ábrán látható eseményekkel navigálhat:



40. ábra: Navigáció a nézetek között

Megjelenítés terve:

A megjelenés terve a 41., 42. és 43 ábrákon látható:



41. ábra: Játékablak terve *42. ábra: Menük terve* *43. ábra: Váróterem terve*

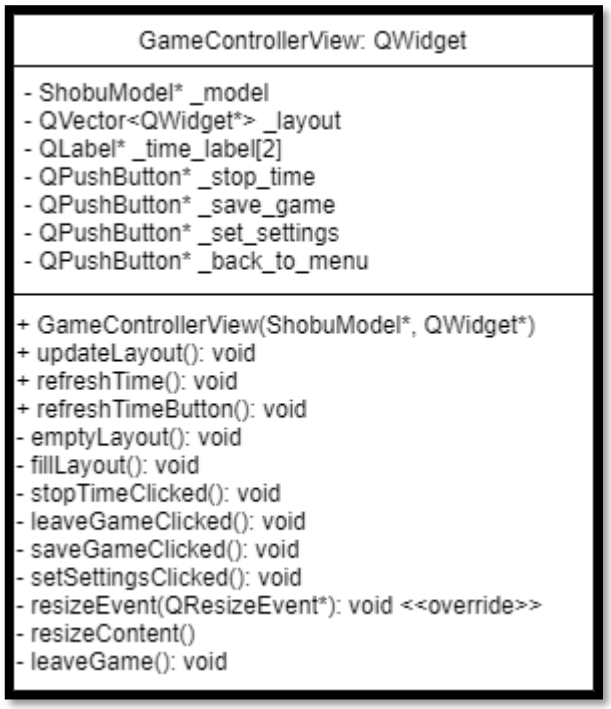
(A nézet gombjaihoz felhasznált ikonok majd az iconarchive.com oldalról származnak.) [4]

3.2.5.2. GameControllerView

A GameControllerView osztály tartalmazza a játékablak fejlécében található gombokat és címkéket. Innen lehet vezérelni azokat a funkciókat, amiknek nincs közvetlen hatásuk az aktuális játékállásra.

Fontosabb függvények:

- **updateLayout():** Frissíti a elemeket az aktuális játékbeállításoktól függően. Elrejti, amire nincs szükség, majd megjeleníti, amire van.
- **refreshTime():** Frissíti az időt kijelző címkék tartalmát.
- **refreshTimeButton():** Frissíti az időt megállító illetve újraindító gomb ikonját aszerint, hogy éppen méri-e a modell az időt.
- **emptyLayout():** Eltávolítja az összes elemet. Többnyire a frissítést megelőzően van rá szükség.



44. ábra: GameControllerView osztály

- **fillLayout():** Megjeleníti a megjelenítendő elemeket az aktuális játékbeállításoktól függően.
- **stopTimeClicked():** Megállítja, vagy újraindítja az idő mérését.
- **leaveGameClicked():** Elhagyja az aktuális játékot.
- **saveGameClicked():** Felugrik a mentések ablak, majd jóváhagyás és elnevezés esetén menti az aktuális játékállást.
- **setSettingsClicked():** Felugrik a beállítások ablak, majd jóváhagyás esetén frissíti a beállításokat.
- **resizeEvent(QResizeEvent*):** Az átméretezés esetén meghívott függvény. Gondoskodik róla, hogy minden elemének mérete az aktuális mérethez igazodjon.

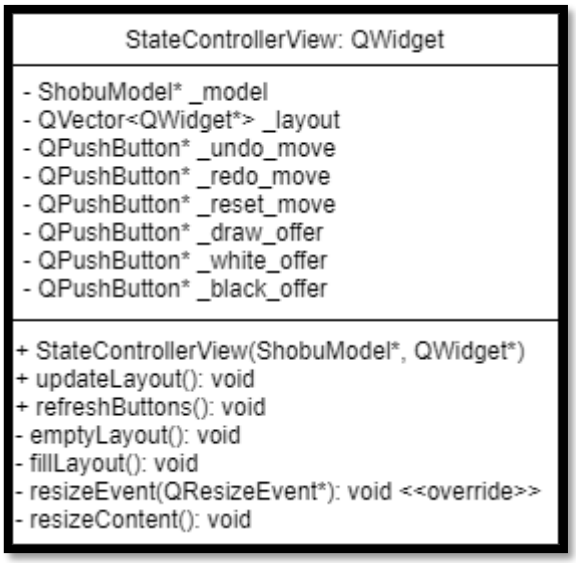
- **resizeContent():** Gondoskodik róla, hogy minden elemének mérete az aktuális mérethez igazodjon. A resizeEvent(QResizeEvent*) is ezt hívja meg.
- **leaveGame():** Szignál, ami jelzi, hogy a felhasználó el szeretné hagyni a játékot.

3.2.5.3. StateControllerView

A StateControllerView osztály tartalmazza a játékablak láblécében található gombokat. Innen lehet vezérelni azokat a funkciókat, amiknek hatása van a játékállásra.

Fontosabb függvények:

- **updateLayout():** Frissíti a elemeket az aktuális játékbeállításoktól függően. Elrejti, amire nincs szükség, majd megjeleníti, amire van.
- **refreshButtons():** Frissíti a játékállástól függően a gombok elérhetőségét.
- **emptyLayout():** Eltávolítja az összes elemet. Többnyire a frissítést megelőzően van rá szükség.
- **fillLayout():** Megjeleníti a megjelenítendő elemeket az aktuális játékbeállításoktól függően.

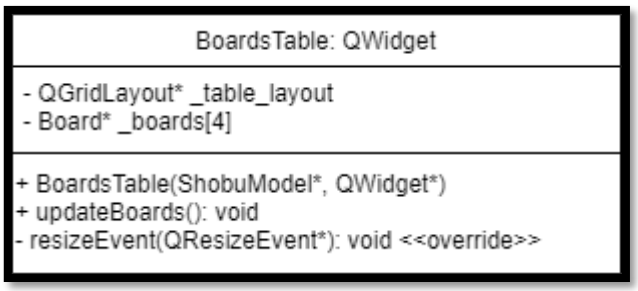


45. ábra: StateControllerView osztály

- **resizeEvent(QResizeEvent*):** Az átméretezés esetén meghívott függvény. Gondoskodik róla, hogy minden elemének mérete az aktuális mérethez igazodjon.
- **resizeContent():** Gondoskodik róla, hogy minden elemének mérete az aktuális mérethez igazodjon. A resizeEvent(QResizeEvent*) hívja meg.

3.2.5.4. BoardsTable

A BoardsTable tartalmazza a négy játéktáblát. Ez felel a táblák szignáljainak továbbításáért, a táblák frissítéséért és elhelyezésükért.



