Készítette:

Bodó József
Email: i7p4uq@inf.elte.hu
Csoportszám: 7

Feladat:

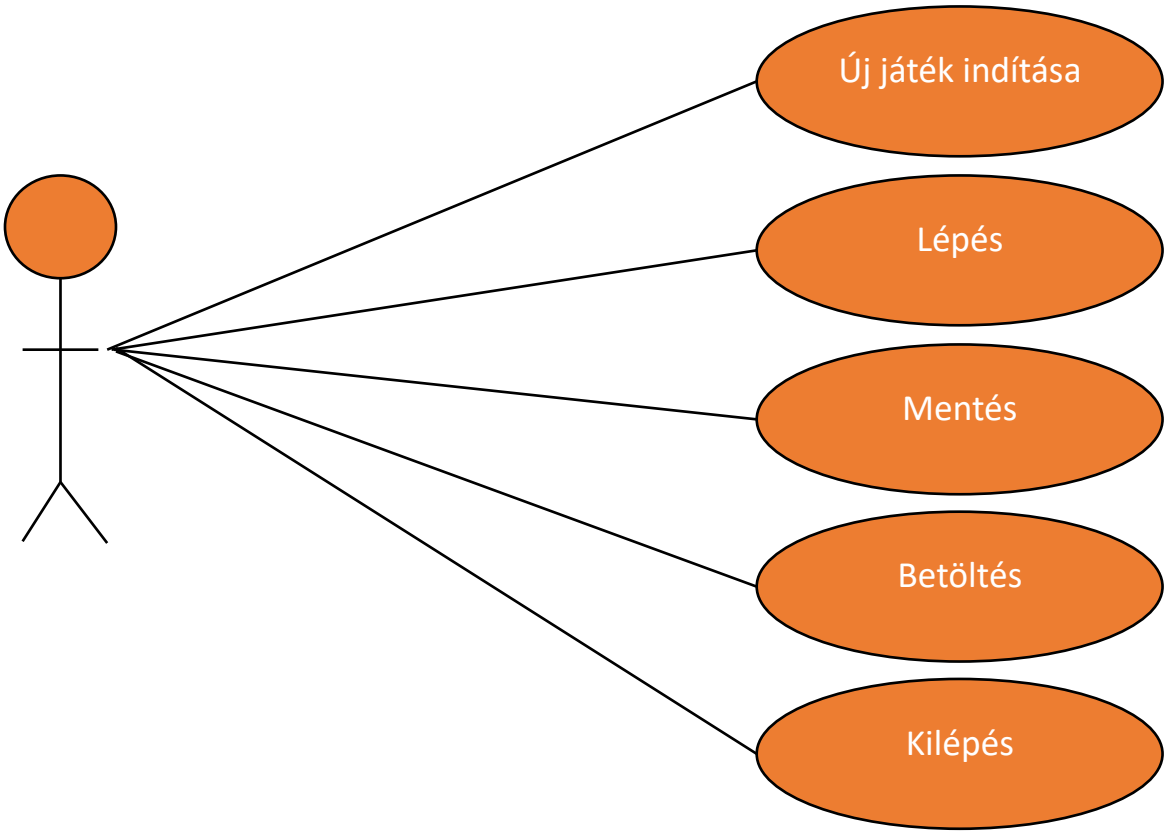
Készítsünk programot, amellyel a **Go** játék egyszerűsített változatát játszhatjuk.

Adott egy $n \times n$ pontból álló tábla, ahol a pontokra a játékosok felváltva köveket helyezhetnek (tradicionálisan fehér, illetve fekete színűt). A lerakott köveknek élete van, ami a négy szomszéd mezőből a szabad mezők száma. Egy csoport olyan kövek halmaza, amelyek szomszédosan összeérnek. A játékos akkor keríti be a másik játékos egy csoportját, ha azok élete elfogy. Ekkor a kövek fogságba kerülnek, és levehetőek a tábláról (ekkor a terület újra üres lesz, és lehet oda követ helyezni).

