Házi feladat

Programozás alapjai 3.

Zelch Csaba

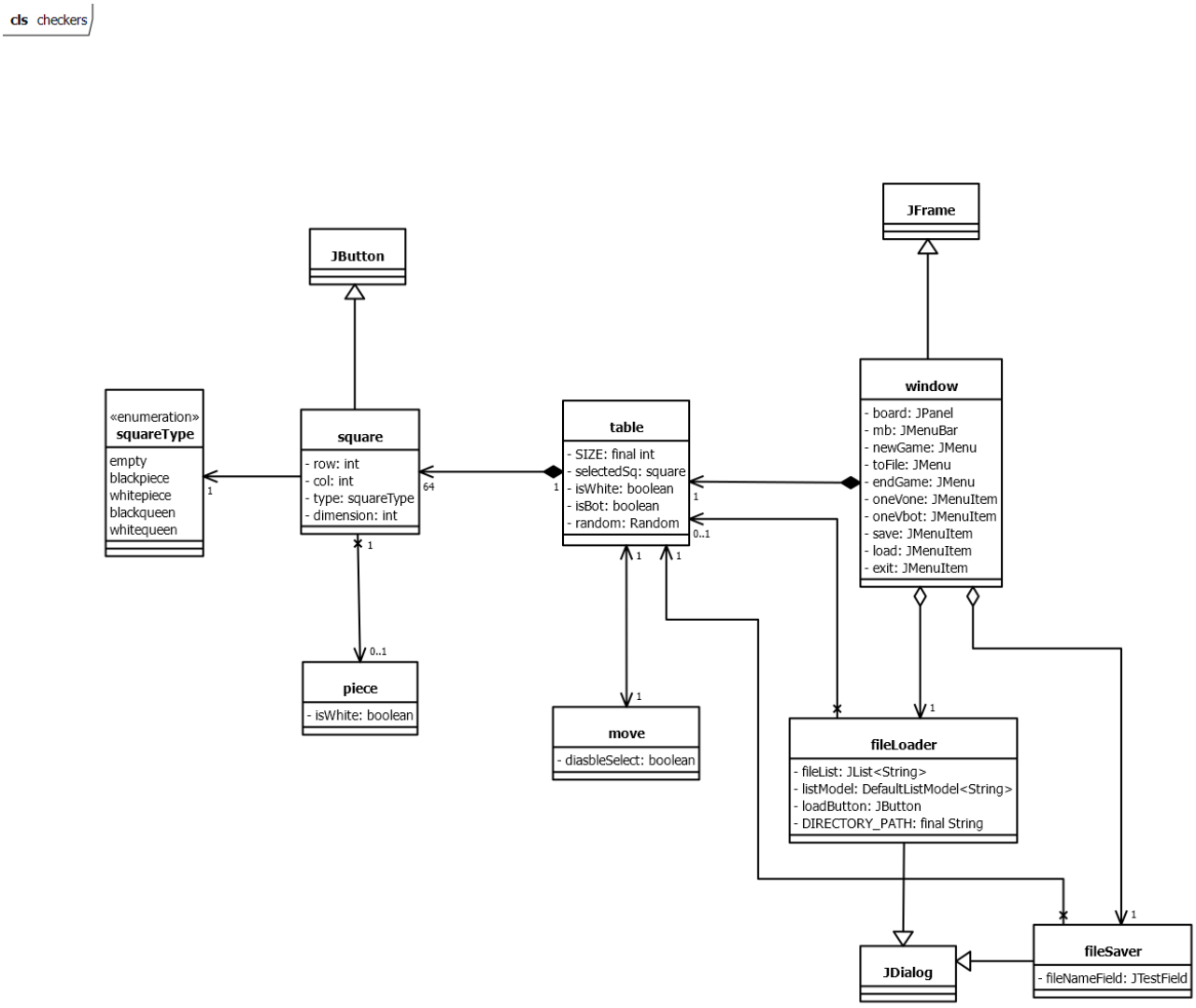
LK0617

Tartalom

1. Feladat

A feladat során dámajátékot készítettem, melyet a Java Swing segítségével grafikusan valósítottam meg. A játékot egy 8x8-as sakktáblán játszhatjuk, melynek két lehetséges módja van: játszhat egymás ellen két személy, vagy játszhat egy játékos akár egy úgynevezett robotjátékos ellen, a cél mindkét esetben a másik játékos lépésképtelenné tétele, követve a dámajáték pontos szabályait, amelyet a házi feladat sepcifikációjában tüntettem fel.

2. Osztálydiagram



//az osztálydiagramnál nem tüntettem fel a metódusokat, kicsit átláthatatlanná válna a diagram és lennebb amúgy is elmagyarázom az összeset.