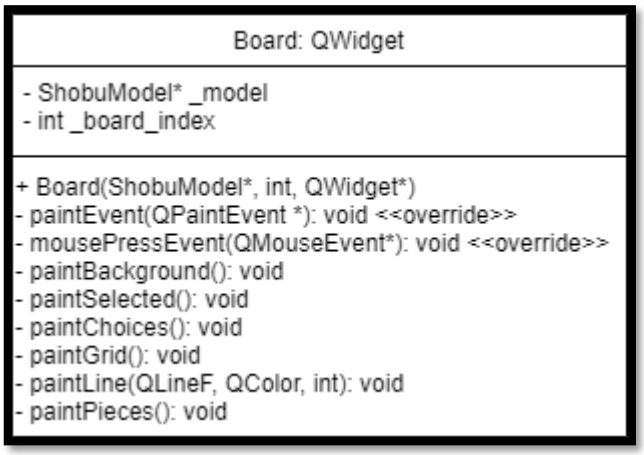
46. ábra: BoardsTable osztály

Fontosabb függvények:

- **updateBoards():** Frissíti az összes táblát az aktuális játékállás szerint.
- **resizeEvent(QResizeEvent*):** Az átméretezés esetén meghívott függvény. Gondoskodik róla, hogy minden elemének mérete az aktuális mérethez igazodjon.

3.2.5.5. Board

A Board osztály a játék egy tábláját jeleníti meg, és a rajta történő kattintásokat továbbítja a modell számára.



47. ábra: Board osztály

Fontosabb függvények:

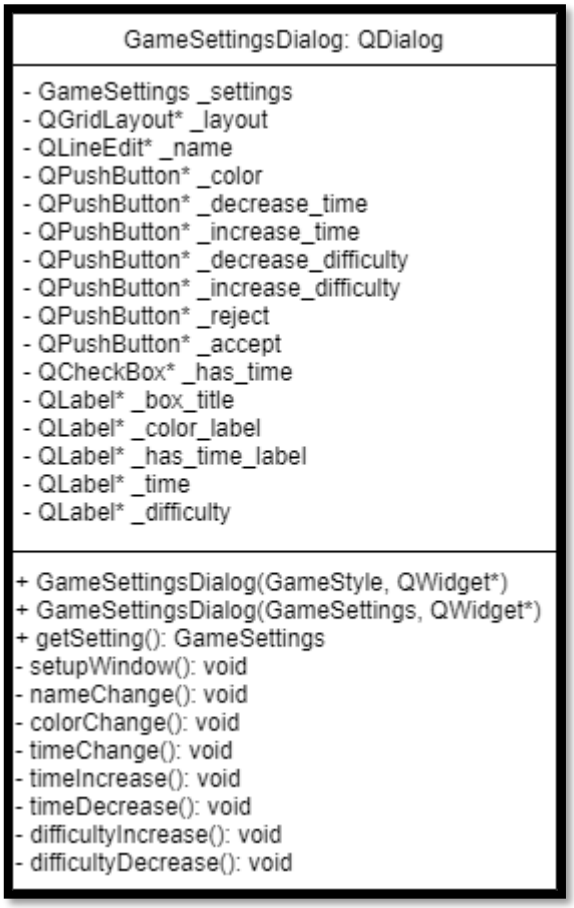
- **paintEvent(QPaintEvent*):** Felülírja a megszokott kirajzoló függvényt. Ennek köszönhetően tudja ábrázolni a táblát.
- **mousePressEvent(QMouseEvent*):** A felhasználó kattintása hatására továbbítja a kattintás koordinátáit a soron következő játékos osztálynak.
- **paintBackground():** Kiszínezi a tábla hátterét a tábla index alapján.
- **paintSelected():** Megjelöli a már folyamatban lévő lépésben kiválasztott mezőket.
- **paintChoices():** Megjelöli a szabályos lépés keretében kiválasztható mezőket.
- **paintGrid():** Megrajzolja a tábla mezőit elválasztó vonalakat.
- **paintLine(QLineF, QColor, int):** Megrajzol egy darab vonalat. A paintGrid() függvény használja.
- **paintPieces():** Megrajzolja a játékban lévő kavicsokat.

3.2.5.6. GameSettingsDialog

A felugró ablak, aminek segítségével a felhasználó megadhatja a játék beállításait. Sikeres beállítás esetén egy GameSettings struktúrát készít.

Fontosabb függvények:

- **getSetting():** Visszaadja a kész beállításokat. A sikeres beállítás után ezzel a függvénnyel kérjük el az eredményt.
- **setupWindow():** Elhelyezi a játékmódhoz szükséges elemeket az ablakon.
- **nameChange():** Megváltoztatja a visszatérítendő GameSettings struktúra név tulajdonságát.
- **colorChange():** Megváltoztatja a visszatérítendő GameSettings struktúra szín tulajdonságát.
- **timeChange():** Megváltoztatja, hogy a visszatérítendő GameSettings struktúra tartalmaz-e időlimitet.
- **timeIncrease():** Növeli a visszatérítendő GameSettings struktúra időlimit tulajdonságát.

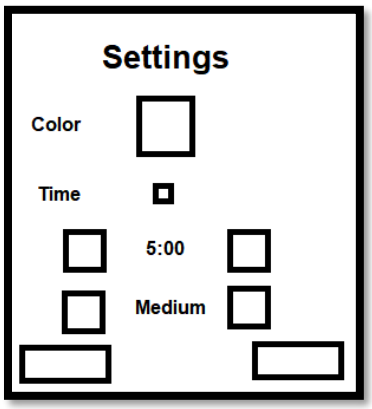


48. ábra: GameSettingsDialog osztály

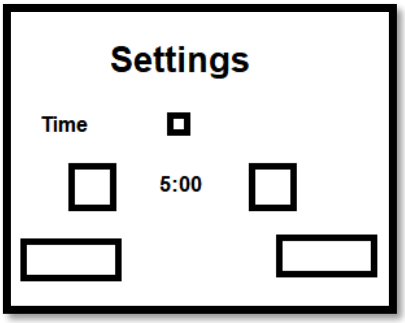
- **timeDecrease():** Csökkenti a visszatérítendő GameSettings struktúra időlimit tulajdonságát.
- **difficultyIncrease():** Növeli a visszatérítendő GameSettings struktúra nehézségi szintjét.
- **difficultyDecrease():** Csökkenti a visszatérítendő GameSettings struktúra nehézségi szintjét.

Megjelenítés terve:

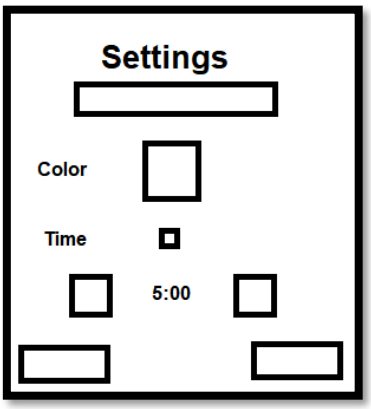
A megjelenés terve az 49., 50. és 51. ábrákon látható:



49. ábra: Egyszemélyes beállítások

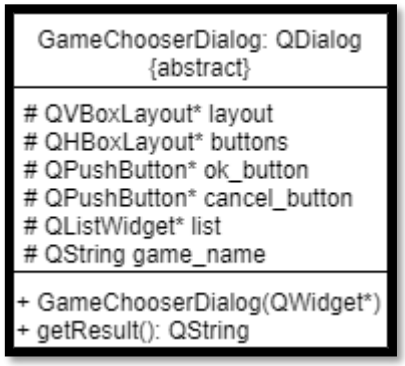


50. ábra: Kétszemélyes beállítások



51. ábra: Hálózati játék beállítások

3.2.5.7. GameChooserDialog



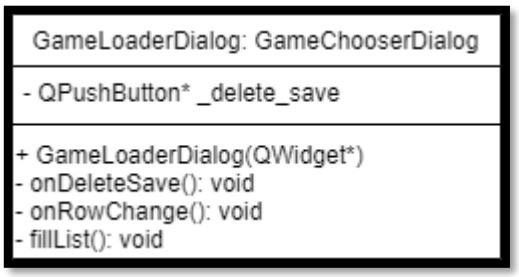
52. ábra: GameChooserDialog osztály

A GameChooserDialog az ősosztálya minden olyan felugró ablaknak, ami valamelyik játék kiválasztására szólíthatja fel a felhasználót. Sikeres választás esetén egy QString-et ad vissza, ami a kiválasztott játék nevét tartalmazza.