A játék meghatározott körszámig (n) tart, és célja minél több fogoly ejtése. Amennyiben ez egyenlő, a játék döntetlen. A program biztosítson lehetőséget új játék kezdésére a táblaméret megadásával (5×5, 9×9, 19×19), az aktuális játék mentésére és egy korábban elmentett játék betöltésére. Ismerje fel, ha vége a játéknak, és jelenítse meg, melyik játékos győzött.

A feladat elemzése:

A játékban két játékos vesz részt, de az alkalmazás szempontjából mindig csak egy felhasználó van. Ő ötféle tevékenységet végezhet:



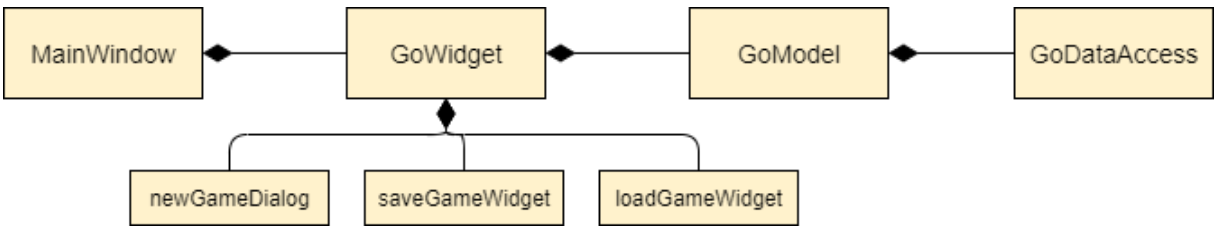
A felhasználói tevékenységek során az alábbi esetek következhetnek be.

1	Alkalmazás indítása	*GIVEN:*	Az alkalmazás telepítve van.
		WHEN:	Alkalmazás indítása.
		THEN:	Megjelenik egy 5x5-s üres játéktábla és az alkalmazás felülete.
2	Kilépés	*GIVEN:*	A játéktábla aktív.
		WHEN:	A játékfelület ablakának bezáró ikonjára kattintunk.
		THEN:	Az alkalmazás bezárul.
3a	Sima lépés	*GIVEN:*	A játéktábla aktív.
		WHEN:	Üres helyre kattintunk, de ez nem az utolsó lépés és nem történik kőlevétel.
		THEN:	Attól függően, hogy ki a soron következő játékos, egy fekete vagy fehér kő kerül a játéktáblára.
3b	Sima lépés	*GIVEN:*	A játéktábla aktív.
		WHEN:	Üres helyre kattintunk és ez az utolsó lépés.
		THEN:	Attól függően, hogy ki a soron következő játékos, egy fekete vagy fehér kő kerül a játéktáblára, és az alkalmazás ezután jelzi a játék végét.
4a	Lépés kőlevétellel	*GIVEN:*	A játéktábla aktív.
		WHEN:	Üres helyre kattintunk, ez nem az utolsó lépés és történik kőlevétel.
		THEN:	Attól függően, hogy ki a soron következő játékos, egy fekete vagy fehér kő kerül a játéktáblára, olyan formán, hogy így az ellenfél játékos köveit, amit bekerített, levesszük a játéktábláról.
4b	Lépés kőlevétellel	*GIVEN:*	A játéktábla aktív.
		WHEN:	Üres helyre kattintunk, ez az utolsó lépés és történik kőlevétel.
		THEN:	Attól függően, hogy ki a soron következő játékos, egy fekete vagy fehér kő kerül a játéktáblára, olyan formán, hogy így az ellenfél játékos köveit, amit bekerített, levesszük a játéktábláról és az alkalmazás ezután jelzi a játék végét.
4c	Lépés kőlevétellel	*GIVEN:*	A játéktábla aktív.
		WHEN:	Foglalt helyre kattintunk.
		THEN:	Nem változik a játék állapota.
5	Új játék	*GIVEN:*	A játéktábla aktív.
		WHEN:	Jelezzük az új játék indítási szándékunkat.
		THEN:	Megjelenik a konfiguráló ablak, amin megadhatjuk a táblaméretét és hogy meddig tartson az adott kör, majd megjelenik az üres játéktábla.
6a	Mentés elkezdése	*GIVEN:*	A játéktábla aktív.
		WHEN:	Jelezzük a mentési szándékunkat.
		THEN:	Megnyílik a dialógus a mentéshez, az első tárhely a kiválasztott.
6b	Mentési hely kiválasztása	*GIVEN:*	A mentés dialógusablaka aktív, egy tárhely kijelölve.
		WHEN:	Kiválasztunk egy tárhelyet.