3. A programban megvalósított osztályok és metódusaik

A program összesen kilenc osztályból áll:

- square
- piece
- table
- squareType
- move
- window
- fileSaver
- fileLoader
- Main

3.1. A square osztály

A square osztály a tábla egy mezőjét valósítja meg, attribútumai tartalmazzák a táblán elfoglalt sor és oszlop számát, a mező típusát, a mezőn elhelyezkedő bábu típusát, illetve a mező nagyságát. A square osztály a JButton leszármazottja, mivel a tábla minden mezője egy gomb lesz, amelyekre kattintva bábukat lehet kijelölni, és lépni lehet velük.

A square osztály konstruktora megkapja azt a sor és oszlopszámot, amely sorban és oszlopban elhelyezkedik, majd beállítja a kirajzolásához szükséges információkat, és létrehoz egy üres típusú mezőt, azaz nincs rajta bábu. A square osztály a következő metódusokkal rendelkezik:

public squareType getType()

- Ez a metódus visszaadja a mező típusát, amelyet az osztály a type attribútumában tárol

public piece getPiece()

- Ez a metódus egy piece típusú objektumot ad vissza, azaz visszaadja a mezőn található bábut, null-t ad vissza, ha a mező üres

public int getRow()

public int getCol()

- Visszaadják a táblán elfoglalt sor, illetve oszlop számát

public void setSquare(squareType sqType)

- Ez a metódus a megadott mező típusa alapján beállítja a mezőt

A mező típusa, vagyis a squareType egy enum objektum, amely a következő értékeket veheti fel: empty, blackpiece, whitepiece, whitequeen, blackqueen. Azaz ahogy a nevük is sugallja az empty az üres mező, a blackpiece a sima fekete bábut tartalmazó mező, a whitepiece a sima fehér bábut

tartalmazó mező, a whitequeen a fehér dámát tartalmazó mező, a blackqueen a fekete dámát tartalmazó mező típusa lesz.

public void setempty()

- Ez a metódus a mező típusát üresre állítja, a bábu értékét null-ra állítja, és a mező ikonját is kitörli, ha volt

public void setblackpiece()

- Ez a metódus a mező típusát blackpiece-re állítja, a mező p, azaz a bábu attribútumát egy létrehozott fekete bábuval teszi egyenlővé, illetve a mezőnek egy fekete bábu ikont állít be, amelyet a játék könyvtárában talál meg a black.png néven

public void setwhitepiece()

- Ez a metódus a mező típusát whitepiece-re állítja, a mező p, azaz a bábu attribútumát egy létrehozott fehér bábuval teszi egyenlővé, illetve a mezőnek egy fehér bábu ikont állít be, amelyet a játék könyvtárában talál meg a white.png néven

public void setblackqueen()

- Ez a metódus a mező típusát blackqueen-re állítja, a mező p, azaz a bábu attribútumát egy létrehozott fekete dámával teszi egyenlővé, illetve a mezőnek egy fekete dáma ikont állít be, amelyet a játék könyvtárában talál meg a blackqueen.png néven

public void setwhitequeen()

- Ez a metódus a mező típusát whitequeen-re állítja, a mező p, azaz a bábu attribútumát egy létrehozott fehér dámával teszi egyenlővé, illetve a mezőnek egy fehér dáma ikont állít be, amelyet a játék könyvtárában talál meg a whitequeen.png néven

private ImageIcon scale(String s)

- Ez a metódus kicsinyíti le az ikon méretét, hogy a mezőre pontosan el lehessen helyezni

3.2. A piece osztály

Ez az osztály a mezőn elhelyezkedő bábut valósítja meg, egyetlen attribútuma azt adja meg egy boolean-ként, hogy az adott bábu fehér-e, vagy fekete. Abban az esetben, ha fehér, az isWhite, azaz a szín meghatározásáért felelős attribútuma igaz értéket vesz fel, ha fekete, akkor pedig false-ot. Ez az osztály azért fontos, mivel minden mezőnek van egy piece attribútuma, de az üres mező esetén ez null értéket vesz fel, így könnyebb megtalálni azokat a mezőket, amelyeken van, és amelyeken nincs bábu.

Konstruktora egy squareType-ot kap, amely által beállítja az isWhite attríbútumot true vagy false értékre. Egyetlen metódusa:

public boolean getIsWhite()

- Visszaadja az isWhite attribútumának értékét

3.3. A table osztály

Ez az osztály magát a táblát valósítja meg, attribútumai a következőket tárolják: egy square títpusú mezőkből álló mátrixot, egy SIZE nevű konstansot, amely a tábla méretét tárolja, azaz a sor, illetve oszlopszámot, egy square típusú, selectedSq nevű attríbútumot, amely a kiválasztott bábut tartalmazza, egy isWhite nevű boolean attribútumot, amely az aktuális játékos színét tartalmazza, egy isBot nevű boolean attribútumot, amely a botjátékos engedélyezéséért felelős, tartalmaz egy mv nevű move objektumot, ez felel majd a bábuk helyes mozgásáért, és egy random nevű Random oobjektumot, amely a random szám generálásáért felelős.

A tábla konstruktora kezdeti helyzetbe állítja a táblát, minden mezőnek beállítva az actionListenert, létrehoz egy új move objektumot, amelyet az mv attribútuma kap meg.