Fontosabb függvények:

- **getResult():** Visszaadja a kiválasztott játék nevét.

3.2.5.8. GameLoaderDialog



53. ábra: GameLoaderDialog osztály

A GameLoaderDialog osztály a GameChooserDialog azon leszármazottja, amely a korábban elmentett játékok közül ad lehetőséget a választásra, valamint megengedi ezeknek törlését is. Rendszerint a kiválasztott játékállás betöltésre kerül sikeres választás esetén.

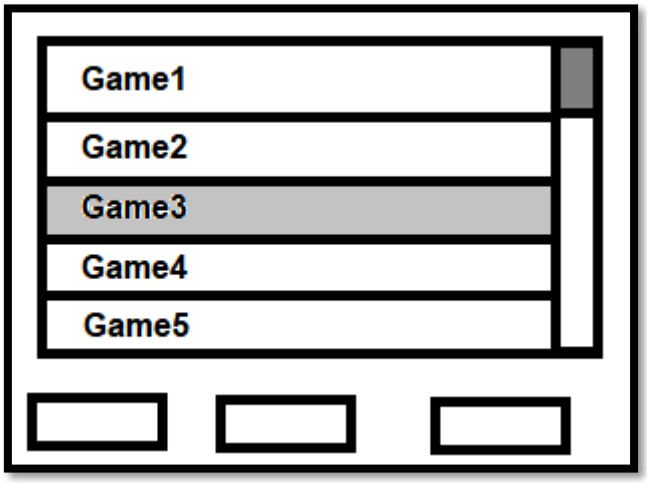
Fontosabb függvények:

- **onDeleteSave():** Megerősítést követően letörli a kiválasztott játékot. Ha nincs kiválasztott játék, semmit nem csinál.

- **onRowChange():** Megváltoztatja a visszatérési értéket úgy, hogy az új kiválasztott név kerüljön bele. Amennyiben a változás végén nincs kiválasztott játék, a visszatérési érték üres lesz.
- **fillList():** Lekérdezi az elérhető játékok listáját, és megjeleníti a választó dobozában.

Megjelenítés terve:

A megjelenés terve az 54. ábrán látható:



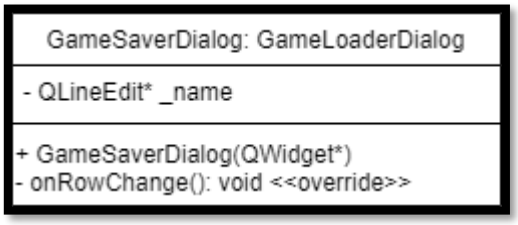
54. ábra: Játék kiválasztó ablak terve

3.2.5.9. GameSaverDialog

A GameSaverDialog osztály a GameLoaderDialog leszármazottja, amely képes egy szövegdobozon keresztül olyan nevet is választani, ami nincs benne az eredeti listában. Rendszerint az általa visszaadott névvel új mentés készül.

Fontosabb függvények:

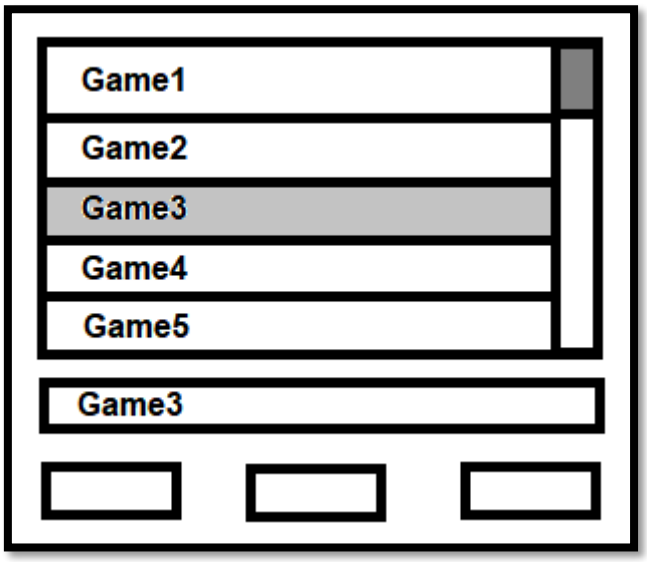
- **onRowChange():** Megváltoztatja a visszatérési értéket úgy, hogy az új kiválasztott név kerüljön bele. Amennyiben a változás végén nincs kiválasztott játék, a visszatérési érték üres lesz. Az aktuális visszatérési értéket megjeleníti a szövegbeviteli mezőben is.



55. ábra: GameSaverDialog osztály

Megjelenítés terve:

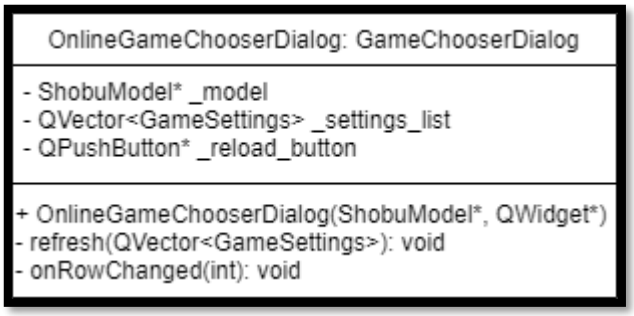
A megjelenés terve az 56. ábrán látható:



56. ábra: Játék elmentő ablak terve

3.2.5.10. OnlineGameChooserDialog

Az OnlineGameChooserDialog osztály a GameChooserDialog azon leszármazottja, amely az elérhető online játékok közül ad lehetőséget a választásra. Rendszerint a játék kiválasztása után a felhasználó váróterembe kerül, amíg a szerver vissza nem jelez.