		THEN:	A mentéshez megnyitott dialógus aktív, a kiválasztott tárhely van kijelölve.
6c	Mentés	*GIVEN:*	A mentés dialógus aktív, egy tárhely kijelölve
		WHEN:	Kijelölt tárhely üres, és mentést kezdeményezünk.
		THEN:	A mentés megtörténik, a dialógus bezárul.
6d	Mentés megszakítása	*GIVEN:*	A mentés dialógus aktív, egy tárhely kijelölve
		WHEN:	Elállunk a mentési szándékunktól.
		THEN:	A dialógus mentés nélkül bezárul.
7a	Betöltés elkezdése	*GIVEN:*	A játéktábla aktív.
		WHEN:	Jelezzük a betöltési szándékunkat.
		THEN:	Megnyílik a dialógus a betöltéshez, az első tárhely a kiválasztott.
7b	Betöltési hely kiválasztása	*GIVEN:*	A betöltés dialógusablaka aktív, egy tárhely kijelölve.
		WHEN:	Kiválasztunk egy tárhelyet.
		THEN:	A betöltéshez megnyitott dialógus aktív, a kiválasztott tárhely van kijelölve.
7c	Betöltés	*GIVEN:*	A betöltés dialógusablaka aktív, egy tárhely kijelölve.
		WHEN:	Kijelölt tárhely nem üres és betöltést kezdeményezünk.
		THEN:	A betöltés megtörténik, a dialógus bezárul.
7d	Betöltés – hiba	*GIVEN:*	A betöltés dialógusablaka aktív, egy tárhely kijelölve.
		WHEN:	Kijelölt tárhely üres és betöltést kezdeményezünk.
		THEN:	Hiba üzenetet kapunk.
7e	Betöltés megszakítása	*GIVEN:*	A betöltés dialógus aktív, egy tárhely kijelölve
		WHEN:	Elállunk a betöltési szándékunktól.
		THEN:	A dialógus betöltés nélkül bezárul.

Architektúra

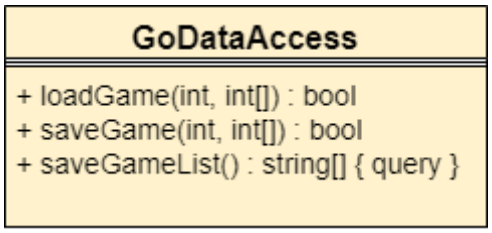
Az alkalmazást három rétegű (nézet-modell-perzisztencia) architektúrában valósítjuk meg. A nézet a játéktáblát megjelenítő ablak mellett három dialógus ablakot is tartalmaz az új játék, a mentés, illetve a betöltés párbeszédhez. A nézetet egy külön osztályban valósítjuk meg, amivel elérhetővé teszünk nyomógombokat is a három dialógus ablak előhívásához.



Perzisztencia

A korábban elmentett, legfeljebb 5 játékállást külön-külön állományokban tároljuk. Az adatelérési réteg egyetlen osztálya ezen állományokhoz történő hozzáférést biztosítja a modell számára úgy, hogy a modellnek nem kell tudnia arról, hogy milyen módon történik a tárolás.