A table osztály számos getterrel, setterrel, meg a bábuk léptetéséért felelős metódussal rendelkezik. A következő metódusai vannak:

public void initTable()

- Ez a metódus felel a tábla kezdeti helyzetbe állításáért, létrehozza és feltölti mezőkkel a board mátrixot, beállítja a kiválasztott bábut null értékűre, a bot játékost kikapcsolja, és a kezdő játékosnak a fehér színűt állítja
- A metódus létrehozza a tábla mezőit, beállítja a mező színeit, és a szabályoknak megfelelően elhelyezi a bábukat a mezőkön

public void setActionListener()

- Ez a metódus a tábla minden mezőjének, azaz square objektumának beállít egy ActionListenert

Ez az ActionListener egy checkForMove objektum lesz, amely konstruktorában megkapja azt a mezőt, amelyre beállítottuk. Egyetlen metódusa, a *public void actionPerformed(ActionEvent e)* a tábla mv attribútumának meghívja a *public void movePiece(square sq)* metódusát a konstruktorában kapott mezőre.

public square getSquare(int i, int j)

- Visszaadja a megadott pozícióban levő mezőt

public int getSiz()

- Visszaadja a tábla méretét

public void setSelected(square sq)

- Beállítja a kiválasztott mezőt

public square getSelected()

- Visszaadja a kiválasztott mezőt, értéke null, ha nincs mező kiválasztva

public boolean getIsWhiteTurn()

- Igazat ad vissza, ha a fehér játékos van soron, fekete esetben hamisat

public void setIsWhiteTurn(boolean isWhite)

- Beállíthatjuk a soron következő játékost

public boolean getIsBot()

- Igazat ad vissza, ha a bot játékos aktiválva van, elenkező esetben hamisat

public void setIsBot(boolean bot)

- Engedélyezni, illetve megszüntetni lehet a botjátékost

public void resetColors()

- Ez a metódus felel a tábla mezői színeinek alaphelyzetbe állítására

public boolean isValidMove(square sq1, square sq2)

- Megnézi, ha az sq1 mezőn elhelyezkedő bábuval léphetünk-e az sq2 mezőre, ennek függvényében ad igaz vagy hamis értéket
- A getMoves metódust meghívva az sq1 mezőre ez visszaadja az sq1 mezőn elhelyezkedő bábu összes lehetséges a szabályoknak megfelelő lépését egy square listában, ha az sq2 benne van ebben a listában, akkor odaléphetünk

public List<square> getMoves(square sq)

- Ez a metódus egy square listában adja vissza a kapott mezőn elhelyezkedő bábu összes a szabályoknak megfelelő lépését
- Először a tábla canBeat metódusával megnézi, ha a megadott mezőn elhelyezkedő bábu ütni tud-e, ezután megnézi a mező típusát, és annak függvényében hív meg metódusokat, amelyek megtalálják és visszaadják egy square listában a megadott mezőn elhelyezkedő bábu összes a szabályoknak megfelelő lépését

public List<square> getBeatMovesBlack(square sq)

- Ezt a metódus egy olyan mezőre lehet alkalmazni, amelyen egy fekete sima bábu, vagy fehér, illetve fekete dáma helyezkedik el, amely ütni tud
- A metódus visszaadja az adott mező bábujának összes lehetséges lépését egy square listában
- A metódus megvizsgálja, hogy a két sorszámmal nagyobb, és két oszlopszámmal nagyobb, mező üres-e, ha igen, akkor megvizsgálja, hogy az egy sorszámmal és oszlopszámmal nagyobb mezőben a kapott mezővel ellentétes színű bábu van-e, ha igen, az előbb vizsgált üres mezőt hozzáveszi a szabályos lépések listájához

- Ezután a metódus megvizsgálja, hogy a két sorszámmal nagyobb, és két oszlopszámmal kisebb, mező üres-e, ha igen, akkor megvizsgálja, hogy az egy sorszámmal nagyobb és egy oszlopszámmal kisebb mezőben a kapott mezővel ellentétes színű bábu van-e, ha igen, az előbb vizsgált üres mezőt hozzáveszi a szabályos lépések listájához

public List<square> getBeatMovesWhite(square sq)

- Ez a metódus nagyon hasonló az előzőhöz, azzal a különbséggel, hogy fehér sima bábut tartalmazó mezőre, illetve fehér és fekete dámát tartalmazó mezőre alkalmazhatjuk, amelyek ütni tudnak
- Ugyanúgy egy egy square listát ad vissza, amely tartalmazza a mezőn elhelyezkedő bábu összes szabályos lépését

public List<square> getMovesBlack(square sq)