57. ábra: OnlineGameChooserDialog osztály

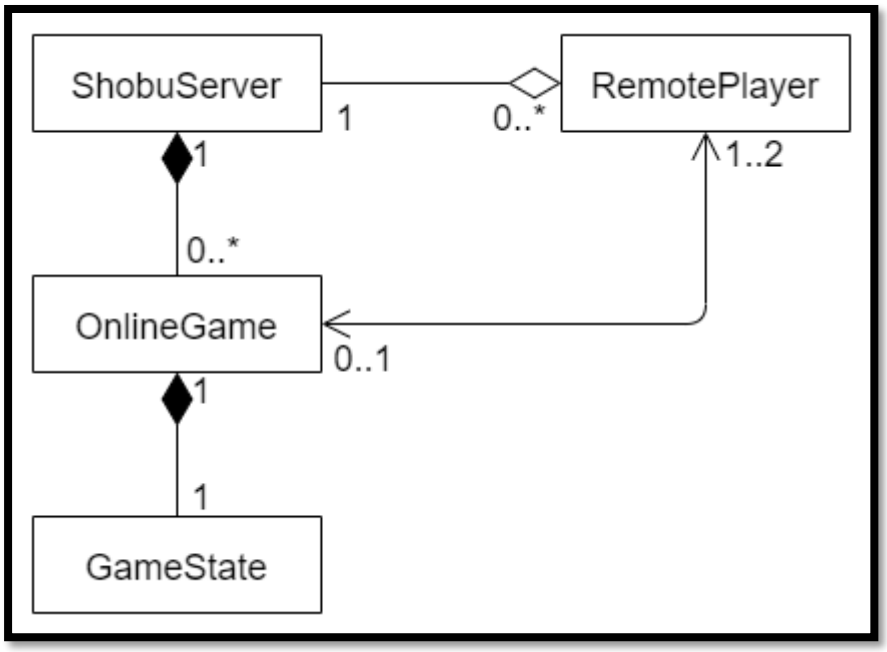
Fontosabb függvények:

- **refresh(QVector<GameSettings>):** Frissíti az elérhető játékok listáját. A modell gotOnlineList(QVector<GameSettings>) szignálja hatására fut le.
- **onRowChange():** Megváltoztatja a visszatérési értéket úgy, hogy az új kiválasztott név kerüljön bele. Frissíti az eltárolt GameSettings értéket is. Amennyiben a változás végén nincs kiválasztott játék, a visszatérési érték üres lesz.

3.2.6. Szerver

A szerver köti össze a hálózaton keresztül játszó felhasználókat. A feladatai közé többek között a játékok létrehozása, a játékosok összekapcsolása és a játék (szabályos) levezénylése tartozik.

A szerver osztályainak kapcsolatát a 58. ábrán lévő UML diagramon láthatjuk. A program méretére való tekintettel az olvashatóság érdekében az egyes osztályokat csak később tárgyaljuk részletesebben.



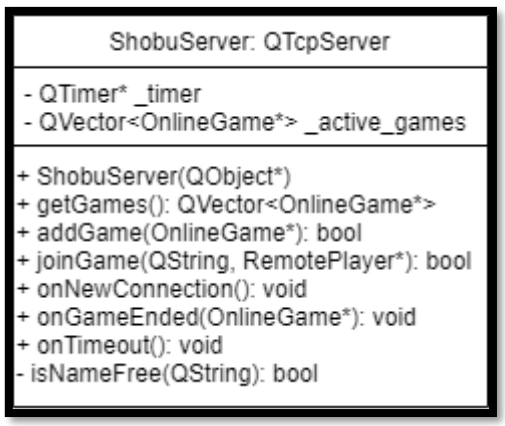
58. ábra: A szerver osztályainak kapcsolata

3.2.6.1. ShobuServer

A ShobuServer osztály fogja össze a szerverrel kapcsolatos osztályokat. Ez fogadja az új klienseket, és itt található a játékok listája is.

Fontosabb függvények:

- **getGames():** Visszaadja azoknak a játékoknak a listáját, amikhez pillanatnyilag lehet csatlakozni.
- **addGame(OnlineGame*):** Hozzáad egy új játékot a játékok listájához, amennyiben még nem foglalt a neve.
- **joinGame(QString, RemotePlayer*):** Hozzáad egy játékost az egyik játékhoz, amennyiben a játék még elérhető, és nem kezdődött el.



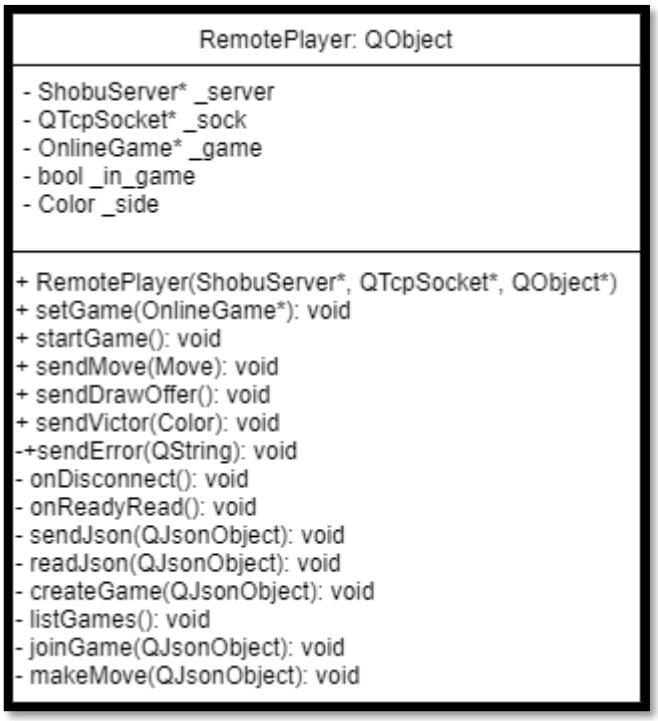
59. ábra: ShobuServer osztály

- **onNewConnection():** Elkészíti az új klienshez tartozó osztályt, amin keresztül a hálózati kommunikáció folyik.

- **onGameEnded(OnlineGame*):** Letörli a játékot, amely véget ért.
- **onTimeout():** Jelzi a játékoknak az idő múlását. Ez alapján mérik az időlimittel rendelkező játékok a hátralévő időt.
- **isNameFree(QString):** Visszaadja, hogy az adott név foglalt-e már az aktuális játéklistában.

3.2.6.2. RemotePlayer

A RemotePlayer osztály felel a klienssel való kommunikációért. Ez értelmezi a megérkező üzeneteket, és ez alakítja a kimenő üzeneteket is megfelelő formátumba.



60. ábra: RemotePlayer osztály

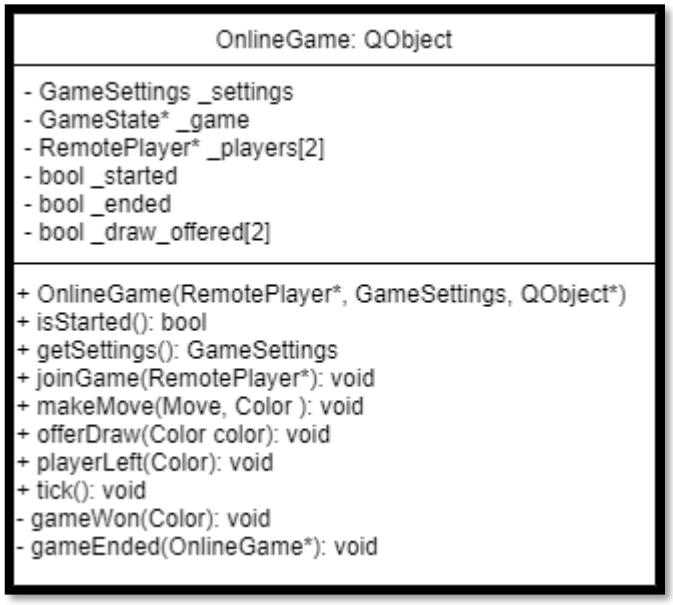
Fontosabb függvények:

- **setGame(OnlineGame*):** Hozzáköti a játékos osztályhoz a megfelelő játékot.
- **startGame():** Üzenetet küld a kliensnek a játék elindulásáról
- **sendMove(Move):** Értesíti a klienst az ellenfél egy lépéséről.
- **sendDrawOffer():** Értesíti a klienst az ellenfél döntetlen ajánlatáról.
- **sendVictor(Color):** Elküldi a kliensnek a játék győztesét.
- **sendError(QString):** Hibaüzenetet küld a kliensnek.
- **onDisconnect():** Lecsatlakozás esetén letörli a feleslegessé vált osztályokat.
- **onReadyRead():** Json formátumúvá alakítja a kliens üzenetét. Új üzenet érkezésének hatására fut le.
- **readJson(QJsonObject):** Kiolvassa a json-ból a kliens üzenetét, és az üzenet típusától függően kezeli azt.
- **sendJson(QJsonObject):** Elküldi a kliensnek szánt üzenetet.