Osztálydiagram:



Metódusok:

Az adatelérési réteg három metódust kínál.

A *saveGame*() egy megadott sorszámú állományba (ezt adjuk meg az első paraméterrel) tud elmenteni egy n[1] egész számmal leírt játékállást (ez a második paraméter).

A *loadGame*() egy megadott sorszámú állományból (ezt adjuk meg az első paraméterrel) tud betölteni egy n egész számmal leírt játékállást (ez a második paraméter).

A *saveGameList*() az elmentett állományok utolsó mentési dátumait adja vissza egy ötelemű sorozatban. A sorozat i-dik eleme üres, ha még nem létezik i-dik állomány.

Mentési fájl felépítése:

Az első négy sor, négy egyszerű adatot tartalmaz: a tábla méretét, az eddig megtett lépések számát, a soron következő játékost és az összes lépés számát.

Ezután mátrixosan ábrázolva (táblaméret sor, táblaméret oszlop) leírjuk a játéktábla állását.

A játéktábla után következő sorban eltároljuk azt, hogy hány lánccsoport jött létre a játéktáblán. Majd ezután felsoroljuk ezeket úgy, hogy az első sorban kiírjuk az adott lánc hosszát (hány kőből áll), a második sorban pedig ezeknek a köveknek a koordinátáit, és legvégül azt, hogy hány szabad területe van még a láncnak.

[1] függ a játéktábla nagyságától, illetve az eltárolt láncoktól

Modell:

A modell segítségével tároljuk az aktuális állást. A modell publikus metódusait a nézet hívja, a modell szignálok segítségével kommunikál a nézettel.

Osztálydiagram:

Adattagok:

A *stepNumber* az eddig megtett lépések számát tárolja, a *currentPlayer* a soron következő játékost, a *gameTable* pedig az aktuális játéktáblát mutatja. A *blackPoints* a fekete kövekkel levő játékos pontszámát, míg a *whitePoints* a fehér kövekkel játszó játékos pontszámát tárolja. A *tableSize* tárolja az aktuális tábla méretét. Az *emptyFields* a még szabad mezőket tárolja, míg a *numberOfSteps* a maximális lépések számát. A *chains*-t egy vektorral valósítjuk meg, ami *Chain* típusú adatokat tárol: az adott játékban létrejött láncokat tároljuk benne.

A *Chain* struktúrában két adattagunk van: a *freefields* ami az adott lánc még szabad mezőit tárolja, illetve a *members* vektor, ami az adott lánc tagjainak koordinátáit tartalmazza páronként.

Metódusok:

A privát adattagok értéke a *getCurrentPlayer*(), *getField*(), *getBlackPoints*(), *getWhitePoints*(), *getTableSize*(), *getNumberOfSteps*(), *getChainSize*(), *getStepNumber*() segítségével kérdezhetők le. A többi adattagnak nem szükséges getter, ugyanis azt maga a modell használja.

A *newGame*(int, int) egy új játék generálását biztosítja, amely paramétereiben megadjuk a táblaméretét, és a maximális lépések számát. A *stepGame*(int, int) a játék egy lépését végzi. A *stepGame*(int, int) metódus hívja meg a *checkNewChain*(int, int), illetve a *checkGameState*() metódust is, amely azt ellenőrzi, hogy vége van-e a játéknak.

A *checkNewChain*(int, int) metódus vizsgálja meg egy adott kőről, hogy új láncot hozott-e létre, ha nem, akkor meghívja *checkFriendlyNeighbours*(int, int)-t és a *checkEnemyNeighbours*(int, int)-t.