- Ezt a metódust olyan mezőre lehet alkalmazni, amelyen egy fekete bábu, vagy egy fekete, illetve fehér dáma helyezkedik el, és amelyekkel nem lehet ütni
- A metódus egy square listában adja vissza a bábu szabályos lépéseit
- Megvizsgálja, ha az eggyel nagyobb oszlopszámmal és sorszámmal rendelkező mező üres-e, ha igen felveszi ezt a listához, ezután megvizsgálja, ha az eggyel nagyobb sorszámmal és eggyel kisebb oszlopszámmal rendelkező mező üres-e, ha igen, felveszi a listához

public List<square> getMovesWhite(square sq)

- Ez a metódus hasonló az előzőhöz, azzal a különbséggel, hogy fehér bábut, illetve fehér vagy fekete dámát tartalmazó mezőre alkalmazható, amelyekkel nem lehet ütni
- A metódus egy square listában adja vissza az adott mezőn elhelyezkedő bábu szabályos lépéseit

public void showMoves(square sq)

- Ez a metódus kiszínezi azokat a mezőket, amelyekre a kapott sq mezőn elhelyezkedő bábu lépni tud
- Megnézi, ha a mezőn elhelyezkedő bábu ütni tud, és ennek függvényében hív meg metódusokat, amelyek megszínezik a lehetséges lépéseknek függvényében a tábla egyes mezőit

public void showBeatMovesBlack(square sq)

- Ez a metódus olyan mezőkre alkalmazható, amelyeken egy sima fekete vagy fekete, illetve fehér dáma található, amelyek ütni tudnak
- A kapott mezőn elhelyezkedő bábu lehetséges lépései alapján kiszínezi a tábla mezőit

public void showBeatMovesWhite(square sq)

- Ez a metódus hasonló az előzőhöz, azzal a különbséggel, hogy fehér sima bábut, fehér, illetve fekete dámát tartalmazó mezőre alkalmazható, amelyek ütni tudnak

- Az előbbi két metódus a lehetséges lépések mezőit pirosra színezi, ezzel jelezve, hogy kötelezően oda kell lépni, mivel ütéshelyzet áll fenn

public void showMovesBlack(square sq)

- Ez a metódus ugyancsak kiszínezi a táblán azokat a mezőket, amelyekre a kapott mezőn elhelyezkedő bábu lépni tud, azonban csak olyan mezőre alkalmazhatjuk, amely fekete bábut, vagy fehér, illetve fekete dámát tartalmaz, amelyek nem tudnak ütni

public void showMovesWhite(square sq)

- Ez a metódus hasonló az előzőhöz, azzal a különbséggel, hogy csak olyan mezőre alkalmazható, amely fehér bábut, fehér dámát, vagy fekete dámát tartalmaz, amelyek nem tudnak ütni
- Az előbbi két metódus a lehetséges lépések mezőit zöld színűre színezi

public boolean findBeat(boolean isWhitePlayer)

- Ez a metódus igazat ad, ha az adott színű játékos valamelyik bábujával ütni tud
- Ezt úgy vizsgálja meg, hogy végigmegy a tábla összes mezőjén, és hogyha a mező olyan színű bábut tartalmaz, amely a játékosé, megvizsgálja, ha ütni lehet vele, ennek függvényében tér vissza igaz értékkel, vagy folytatja tovább a keresést
- Ha már elfogytak a mezők és nem talált ilyen bábut hamis értékkel tér vissza

public boolean canBeat(square sq)

- Megnézi, hogy a kapott mezőn elhelyezkedő bábuval lehet-e ütni, és ennek függvényében ad vissza igaz vagy hamis eredményt
- A metódus kezdetben megvizsgálja, hogy milyen típusú mezőt kapott, ennek függvényében hív meg metódusokat, amelyek megvizsgálják, hogy a kapott mezőn elhelyezkedő bábu ütni tud-e

public boolean canBeatBlack(square sq)

- Ez a metódus fekete sima, fekete és fehér dáma bábut tartalmazó mezőkre alkalmazható
- Igaz értéket ad vissza, ha a mezőn elhelyezkedő bábuval ütni lehet

public boolean canBeatWhite(square sq)

- Ez a metódus hasonló az előzőhöz, azzal a különbséggel, hogy csak fehér sima, fekete és fehér dáma bábut tartalmazó mezőkre alkalmazható

public square getBeatPiece(boolean isWhiteTurn)

- A metódus visszaadja az első olyan mezőt, amelyen elhelyezkedő bábuval a megadott színű játékos ütni tud
- A metódus végignézi a tábla összes mezőjét, majd ha talál egy ilyen mezőt, akkor ezt visszaadja

- Ha már végigment az összes mezőn és nem talált a feltételnek megfelelőt, akkor null értéket ad vissza

public List<square> getMovablePieces(boolean isWhitePlayer)

- A metódus visszaadja egy square listában a táblán elhelyezkedő összes olyan mezőt, amely olyan bábut tartalmaz, amellyel a megadott színű játékos lépni tud
- A metódus végigmegy a tábla összes mezőjén, majd ha olyan mezőt talál, amely a játékos bábuját tartalmazza megnézi, ha lépni tud-e vele a játékos, azaz a getMoves metódus nem egy üres listát ad-e vissza, ennek függvényében veszi fel a mezőt a listába

public void botMove(boolean isWhitePlayer)