- **createGame(QJsonObject):** Megpróbál létrehozni egy új játékot a json adatainak megfelelően.
- **listGames():** Elküldi az elérhető játékok listáját a kliensnek.
- **joinGame(QJsonObject):** Megpróbálja a klienst csatlakoztatni egy játékhoz.
- **makeMove(QJsonObject):** A kliens lépését közli az OnlineGame osztállyal.

3.2.6.3. OnlineGame

Az OnlineGame osztály tartalmazza a játék lebonyolításához szükséges részeket. Ez jelez az egyes játékosoknak, ha történik valami, és ez frissíti az aktuális játékállást, ha szükséges.



61. ábra: OnlineGame osztály

Fontosabb függvények:

- **joinGame(RemotePlayer*):** Hozzákapcsolja a játékhoz a második játékost, és elkezdi a játékot.
- **makeMove(Move, Color):** Alkalmazza a megadott lépést, és közvetíti a másik játékos felé.
- **offerDraw(Color):** Továbbítja a döntetlen ajánlatot a másik játékosnak. Amennyiben mindkét játékos döntetlent ajánlott, a játék döntetlennel véget ér.
- **playerLeft(Color):** Az egyik játékos idő előtti távozása esetén a másik játékosnak ítéli a győzelmet, amennyiben a játék már elkezdődött.
- **tick():** Időlimites játék esetén csökkenti a soron következő játékos hátralévő idejét. Ha az idő elfogy, a győzelmet a másik játékosnak ítéli.
- **gameWon(Color):** Továbbítja a játékosoknak a győztes színét.
- **gameEnded(OnlineGame*):** Szignál, amivel jelzi a ShobuServer osztálynak, hogy a játék törölhető.

3.2.7. Struktúrák és enumerátorok

Az alkalmazásszámos helyen saját adattípusok segítségével kommunikál az egyes osztályok között, valamint enumerátorokkal könnyíti a gyakran előforduló számértékek használatát.

Fontosabb enumerátorok:

- **Color**: Egy mezőnek a színét, valamint a soron következő játékos színét jelöli. Lehet fehér, fekete vagy üres. (A soron következő játékos nem lehet üres.)
- **GameStyle**: A játék típusát tárolja el. Ez lehet egyszemélyes, kétszemélyes vagy online játék.
- **Difficulty**: A gépi ellenfél által használt logika erejét határozza meg. A könnyű, a közepes vagy a nehéz értékeket veheti fel
- **NetworkEvents**: A kliens és a szerver közti kommunikációban ez jelöli meg az üzenet típusát. Mindig az elküldött json objektum „event” tulajdonságába kerülnek.

Fontosabb struktúrák:

- **GameSettings**: A játék beállításait tartalmazza. Ebben kerül eltárolásra a játék neve, típusa, a játékos színe, az időlimit és a játékosonkénti hátralévő idő.
- **Coordinate:** Három szám segítségével azonosítja a tábla egy mezőjét.
- **Move:** Egy lépéshez szükséges adatokat tárolja. Két koordinátát és egy (három számból álló) vektort tartalmaz.
- **ReverseData:** Egy lépés visszaállításához szükséges adatokat tárolja.
- **MoveState:** Egy Move struktúrát tartalmaz, valamint a lépés állapotáról információkat. Ennek segítségével kommunikál a játékos, a modell és a nézet.

3.2.8. Gépi logika

A számítógép három különböző erősségű ellenféllel rendelkezik, amik a következő logikák szerint választják ki a legjobb lépést:

3.2.8.1 Könnyű

A könnyű ellenfél semmiféle stratégiát nem alkalmaz. A lehetséges lépések közül véletlenszerűen választ ki egyet.

3.2.8.1. Közepes

A közepes ellenfél mohó algoritmust használ. A lehetséges lépések közül mindig azt választja ki, amelyiken az ellenfelének a legkevesebb bábuja van. Ha több állás ugyan azt a számot adja, azt választja, ahol a legkevesebb ellenfelet tartalmazó tábla a lehető legkevesebb ellenfelet tartalmazza, azaz mindig az ellenfél leggyengébb tábláján próbál nyerni. Ha meg tudja nyerni egy lépéssel nyerni a játékot, azt ennek köszönhetően meg is teszi, mivel ebben az esetben az ellenfélnek 0 kavicsa lesz a leggyengébb tábláján. Ha több egyenlő erősségű lépés áll rendelkezésére, ezek közül véletlenszerűen választ. A logika gyengesége viszont, hogy egyáltalán nem vigyáz a saját kavicsaira, és nem próbálja meg aktívan sarokba szorítani az ellenfelét a leggyengébb táblán.

3.2.8.1. Nehéz

A nehéz ellenfél szintén mohó algoritmust használ, de ez az egyes mezőkhöz pontértékeket rendel. Ezen kívül a közepes ellenfélhez nagyon hasonlóan játszik, mindig az ellenfele kavicsainak leütésére törekszik. Erőssége viszont, hogy nem tesz olyan lépést, amire válaszul a felhasználó megnyerheti a játékot, amennyiben az ilyen lépések elkerülhetőek, és mindenképpen meglépi azokat a lépéseket, amik azonnal a győzelméhez vezetnek. Gyengesége, hogy csak egy lépéssel tud előre tekinteni, így, ha az egyik lépése potenciálisan két lépésen belül sarokba szorítja az utolsó kavicsát az egyik táblán, azt nem fogja észrevenni. Személyes tapasztalat alapján a korai agresszív támadás hatékony ellene. Ha túl hosszúra nyúlik a játék, egyre nehezebb a biztos vereség irányába terelni.

3.3. Megvalósítás

3.3.1. Qt 5

A program a Qt 5 keretrendszer felhasználásával készült. Ez lehetővé tette, hogy az erőforrásigényes műveleteknél, mint például a gépi játékos döntéshozási műveleténél, kihasználjuk a C++ hatékonyságát, de emellett magával hozta egy magasabb szintű programozási nyelv előnyeit is:

- A QObject fák tesznek róla, hogy minden objektum megsemmisüljön, amikor a szülőobjektuma megsemmisül, így a memóriaszivárgást jelentősen könnyebb elkerülni.

