Tartalomjegyzék

BattleSudoku

Folytathatjátok a múltórai project-et, de ha valakinek nem lenne meg: gyak9_megold.zip

KeyboardController

Írjunk egy **KeyboardController** nevû osztályt, mely örököl a **KeyAdapter**-bõl és a segítségével a játék érzékeli, ha egy számot nyomtunk le. (**KeyEvent.VK_1** az 1-es gomb és **KeyEvent.VK_9** a kilences, a többinek az értéke ezek között van)

- Legyen egy **Game** adattagja és konstruktora, hasonlóan a **MouseController**-hez.

- Írjuk meg a **keyPressed** metódusát, mely ha szám lenyomását érzékeli akkor hívja meg a tárolt **Game** adattag **keyPress** metódusát, paraméterként adja át neki a lenyomott számot. (Kiszámolható a szám a következõ módon, ha **keycode** a lenyomott gomb kódjának változóneve: **keycode - KeyEvent.VK_0**)
- Hibát kapunk hisz a **keyPress** metódusa a **Game**-nek még nem létezik, hozzuk ezt létre (létrehozhatjuk gyorsan ha a hiba felé visszük az egeret és a create...-ot választjuk)
- A **keyPress** metódus hívja meg a **gameBoard**-nak a **writeToSelectedIfPossible** metódusát a kapott számmal, majd hívja meg magán a **repaint** metódust (mint a **mouseClick**-nél).
- Hozzuk létre a **writeToSelectedIfPossible** metódust a **GameBoard** osztályban.
- Ez ellenõrizze, hogy ki van-e választva valami, ha nincs akkor ne csináljunk semmit, ha van valami, akkor nézzük meg, hogy üres-e a mezõ, ehhez írjunk a **Field** osztályba egy **isEmpty** metódust.
- Az **isEmpty** adjon vissza **true**-t ha a **number** adattagja **0**, különben **false**-t
- Fejezzük be most a **writeToSelectedIfPossible** metódust úgy, hogy ha üres a mezõ (azaz 0 van rajta), akkor hívja meg a **setNumber** metódusát a **selectedField**-nek, paraméterként adjuk meg a kapott számot.
- Írjuk meg a **setNumber** metódust, ami a **Field number** adattagját átállítja a kapott számra.

- Mindezek csak akkor fognak bármit is csinálni, ha a **KeyboardController**-t hozzáadjuk a Frame-hez. Szóval menjünk a **Game initUI** metódusához és az **addMouseListener** után írjátok az alábbit:

```
addKeyListener(new KeyboardController(this));
```

Ezzel hozzáadja az új osztályunkat, mint **KeyAdapter**-t.

Ezek után a kiválasztott mezõkben a 0-kat át tudjuk írni más számra.

Szebb kirajzolás

Módosítsuk a **GameBoard initBoard** metódusát a következõre:

```
private void initBoard() {
int t[] = {8, 0, 0, 4, 0, 6, 0, 0, 7,
                              0, 0, 0, 0, 0, 0, 4, 0, 0,
                              0, 1, 0, 0, 0, 0, 6, 5, 0,
                              5, 0, 9, 0, 3, 0, 7, 8, 0,
                              0, 0, 0, 0, 7, 0, 0, 0, 0,
                              0, 4, 8, 0, 2, 0, 1, 0, 3,
                              0, 5, 2, 0, 0, 0, 0, 9, 0,
                              0, 0, 1, 0, 0, 0, 0, 0, 0,
                              3};0, 0, 9, 0, 2, 0, 0, 5
for(int i = 0; i < drawables.length; i++) {
int x = (i % boardDimension) * Field.width;
int y = (i / boardDimension) * Field.height;

          [i] = new Field(x, y, t[i]);
}
}
```

Ez tényleg egy sudoku kezdeti állapota.

Most oldjuk meg, hogy ha egy mezõben 0 lenne, akkor oda ne is írjon ki semmit, hagyja üresen a mezõt. (A **Field** osztály **draw** metódusát kell módosítani.)

Beírható-e a mezõbe

Emlékezzetek vissza hogyan tároljuk a Field-eket jelenleg. (Nézzétek meg a **drawables** tömböt a **GameBoard**-ban.)

A következõ metódsokat a **GameBoard**-ba kell írni.

getFieldIndexByField

Írjuk egy metódust, ami visszaadja az indexét (egy **int**-et) a **Field**-nek a **drawables** tömbben, ami a paraméterként kapott **Field**-el egyezik meg. (Könnyû megírni a metódust a **getFieldByCoord** mintájára. Itt ==-vel kell összehasonlítani, mert azt akarjuk tudni, hogy pontosan az az objektum-e amit keresünk.)

Sorvizsgálat

Ez a metódus (nem mondom meg a nevét, ti találjátok ki) kap két **int**-et, az elsõ az, hogy hanyadik sorba akarjuk berakni, a második az, hogy melyik számot. Adjon vissza (igaz-hamis értéket), hogy az adott szám beilleszthetõ-e abba a sorba. (Lehet, hogy kell egy **getNumber** metódust írnunk a **Field**-nek.)

Teszteléshez módosítsuk a **writeToSelectedIfPossible** metódust, hogy vegye figyelembe a most megírt sorvizsgálatot is.

Oszlopvizsgálat

Ugyanaz mint a fenti, csak oszloppal.

Négyzetvizsgálat

Na ez a legbonyolultabb (3x3-as négyzetek vizsgálata), de találjatok ki valami ügyes megoldást rá. (Én mutatok majd egy kevésbé ügyeset.)

Bónusz feladatok

- Csináljátok meg, hogy a 3x3-as blokkokat vastagabb vonal válassza el egymástól.
- Oldjátok meg, hogy az általunk beírt mezõket lehessen módosítani, de az eredetileg bentlevõket már ne. (A legkönnyebb talán, ha a **Field**-nek bevezettek egy új adattagot, ami jelzi, hogy eredeti-e.)
- Ha az összes mezõt kitöltöttük, akkor jelezze ezt azzal, hogy egy nagy "http://wiki.math.bme.huYou're winner !"http://wiki.math.bme.hu felírat jelenjen meg a játék közepén.
- Gondolkozzatok el rajta, hogyan lehetett volna szebben megoldani a **selectedField**, **getFieldIndexByField**, **drawables** okozta "http://wiki.math.bme.hunem szép"http://wiki.math.bme.hu kódot.