- Ez a metódus felel a robot játékos lépéséért, paraméterként egy booleant kap, amely megadja, hogy milyen színű játékosként kell lépjen
- A metódus megvizsgálja, ha van-e olyan bábu a táblán, amely a robot játékosé, és azzal ütni tud
- Ha talál ilyet, akkor kiválasztja az elsőt, és ha több ütési lehetősége van az adott bábunak random kiválaszt egyet, és üt vele
- Ütés után leellenőrzi, hogy nem érte el a bábu a tábla végét, ekkor dámává változtatja a bábut
- Ezt addig folytatja, amig a kiválasztott bábuval ütni tud
- Ha nincs olyan bábuja a robot játékosnak, amellyel ütni tud, random kiválaszt egyet azok közül, amelyekkel lépni tud, majd ennek a lehetséges lépései közül is kiválaszt egyet, és lépteti a bábut
- Itt is leellenőrzi, hogy a bábu nem érte-e el a tábla végét, ekkor a bábut dámává változtatja

public void removeBeated(square sq1,square sq2)

- Ez a metódus üresre állítja azt a mezőt, amely az sq1 és sq2 között van, azaz kitörli a leütött bábut a tábláról

public boolean reached(square sq1, square sq2)

- A metódus megvizsgálja, ha az sq2 mezőn levő bábu az sq1 mezőre lép, akkor eléri-e a tábla végét

public boolean checkWin(boolean isWhiteTurn)

- A metódus megnézi, ha az adott színű játékos nyert-e, ha igen, egy JOptionPane-n keresztül értesülünk róla

public boolean isWin(boolean isWhiteTurn)

- Igazat ad vissza, ha a megadott színű játékos nyert
- Megvizsgálja, ha az ellenkező színű játékosnak van-e olyan bábuja, amivel lépni tud, ha nincs igazzal tér vissza

3.4. A move osztály

Ez az osztály a bábuk mozgatását valósítja meg. Egy table attribútumot tartalmaz, amely az a tábla lesz, ahol a bábukat mozgatni akarjuk. Egy másik attribútuma egy boolean típusú disableSelect nevű attribútum. Ez biztosítja, hogyha egy játékos a bábujával ütött, és utána újra tud ütni, akkor ne tudjon más bábut kiválasztani a tábláról. Konstruktora egy table objektumot kap, ezzel teszi egyenlővé a table t attribútumát. A move osztály a következő metódusokkal rendelkezik:

public void movePiece(square sq)

- Ez a metódus a táblán történő helyes lépésért felelős, egy square típusú mezőt kap paraméterként
- A metódus először megnézi, hogy már választottunk-e ki a táblán bábut, ezt a tábla setSelectedSq attribútuma tárolja, valójában ez egy mezőt tárol el, amelyen a bábu van, ha igen, akkor megnézi, hogy a most kiválasztott mező üres-e, ekkor két eset lehet, ha a táblán kiválasztott bábu tud ütni, akkor a beatMoveTo metódussal megvizsgáljuk, hogy az újonnan kiválasztott mezőre lépés helyes ütést eredményez-e, és ha igen odalépünk, ha a táblán kiválasztott bábu nem tud ütni, akkor a simpleMoveTo metódussal, megnézzük, hogy az újonnan kiválasztott mezőre lépés ezzel a bábuval helyes lépést eredményez-e
- Ha a most kiválasztott mező nem üres, és már a táblán választottunk ki bábut, és a disableSelect nem aktív, akkor a setSelectedSquare metódust meghívva az újonnan kiválasztott mezőre ez eldönti, hogy felülírja a táblán kiválasztott bábu mezőjét vagy sem
- Ha még a táblán nincs kiválasztott bábu, és az újonnan kiválasztott mező egy bábut tartalmaz, akkor a setSelectedSquare metódus eldönti, hogy ez legyen-e a táblán a kiválasztott bábu vagy sem

public void beatMoveTo(square sq)

- Ha a táblán már van bábu kiválasztva, amely ütni tud, akkor ez a metódus leellenőrzi, hogy a bábuval a kapott mezőre lépés szabályos ütést eredményez-e, és elvégzi a lépést
- Először leellenőrzi, hogy szabályos-e a lépés, majd törli az ütött bábut, ezután megnézi, hogyha a bábuval a kijelölt mezőre lépünk nem értük-e el a tábla végét, és ennek függvényébe a kijelölt mezőbe, ahova a bábunak lépnie kell egy dámát vagy csak a bábut helyezi el
- Ezután a tábla mezőinek színeit a kezdeti helyzetbe állítja, és törli a tábláról a kijelölt bábut
- Megnézi, hogyha a lépés után a bábu, amellyel léptünk tud-e még ütni, ha igen, akkor kijelöli ezt a bábut a táblán, a disableSelect attribútumot igaz értékre állítja, azaz nem engedi, hogy más mezőt kiválasszunk, és a táblán szemlélteti, hogy hova tudunk lépni a kiválasztott bábuval, ezt addig folytatja, amig a bábuval tudunk ütni
- A metódus leellenőrzi, hogyha a lépés során nyert-e a játékos