- A signal-slot rendszer megkönnyíti az eseményvezérelt működést, és lehetővé teszi az osztályok kommunikációját úgy, hogy nem kell feltétlenül mindkét osztálynak tartalmaznia egy, a mások osztályra mutató referenciát.
- A megjelenítést a QWidget osztály leszármazottjaival, a felugró ablakokat pedig a QDialog leszármazottjaival lehet megvalósítani. Ezek személyre szabását megkönnyítik a qss stíluslapjai.
- A QTcpServer és a QTcpSocket osztályok lehetővé teszik a szerver és a kliens közti kommunikációt.
- A fájlkezeléshez előre definiált osztályokat tartalmaz.
- A Qt Installer Framework lehetővé teszi a telepítő egyszerű előállítását.
- A teszteléshez kész keretet ad. [5]

3.3.2. Qt Creator

A program megírása a Qt Creator fejlesztői környezetben készült. Továbbfejlesztése is ebben ajánlott, a 4.14.2-es verzió, vagy annál későbbi használatával. Ekkor a Qt Creator segítségével megnyithatjuk a Shobu/Shobu.pro, ShobuServer/ShobuServer.pro, illetve a ShobuTest/ShobuTest.pro fájlokat attól függően, hogy melyik projektet szeretnénk kiegészíteni.

3.3.3. Párhuzamosság

A Qt rendszerén belül az idő mérésére használt QTimer párhuzamosan működik, de az egy szálon futó függvények (a mi esetünkben mindegyik) soha nem futhatnak egyszerre. Ha az egyik művelet éppen folyamatban van, a QTimer timeout() szignálja által meghívott függvény kivárja, amíg az befejeződik, és csak utána fut le. Így az idő mérésének leállításával az időzítő minden függvénye könnyen megszakítható anélkül, hogy a különböző műveletek összeakadnának.

3.3.4. Szignálok és kezelésük

A program számos helyen szignálok segítségével kommunikál. Néha a szignálok feldolgozása egyértelműen kiolvasható a kódból, mint például a nézet a saját gombjának kattintás eseményét

figyeli, de előfordul, hogy egy szignál kezelését egy másik osztályban kell keresni. Az ilyen, kevésbé kézenfekvő eseteket a következő táblázat segítségével látható át könnyebben:

Sender	Signal	Receiver	Slot
_game_header	leaveGame	main_window	setMainMenuLayout
_model	stepGame	main_window	onStepGame
_model	boardChange	main_window	onBoardChange
_model	gameOver	main_window	gameOver
_model	timeIsPassing	main_window	tick
_model	gotServerMessage	main_window	onServerMessage
_model	gameOver	main_window	gameOver
_model	gotOnlineList	OnlineGameChooserDialog (dialog)	refresh
_client	gamesArrived	_model	gotOnlineList
_client	serverMessage	_model	gotServerMessage
_client	gameStarted	_model	onGameStarted
_client	gamesArrived	_model	gotOnlineList
_players[2]	moveMade	_model	makeMove
_players[2]	moveProgress	_model	boardChange

3.4. Tesztelés

Ebben a részben esik szó a tesztelésről és program korábbi hiányosságai miatti változtatásokról. A modell osztályainak helyességét egységtesztekkel igazolhatjuk, míg a megjelenést végfelhasználói tesztesetekkel.

3.4.1. Egység tesztek

Az egységtesztek a modellhez tartozó osztályok publikus függvényei helyességének ellenőrzésére készültek. Mivel az egységtesztek a Qt keretrendszer segítségével (a QtTest

osztály makróival) készültek futtatásuk a Qt Creator-on keresztül javasolt, a ShobuTest/ShobuTest.pro fájl megnyitásával. A tesztelés külön projectként készült, így a program változtatása esetén frissíteni kell az érintett osztályokat a tesztek újra futtatása előtt. A tesztek neve gyakran megegyezik a tesztelt függvényével, csak a teszt snake_case-t használ az elnevezéskor, míg az eredeti függvény camelCase-t.

3.4.1.1. GameState

- **initialize_blank():** Ellenőrzi, hogy inicializálást abban az esetben, amikor még nincs előző játékállapot.
- **initialize_occupied():** Ellenőrzi, hogy inicializálást abban az esetben, amikor már van előző játékállapot.
- **is_home_board():** Ellenőrzi az isHomeBoard függvény működését érvényes és érvénytelen bemenetekkel.
- **set_state_null():** Ellenőrzi, hogy hibát dob-e a setState függvény, ha „nullptr” értéket kap másik GameState helyett.
- **set_state():** Ellenőrzi a setState függvény működését érvényes bemenet esetén.
- **set_field():** Ellenőrzi, hogy valóban megváltoznak-e a mező értékei, ha megváltoztatjuk őket a setField függvénnyel.
- **set_turn():** Ellenőrzi, hogy valóban megváltozik-e a soron következő játékos, ha megváltoztatjuk a setTurn függvénnyel.
- **make_move():** Ellenőrzi, hogy a szabályos lépések valóban megváltoztatják-e a játékállást, a szabálytalanok pedig változatlanul hagyják a makeMove függvény használata esetén.
- **end_turn():** Ellenőrzi, hogy a következő játékos valóban sorra kerül-e az előző kör végén.
- **get_victor():** Ellenőrzi, hogy a játék felismeri-e, ha az egyik játékos megnyerte a játékot.
- **is_legal_move():** Kipróbálja a lehetséges hibás lépéseket, majd miután azok szabálytalannak minősülnek, egy helyeset is, ami elfogadásra kerül.
- **is_legal_passive():** Kipróbálja a lehetséges hibás passzív kavicsokat, majd miután azok szabálytalannak minősülnek, egy helyeset is, ami elfogadásra kerül.

- **is_legal_vector():** Kipróbálja a lehetséges hibás vektorokat, majd miután azok szabálytalannak minősülnek, egy helyeset is, ami elfogadásra kerül.
- **is_legal_agressive():** Kipróbálja a lehetséges hibás agresszív kavicsokat, majd miután azok szabálytalannak minősülnek, egy helyeset is, ami elfogadásra kerül.
- **get_moves():** Leellenőrzi, hogy a getMoves függvény valóban szabályos lépéseket ad-e vissza
- **get_passive_pieces():** Leellenőrzi, hogy a getPassivePieces függvény valóban szabályos passzív kavicsok koordinátáit adja-e vissza.
- **get_destinations():** Leellenőrzi, hogy a getDestinations függvény valóban szabályos passzív célállomások koordinátáit adja-e vissza a megadott passzív kavicshoz.
- **get_agressive_pieces():** Leellenőrzi, hogy a getAgressivePieces függvény valóban szabályos agresszív kavicsok koordinátáit adja-e vissza a megadott passzív kavicshoz és vektorhoz.
- **get_applied():** Leellenőrzi, hogy a getApplied függvény valóban ugyan azt az állapotot adja-e vissza, mint amit létrehozásánál használt a szabályos lépéssel kapnánk.
- **apply_move():** Ellenőrzi, hogy a szabályos lépések valóban megváltoztatják-e a játékállást, a szabálytalanok pedig hibát dobnak a applyMove függvény használata esetén.
- **reverse_move():** Ellenőrzi, hogy a reverseMove függvény valóban a lépés előtti állapotot adja-e vissza.

3.4.1.2. MachineLogic

- **random_legal():** Ellenőrzi a RandomLogic osztály makeMove függvénye által visszaadott lépés szabályosságát.
- **greedy_legal():** Ellenőrzi a GreedyLogic osztály makeMove függvénye által visszaadott lépés szabályosságát.
- **hard_legal():** Ellenőrzi a HardLogic osztály makeMove függvénye által visszaadott lépés szabályosságát.
- **greedy_beats_random():** Ellenőrzi, hogy a közepes nehézségű gépi ellenfél megveri-e a könnyűt. A random tényező miatt ez időnként elbukhat, de aggodalomra nem ad okot.