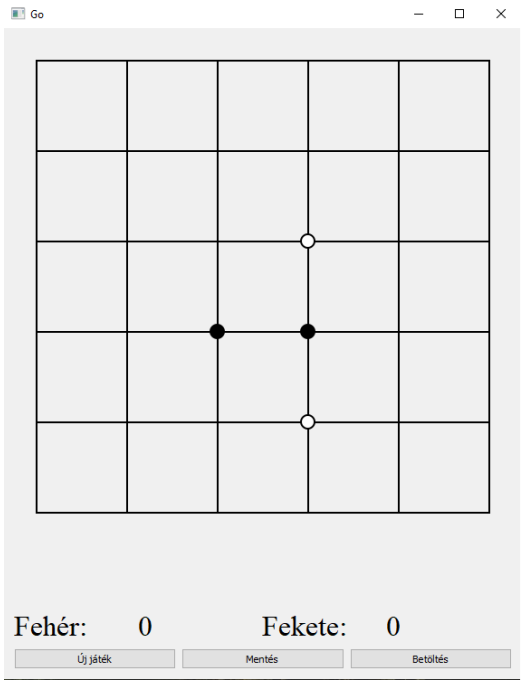
A *checkFriendlyNeighbours*(int, int) egyesíti a közös pontokkal rendelkező, azonos színű láncokat, ehhez meghívja a *mergeChainsOnCondition*(Chain, int, int)-t. A *checkEnemyNeighbours*(int, int) vizsgálja meg az ellenséges színű láncokat, amelyek szomszédosak, az adott lánccal és hívja meg a *deleteChainOnCondition*(Chain[] , int, int, Player)-t amely ellenőrzi az ellenséges színű láncokat, hogy kell-e azokat törölni.

Az *isNeighbour*(int, int, int, int), a *deleteChain*(Chain, Player), *checkLegalPlace*(int, int), a *countFreeFields*(Chain) függvények segédfüggvények. Az *isNeighbour*() két pontról dönti el, hogy szomszédosak-e, a *deleteChain*() segítségével törlünk egy meghatározott láncot a játéktábláról és hozzáadjuk a pontot a megfelelő játékoshoz, a *checkLegalPlace*() ellenőrzi, hogy létező mezőt vizsgálunk-e meg, és a *countFreeFields*() pedig megszámolja egy lánchoz tartozó üres mezők számát.

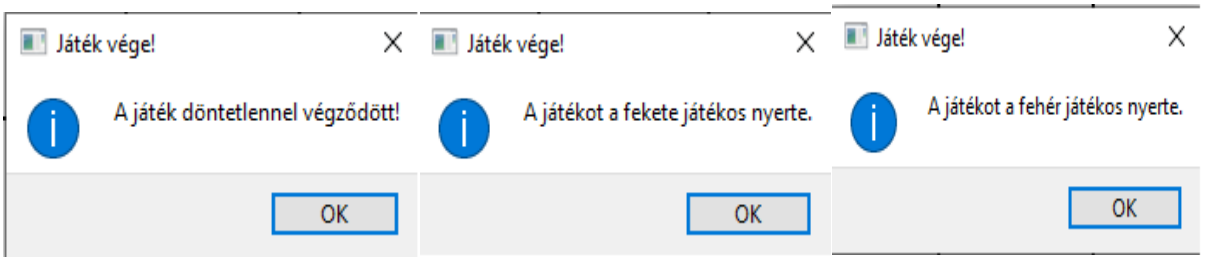
Nézet

Felhasználói felület terve

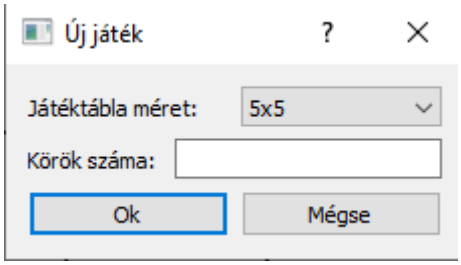
A játék (minimális méret mellett) tetszőlegesen átméretezhető felületén grafikusan kirajzolt fekete követ és fehér követ lehet elhelyezni a megadott pályaméretet jelző vonalak rácspontjain. Ezenkívül megjelenítjük az adott játékosok pontjainak számát (kezdetben 0) és lehetőséget adunk új játék elindítására, meglévő játék betöltésére vagy esetleg az éppen folyamatban lévő játék mentésére.