- A metódus a végén leellenőrzi, hogyha aktiválva van-e a botjátékos, ekkor a botjátékos lép egyet valamelyik bábujával, és leellenőrzi, hogy a lépés után nyert-e, ha nincs aktiválva, akkor beállítja, hogy a következő lépésnél az ellenkező színű játékos következzen

public void simpleMoveTo(square sq)

- Ez a metódus felel a táblán az egyszerű lépés helyes végrehajtásáért
- Kezdetben leellenőrzi, hogyha a táblán kiválasztott bábuval a kapott mezőre lépünk, az szabályos lépést eredményez-e
- Ha igen, akkor megnézi, hogy a lépés során elértük-e a tábla végét vagy sem, ennek függvényében a mezőbe, ahova lépünk egy dáma, vagy csak a bábu kerül, amellyel léptünk, a bábu eredeti helyét pedig üresre állítja
- A metódus kezdeti helyzetbe állítja a tábla színét, és törli a táblán kijelölt bábut
- A metódus leellenőrzi, hogy a lépés után az adott játékos nyert-e
- Abban az esetben, ha a botjátékos aktiválva van, akkor lép egyet a botjátékos, ezután is leellenőrzi, hogyha a botjátékos nyert-e

public void setSelectedSquare(square sq)

- Ez a metódus választja ki a táblán a bábut
- Először megnézi, ha a játékosnak van-e olyan bábuja, amellyel ütni tud, mivel ekkor kötelezően ezzel kell lépnie, így nem választhat ki olyan bábut, amellyel nem tud ütni, és ha a megadott mezőben egy olyan bábu van, amellyel ütni lehet, akkor ez lesz a táblán a kiválasztott bábu
- Ha nincs olyan bábuja a játékosnak, amellyel ütni tud, akkor egyszerűen csak beállítja a megadott mezőn levő bábut a táblán kiválasztott bábunak

A move osztály segítségével csak szabályosan léphetünk a táblán, figyel, hogy minden lépés után más játékos következzen, vagy botjátékos esetén figyel arra, hogy ez lépjen, és segít a játék lebonyolításában, és a játék végkimenetelének eldöntésében

3.5. A window osztály

Ez az osztály a tábla megjelenítését, valamint a menügombok megjelenítését valósítja meg. A JFrame osztályból származik le. A következő attribútumokkal rendelkezik: egy t nevű table attribútum, amely a megjelenítendő táblát tartalmazza, egy board nevű JPanel, amely ugyancsak a tábla megjelenítéséért felelős, egy mb nevű JMenuBar-t, egy newGame, toFile, endGame nevű JMenu-t, és egy oneVone, oneVbot, save, load, exit nevű JMenuItem-et, ezek a program funkcióinak kiválasztását teszik lehetővé, illetve tartalmaz meg egy from_File nevű fileLoader, és egy to_File nevű fileSaver objektumot, ezek a tábla kiíratását és betöltését teszik lehetővé fájlokba. A window konstruktora létrehozza a táblát, beállítja a menü sort, a JMenuItem típusú objektumokhoz actionListenereket rendel, beállítja a tábla paneljét, és végül beállítja a megjelenítendő ablakot. A tábla a következő metódusokkal rendelkezik:

public void actionPerformed(ActionEvent e)

- Ez a metódus felel a program funkcióinak megvalósításáért, ha egy megadott JMenuItem-re kattintunk
- Ha az 1 VS 1-re kattintunk, akkor egy új játékot kezdünk két játékos módban
- Ha az 1 VS bot-ra kattintunk, akkor egy új játékot kezdünk egy játékos módban
- Ha a Save Game-re kattintunk, akkor megjelenik egy új ablak, ahova beírhatjuk, hogy melyik fájlba szeretnénk a táblát elmenteni, ezután a tábla elmentődik abba a fájlba
- Ha a Load Game-re kattintunk, akkor megjelenik egy új ablak, ahonnan az eddig elmentett táblák közül választhatunk egyet, ha kiválasztottuk, akkor ez fog megjelenítődni a képernyőn
- Ha az Exit-re kattintottunk kilépünk a játékból

public table getShowedTable()

- Ez a metódus adja vissza azt a táblán, amelyet éppen megjelenítünk

public void setBoardPanel()

- Ez a metódus állítja be a tábla paneljét

3.6. A fileSaver osztály

Ez az osztály valósítja meg a tábla fájlba mentését. A JDialog osztályból származik le. Attribútumai egy JTextField-et, amelybe a fájl nevét írjuk, valamint egy táblát, amit elmentünk a fájlba tartalmaznak. Konstruktora egy window objektumot kap, ez alapján hoz létre egy új ablakot, amely tartalmazza a JTextField-et, ahova beírhatjuk a fájl nevét, és ettől kapja a táblát, amit el kell menteni. A következő metódusokkal rendelkezik:

public void saveTableToFile(String fileName)