- **hard_beats_random():** Ellenőrzi, hogy a nehéz nehézségű gépi ellenfél megveri-e a könnyűt. A random tényező miatt ez időnként elbukhat, de aggodalomra nem ad okot.
- **hard_beats_greedy():** Ellenőrzi, hogy a nehéz nehézségű gépi ellenfél megveri-e a közepeset. A random tényező miatt ez időnként elbukhat, de aggodalomra nem ad okot.
- **machine_logic_error():** Ellenőrzi, hogy a MachineLogic leszármazottjai hibát dobnak-e, ha olyan esetben kérnek tőlük lépést, amikor nem létezik szabályos lépés.

3.4.1.3. ShobuPlayer

- **organic_moves():** Ellenőrzi, hogy az OrganicPlayer osztály lépései befolyásolják-e a lépést eltároló MoveState struktúrát.
- **machine_moves():** Ellenőrzi, hogy a MachineMove osztály szabályos lépéseket tesz-e a MoveState struktúrába.

3.4.1.1. ShobuModel

- **set_settings():** Leellenőrzi, hogy helyesen változik-e meg a ShobuModel osztály GameSettings struktúrája a setSetting függvény használata esetén.
- **change_time():** Leellenőrzi, hogy valóban leáll, illetve újraindul-e az idő mérése a changeTime függvény meghívásának hatására.
- **new_game():** Megvizsgálja, hogy alaphelyzetbe áll-e a játék új játék kezdése esetén.
- **model_move():** Megvizsgálja, hogy a modell valóban lép-e, amikor a ShobuPlayer osztálya jelzi, hogy a lépése készen áll.
- **reset_move():** Ellenőrzi, hogy a resetMove függvény valóban eltörli-e a folyamatban lévő lépést.
- **change_settings():** Leellenőrzi, hogy helyesen változik-e meg a ShobuModel osztály GameSettings struktúrája a changeSettings függvény használata esetén.
- **undo_step():** Ellenőrzi, hogy a lépés visszavonásával valóban a korábbi játékállást kapjuk-e vissza.
- **redo_step():** Ellenőrzi, hogy a visszavont lépés helyreállításával valóban a korábbi játékállást kapjuk-e vissza.

- **has_undo():** Ellenőrzi, hogy a hasUndo függvény valóban észleli-e, ha van visszaállítható játékállás.
- **has_redo():** Ellenőrzi, hogy a hasRedo függvény valóban észleli-e, ha van visszaállítható, korábban visszavont játékállás.

3.4.2. Végfelhasználói tesztesetek

Be- és kilépés:

	Teszteset	Elvárt hatás
1	Alkalmazás indítása	Megjelenik a főmenü.
2	Kilépés	Az alkalmazás leáll.

Főmenü:

	Teszteset	Elvárt hatás
3a	Új játék indítása	Megjelenik a beállítások ablak az alapbeállításokkal. (Világos színű játékos, közepes ellenfél, időlimit nélkül)
3b	Új játék beállítása (szín)	A beállítások ablakon megváltozik a játékos színe
3c	Új játék beállítása (időlimit)	A beállítások ablakon elérhetővé vagy elérhetetlenné válik az időlimit
3d	Új játék beállítása (időlimit állítás)	Ha aktív időlimit növelő vagy csökkentő gombot nyomunk le, az időlimit és a gombok elérhetősége frissül
3e	Új játék beállítása (nehézség)	Ha aktív nehézség növelő vagy csökkentő gombot nyomunk le, az nehézség és a gombok elérhetősége frissül
3f	Új játék elvetése	A játék nem indul el
3g	Új játék indítása	A játék elindul a kiválasztott beállításokkal
4a	Játék betöltése	Megjelenik a játék betöltő ablak. Az első elem ki van választva, amennyiben létezik
4b	Betölthető játék törlése	Megerősítés után a kiválasztott játék törlésre kerül.
4c	Betöltés elvetése	Nem indul el játék
4d	Betölthető játék indítása	A kiválasztott játékállás betöltődik.
5	Többjátékos menüre lépés	A többjátékos menü megjelenik

Többjátékos menü:

	Teszteset	Elvárt hatás
6a	Helyi kétszemélyes játék indítása	Megjelenik a beállítások ablak az alapbeállításokkal. (Világos színű játékos, időlimit nélkül)
6b	Új játék beállítása (időlimit)	A beállítások ablakon elérhetővé vagy elérhetetlenné válik az időlimit
6c	Új játék beállítása (időlimit állítás)	Ha aktív időlimit növelő vagy csökkentő gombot nyomunk le, az időlimit és a gombok elérhetősége frissül
6d	Új játék elvetése	A játék nem indul el
6e	Új játék indítása	A játék elindul a kiválasztott beállításokkal
7a	Új, online játék létrehozása	Szerverhez csatlakozó ablak felugrik
7b	Csatlakozás elvetése	Nem csatlakozik a szerverhez
7c	Csatlakozás a szerverhez	Megpróbál csatlakozni a megadott szerverhez. Siker esetén felugrik a beállítások ablak az alapbeállításokkal („ShobuGame” név, világos szín, időlimit nincs), különben figyelmeztető üzenet megjelenik
7d	Új játék beállítása (név)	A név frissül. Legfeljebb 15 karakter hosszú lehet
7e	Új játék beállítása (szín)	A beállítások ablakon megváltozik a játékos színe
7f	Új játék beállítása (időlimit)	A beállítások ablakon elérhetővé vagy elérhetetlenné válik az időlimit
7g	Új játék beállítása (időlimit állítás)	Ha aktív időlimit növelő vagy csökkentő gombot nyomunk le, az időlimit és a gombok elérhetősége frissül
7h	Új játék elvetése	A játék nem indul el
7i	Új játék indítása	A váróterem megjelenik. Amennyiben a név már foglalt, figyelmeztető üzenet megjelenik, majd főmenü megjelenik.
8a	Csatlakozás online játékhoz	Szerverhez csatlakozó ablak felugrik
8b	Csatlakozás elvetése	Nem csatlakozik a szerverhez
8c	Csatlakozás a szerverhez	Megpróbál csatlakozni a megadott szerverhez. Siker esetén felugrik az online játék kiválasztó ablak, különben figyelmeztető üzenet megjelenik. Első elem ki van választva, ha létezik, figyelmeztető üzenet megjelenik, ha nem létezik.
8d	Elérhető játékok frissítés	Az új játékok megjelennek. Első elem ki van választva, ha létezik, figyelmeztető üzenet megjelenik, ha nem létezik.
8e	Csatlakozás elvetése	Nem csatlakozik a játékhoz

8f	Csatlakozás a játékhoz	A váróterem megjelenik. Amennyiben a játék már elérhetetlen, figyelmeztető üzenet megjelenik, majd a többjátékos menü megjelenik
9	Főmenüre lépés	A főmenü megjelenik

Váróterem:

	Teszteset	Elvárt hatás
10	Váróterem elhagyása	Megerősítés után főmenü megjelenik
11	Új játék megkezdése	A szerver jelzése után a játék elkezdődik

Játékablak:

	Teszteset	Elvárt hatás
12	Játék kezdete	Játékablak megjelenik. Játékmódtól függően megjelennek a megfelelő kijelzők. Helyi játék esetén ezek a beállítások gomb, mentés gomb, idő megállító gomb (időlimit esetén) és a lépés visszavonó gombok. Online játék esetén döntetlen ajánló gomb és hozzá tartozó kijelzők. Minden esetben megjelenik a hátralévő idő (ha van időlimit), a játék elhagyó gomb, a játéktáblák és a lépés megszakító gomb.
13	Idő megállítása, újraindítása	Az idő mérése megáll, illetve újraindul. A gomb ikonja frissül
14a	Játék elmentése	A játék elmentő ablak felugrik, első elem ki van választva, és a neve a szövegbeviteli mezőben van
14b	Mentett játék törlése	Megerősítés után a mentett játék törlésre kerül, a játékok listája frissül, és az első elem kiválasztásra kerül
14c	Mentés elvetése	Nem kerül elmentésre a játékállás
14d	Mentés elnevezése	A név megváltozik a szövegbeviteli mezőben
14e	Játékállás mentése	A szövegbeviteli mezőben lévő néven elmentésre kerül az aktuális játékállás. Siker esetén megerősítő üzenet jelenik meg.
15a	Beállítások módosítása	Felugrik a beállítások ablak az aktuális beállításokkal
15b	Beállítások megváltoztatása	A játékmódnak megfelelően megváltoznak az ablakon lévő beállítások
15c	Módosítások elvetése	Nem változnak meg a beállítások
15d	Módosítások mentése	A játék beállításai megváltoznak. A játékablak kinézete ennek megfelelően frissül

16	Játék elhagyása	Megerősítés után a főmenü megjelenik
17	Szabályos lépés	A kattintás helye megjelölésre kerül, a kiválasztható mezők szintén megjelölődnek. Lépés vége esetén a tábla frissül, és a következő játékos köre következik. Függőben lévő döntetlen ajánlat esetén az eltörlődik. Gombok frissülnek.
18	Szabálytalan lépés	A kattintás hatására semmi nem történik
19	Gépi lépés	Pár másodperces késleltetés után a gépi játékos lép
20	Lépés visszavonása	Korábbi játékállás visszaállításra kerül, a gombok elérhetősége frissül
21	Visszavonás visszavonása	A korábban visszavont játékállás helyreáll, a gombok elérhetősége frissül
22	Lépés megszakítása	A folyamatban lévő lépés eltörlődik, a lépést elölről lehet kezdeni.
23	Döntetlen ajánlat érkezése	A megfelelő színű döntetlen ajánlat kijelző megjelenik
24	Döntetlen felajánlása	A megfelelő színű döntetlen ajánlat kijelző megjelenik, a gomb elérhetősége frissül
25a	Helyi játék vége	Felugró ablak kihirdeti a győztest
25b	Játék megismétlése	Elindul a játék ugyan azokkal a beállításokkal
25c	Visszatérés a főmenühöz	Főmenü megjelenik
25d	Online játék vége	Felugró ablak kihirdeti a győztest, majd a többjátékos menü megjelenik

3.4.3. Szükséges módosítások

Az eredeti terv szerint a legerősebb gépi logika minimax algoritmust használt volna a legjobb lépés megtalálására. A lehetséges lépések száma azonban ezt nem tette lehetővé. A kezdeti pozícióban a gépnek legrosszabb esetben 174 lépési lehetőséget kell megvizsgálnia, így, ha két szintes mélységben akarnánk megvizsgálni az általa generált játékfát, az akár 30276, ha három szintes mélységben, akkor 5268024 lehetséges játékállás vizsgálatát követelné meg. A fejlesztéshez használt számítógépen hamar túllépte a pár másodpercet, amit a nem-funkcionális követelmények között kikötöttünk, és a lépés nem is volt látványosan erősebb a mohó stratégiát követő logika lépésénél.

Az alfa-béta algoritmus valamennyit javított az előző állapoton, de még ott is 21000 körül mozgott a megvizsgálandó állások száma két szintes mélységben, az ereje pedig megegyezett a minimax algoritmuséval.

Mivel az egyértelműen hátrányos lépések kiszűrésére nincs mód, a játékfa többszintes kiértékelésén alapuló algoritmusok elvetésre kerültek. Helyét egy kifinomultabb mohó algoritmus vette át, ami az egy lépésen túli előretekintésében csak a vereséget ismeri fel.

Az alfa-béta algoritmust használó gépi logika osztálya a ForwardThinkerLogic osztályban található. A program többi részére nincs kihatással.

5. Fejlesztési lehetőségek

Habár a program megjelenítése kellően kontrasztos, és a program vezérlésére tökéletesen alkalmas, a játék népszerűségének várható növekedése után egy szebb, kevésbé minimalista megjelenés sokat emelne a játék hangulatán.

Ezen felül a legerősebb gépi logikánál lehetne egy még erősebbet készíteni, ha betanítanánk egy mély neuronhálót a legjobb lépés megtalálására egy adott játékállás alapján. Siker esetén akár egy közel legyőzhetetlen gépi ellenfél is elkészülhetne.

Ezen felül a szervert tovább lehetne fejleszteni úgy, hogy valódi tömegek kiszolgálására legyen képes, akár többszálúvá lehetne tenni, és ki lehetne bővíteni a hibakezelő funkcióit.

6. Összefoglaló

Ahogyan a bevezető megírásakor reméltem, az intelligens ellenfél elkészítése valóban egyedi kihívásnak bizonyult. A sakk esetén használt algoritmusok, habár jó kiindulási pontok voltak, a lépések nagy száma nem engedte meg, hogy az ott népszerű minimax algoritmust, és annak továbbfejlesztett változatait használjam.

Ennek ellenére a mohó algoritmusok meglepően erős ellenfelet voltak képesek előállítani. Azt tapasztaltam, hogy az agresszív játék általában hatékonyabb volt, mivel a kavicsok védelmére nem sok módszer áll rendelkezésünkre

Összességében a program így eleget az elvárásoknak, és habár nyerőstratégiát (vagy nemvesztő stratégiát) nem találtam, az elkészült algoritmus kihívást jelenthet akár tapasztaltabb játékosoknak is.

7. Forrásjegyzék

[1] „SHOBU,” tarsasjatekok.com, [Online].
Elérhető: https://tarsasjatekok.com/tarsasjatek/shobu-2019
[Hozzáférés dátuma: 28. 04. 2021.].

[2] „SHŌBU Game Rules,” Ultra BoardGames, [Online].
Elérhető: https://www.ultraboardgames.com/shobu/game-rules.php
[Hozzáférés dátuma: 29. 04. 2021.].

[3] „Qt Documantation Archives,” The Qt Company, [Online].
Elérhető: https://doc.qt.io/archives/qt-5.11-devicecreation/qtee-requirements-windows.html
[Hozzáférés dátuma: 30. 04. 2021.].

[4] „IconArchive,” IconArchive.com, [Online].
Elérhető: https://iconarchive.com/
[Hozzáférés dátuma: 19. 04. 2021.].

[5] „Qt Documantation,” The Qt Company, [Online].
Elérhető: https://doc.qt.io/qt-5/
[Hozzáférés dátuma: 25. 04. 2021.].