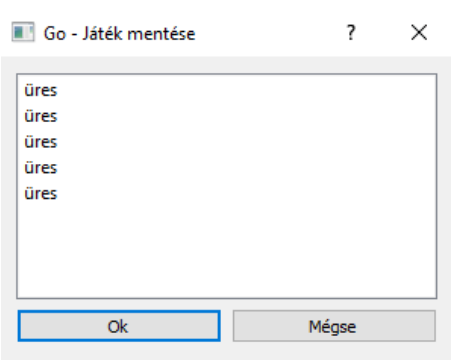
A játék végét üzenet-ablakok jelzik:

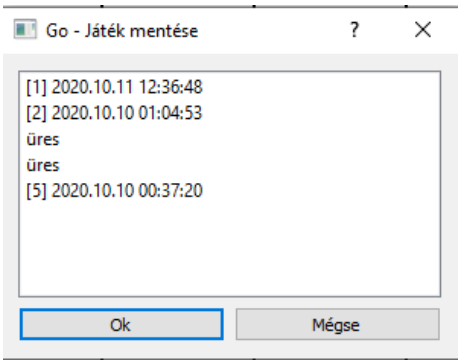


Új játék létrehozása dialógusablak segítségével zajlik.

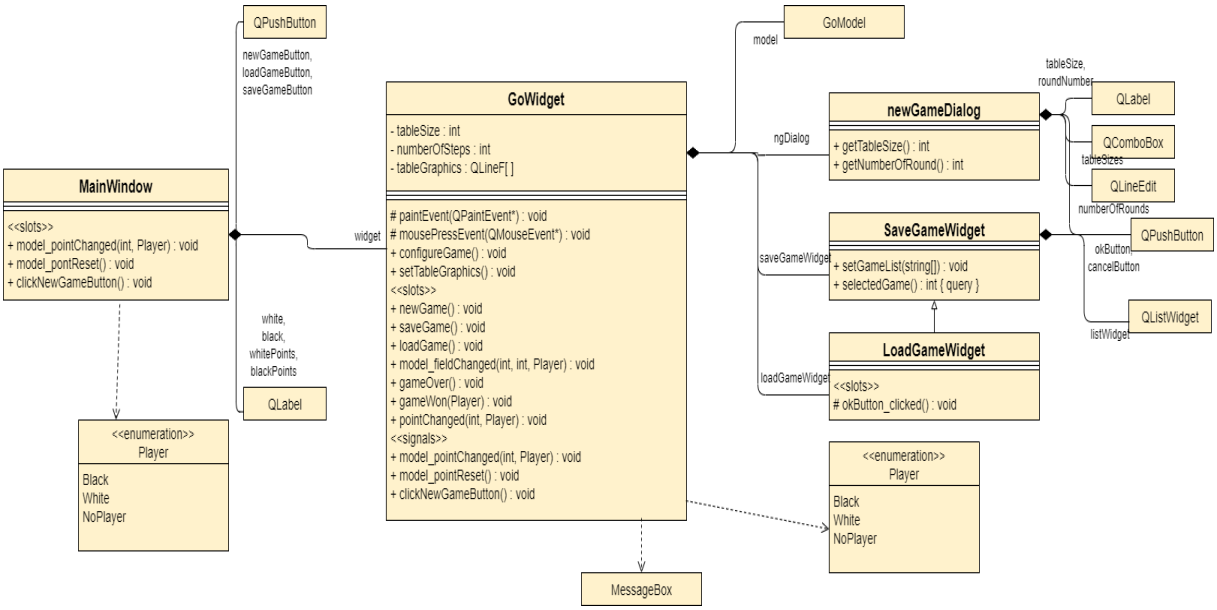


A mentés és betöltés dialógus-ablakokon keresztül zajlik. A hibaüzeneteket üzenet-ablakok segítségével jelezzük:





Osztálydiagram:



Vezérlők és adattagok

A játéknézet osztályának adattagjai a játék megjelenítésére szolgáló grafikai elemeket (tábla) tárolják, a táblaméretét, illetve a maximális lépések számát, valamint egy-egy hivatkozást az új játék, a mentés és a betöltés dialógus ablakokra és a modellre.