- Ez a metódus paraméterként kap egy fájlnevet, amelybe szerializáció segítségével elmenti a táblát

public void actionPerformed(ActionEvent e)

- Ha a megjelenő ablakon a Save Table gombra kattintunk, ha írtunk be fájlnevet, akkor ennek a metódusnak a hatására a kapott tábla elmentődik a tábla, ha nem egy hibaüzenet jelenik meg a képernyőn

3.7. A fileLoader osztály

Ez az osztály valósítja meg a tábla fájlból történő betöltését. Az osztály a JDialog-ból származik le. A következő attribútumokat tartalmazza: egy fileList nevű JList-et, és egy listModel nevű DefaultListModelt, amelyek a fájlok kilistázását valósítják meg, egy loadButton nevű JButton-t, egy DIRECTORY_PATH nevű String-et, amely az aktuális mappa nevét tartalmazza, illetve egy

t nevű table objektumot, ez lesz a betöltött tábla. Konstruktora egy window objektumot kap paraméterül, ez alapján hoz létre egy új ablakot, majd kilistázza az aktuális mappában található fájlokat, amelyek .txt-re végződnek. A következő metódusokkal rendelkezik:

public void loadFilesList()

- Ez a metódus felel a file-ok kilistázásáért az aktuális könyvtárból

public void loadTableFromFile(String fileName)

- Ez a metódus a paraméterében egy fájlnevet kap, amelyből szerializáció segítségével beolvassa a táblát

public table getLoadedTable()

- Ez a metódus adja vissza a betöltött táblát

public void actionPerformed(ActionEvent e)

- Ha a Load Table-re kattintunk, akkor ennek a metódusnak a hatására fog betöltődni a tábla, ha választottunk ki fájlt

3.8. A Main osztály

Ez az osztály hozza létre és teszi láthatóvá a játék ablakát, egyetlen metódusa a *public static void main(String[] args)*.

A program osztályai közül a table, a square, a piece és a move osztály implementálja a Serializable interfészt, így a táblát szerializáció segítségével menthetjük el.

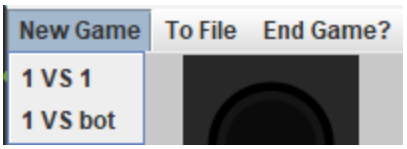
A window, fileLoader, fileSaver osztályok pedig az ActionListener osztályt implementálják.

4. Felhasználói kézikönyv

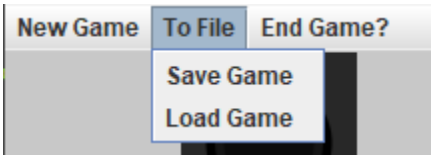
A program elindítása után egy 8x8-as sakktábla jelenik meg, melyre a bábuk a játék kezdéséhez már fel vannak állítva. A program indítás során egyből beállítja a játékot kezdeti helyzetbe és kétjátékos módba, így a fehér játékos azonnal kezdhet is.

A program egy menüsort is tartalmaz, ez három menüt: New Game, To File és End Game?.

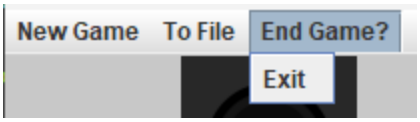
A New Game-re kattintva két lehetőség közül választhatunk: 1 VS 1 és 1 VS bot, az egyik a kétjátékos módot, a másik pedig az egy játékos módot indítja el.



A To File-ra kattintva ugyancsak két lehetőség közül választhatunk: Save Game és Load Game. Az első a tábla mentését teszi lehetővé, a másik pedig egy tábla betöltését teszi lehetővé fájlból.

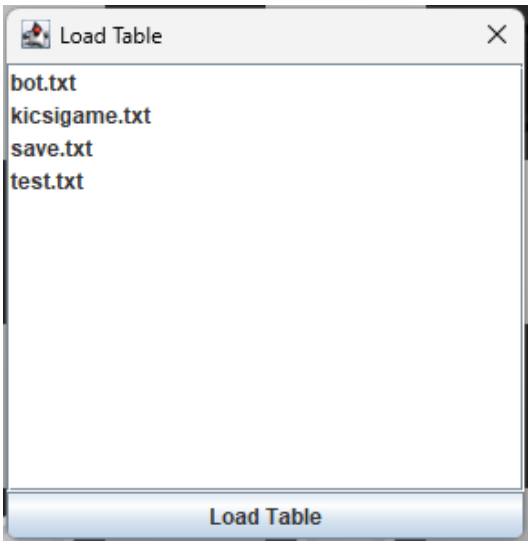


Az End Game? -re kattintva egy lehetőség közül választhatunk, az Exit, amelyre kattintva kilépünk a játékból.