Metódusok

A játéknézet osztály felüldefiniálja a paintEvent() és a mousePressEvent() metódusokat. Az első felel a játékállás megjelenítéséért, a másik kezeli a felhasználói akciót.

A főablak metódusai a következők: *model_pointChanged*() változtatja meg a paraméterként kapott játékos pontszámát a főablakon, *model_pointReset*() állítja 0-ra a játékosok pontszámát és a *clickNewGameButton*() egy *clicked*() eseményt vált ki, mintha az új játék gombra kattintottak volna.

Eseménykezelés:

sender	signal	receiver	slot
modell	fieldChanged()	GoWidget	model_fieldChanged()
modell	gameOver()	GoWidget	gameOver()
modell	gameWon()	GoWidget	gameWon()
modell	pointChanged()	GoWidget	pointChanged()
GoWidget	model_pointChanged()	MainWindow	model_pointChanged()
GoWidget	model_pointReset()	MainWindow	model_pointReset()
GoWidget	clickNewGameButton()	MainWindow	clickNewGameButton()
saveGameWidget	accepted()	GoWidget	saveGame()
loadGameWidget	accepted()	GoWidget	loadGame()

Végfelhasználói tesztesetek:

	Teszteset	Elvárt hatás
1	Az alkalmazás indítása	Megjelenik az üres játéktábla.
2	Kilépés a folyó játékból	Az alkalmazás leáll.
3a	Váltakozó lépések	Az üres mezőkre kattintva váltakozva, hol fekete, hol fehér köveket rajzolunk ki a táblán.
3b	Nem üres mezőre kattintás	Nem történik változás a táblán.
3c	Véget ér a játék, mert elfogyott a lépéslehetőség, vagy lejárt a körök száma, az állás döntetlen	Megjelenik a döntetlen felirat egy információs ablakban, majd új játék kezdődik.
3d	Véget ér a játék, mert elfogyott a lépéslehetőség, vagy lejárt a körök száma, az egyik játékos pontja nagyobb a másikénál	Megjelenik a győztes egy információs ablakban, majd új játék indul.
4a	Új játék indítása egy folyamatban lévő játéknál	Megjelenik az új játékot konfiguráló ablak, majd új játék indul.
4b	Új játék indítása egy befejeződött játéknál	Megjelenik az új játékot konfiguráló ablak, majd új játék indul.
5a	Mentés egy folyamatban lévő játékról	A program elmenti az aktuális játékállást, majd utána lehet folytatni a játékot.
5b	Mentés egy befejeződött játékról	A program elmenti az aktuális játékállást, majd utána új játékot lehet kezdeni.
5c	Mentés elvetése	Nem történik változás.
6a	Betöltés egy folyamatban lévő játéknál	Az elmentett játék betöltődik, azzal lehet tovább játszani.
6b	Betöltés egy befejezett játéknál	Az elmentett játék betöltődik, azzal lehet tovább játszani.
6c	Betöltés, de kiválasztott játék nélkül	Figyelmeztető üzenetablak, de más változás nem történik.
6d	Betöltés elvetése	Nem történik változás.
7a	Láncok létrejötte	Amikor egy követ lerakunk, új lánc jön létre.
7b	Láncok egyesülése	Amikor egy kő mellé, lerakunk egy azonos színűt, a két lánc egyesül.
7c	Láncok törlése	Ha egy láncnak elfogy a szabad mezőinek száma, akkor törlődik a lánc, és eltűnik a lánc a tábláról.