A Save Game-ra kattintva egy új ablak jelenik meg Save Table néven. Ebben az ablakban adhatjuk meg a fájl nevét, ahova el szeretnénk menteni a játék jelenlegi helyzetét, majd a Save Table gombra kattintva megtörténik a mentés, ha megadtunk egy fájlnevet, ellenkező esetben a program értesít arról, hogy meg kell adni egy fájlnevet.

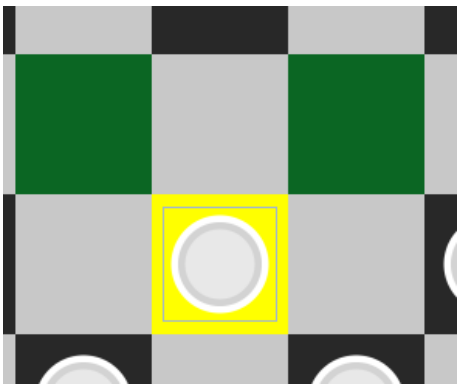
A Load Game-ra kattintva egy új ablak jelenik meg a Load Table néven, itt megjelennek az eddig elmentett játékok, amelyekből kattintással tudunk választani, és a Load Table gombra kattintva betölthetjük őket.



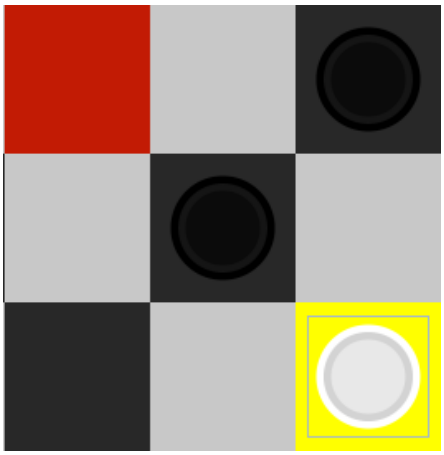
Az 1 VS 1 gombra kattintva a tábla a kezdeti állapotba állítódik, így elvesztjük az előző játékhelyzetet, ha még nem mentettük el, és a játékot kétjátékos módba állítja, azaz miután az egyik játékos lépett a bábujával a másik játékos következik.

Az 1 VS bot gombra kattints a tábla a kezdeti állapotba állítódik, így elvesztjük az előző játékhelyzetet, ha még nem mentettük el, és a játékot egyjátékos módba állítja, azaz miután a játékos lépett valamelyik bábujával a program magától lépni fog egy másik színű bábuval.

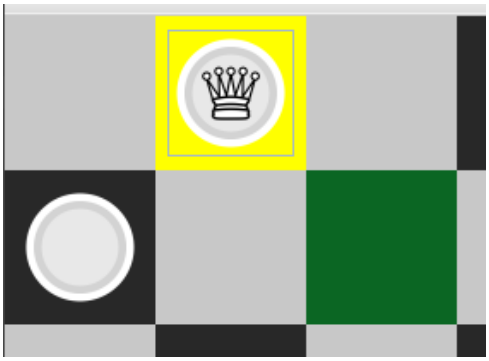
A játékos kiválasztani bábut úgy tud, hogy rákattint arra a mezőre, amelyikben a kiválasztott bábu van. Ezután a mező színe sárga színű lesz, és zölddé változnak azok a mezők, ahova a játékos a bábuval lépni tud.



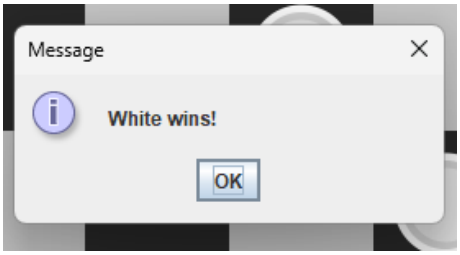
Abban az esetben, ha a játékosnak van olyan bábuja, amelyikkel ütni tud, nem tud kiválasztani olyan bábut, amellyel nem lehet ütni, így a csak akkor válik sárgává a bábu mezője, ha olyan bábu választunk ki, amely ütni tud. Ekkor piros színűre változnak azok a mezők, ahova a bábu lépve ütni tud, ezzel jelezve, hogy ütéshelyzet van, és kötelezően azok közül a mezők közül kell választani.



Ha a bábuval elértük a tábla végét, akkor a bábu automatikusan dámává változik, és ezzel már hátrafele is léphetünk és üthetünk.



Abban az esetben, ha egy játékos lépésképtelenné válik, a játék kiírja, hogy az ellenkező színű játékos nyert.



5. Tesztek

A tesztelés során a következő tesztoszályokat készítettem:

- squareTest: Itt tesztelem le a square osztály metódusait
- pieceTest:Itt tesztelem le a piece osztály metódusait
- tableTest: Itt tesztelem le a table osztály metódusait
- moveTest: Itt tesztelem le a move osztály metódusait, hogy a bábukkal történő lépések valóban a szabályok szerint történnek

- fileSaverLoaderTest: Itt tesztelem le, hogy a táblák mentése és fájlból történő betöltése valóban helyesen történik-e