Programozási technológia

II. Beadandó feladat

Boda Bálint
KDHPNI

2022. 11. 01

1. Feladat

Készítsünk programot, amellyel a következő két személyes játékot lehet játszani. Adott egy $n \times n$ mezőből álló tábla, amelynek mezői 0 és 4 közötti értékeket tartalmaznak. Kezdetben minden mezőn a 0 érték van. Ha a soron következő játékos a tábla egy tetszőleges mezőjét kiválasztja, akkor az adott mezőn és a szomszédos négy mezőn az aktuális érték eggyel nő felfelé, ha az még kisebb, mint 4. Aki a lépésével egy, vagy több mező értékét 4-re állítja, annyi pontot kap, ahány mezővel ezt megtette. A játékosok pontjait folyamatosan számoljuk, és a játékmezőn eltérő színnel jelezzük, hogy azt melyik játékos billentette 4-esre. A játék akkor ér véget, amikor minden mező értéke 4-et mutat. Az győz, akinek ekkor több pontja van.

A program biztosítson lehetőséget új játék kezdésére a táblaméret megadásával (3×3, 5×5, 7×7), és ismerje fel, ha vége a játéknak. Ekkor jelenítse meg, melyik játékos győzött, majd automatikusan kezdjen új játékot.

2. Terv

2.1. A feladat elemzése

A játék létrehozásához a következő dolgokat kell megvalósítanunk:

- játékosok (és pontszám)
- játéktábla és az azt alkotó cellák
- mező kiválasztása és az adott mezők értékének megváltoztatása
  - ellenőrizni kell, hogy mennyi az érték és az alapján pontot adni / nem engedni hogy a játékos olyan cellát válasszon ki aminek az értéke már 4
  - ellenőrizni hogy tart-e még a játék
- körváltás
- grafikus kezelőfelület

2.2. Típusok

2.2.1. Player

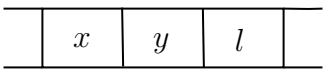
A játékosokat reprezentáló osztály, mely egyedül a játékos pontszámát tárolj el. Két művelete van a játékos pontszámának lekérése és a pontszám egyel való megnövelése.

2.2.2. Tile

A `Tile` osztály a tábla egyetlen egységét reprezentálja. Egyetlen adattagja az tárolja el hányszor érintette már egy játékos. A megérintés számát a `tipTile(Player player)` metódus növeli mely abban az esetben h a megnövelést követően a cella értéke 4 lesz igazat ad vissza. A "pöccintések" száma lekérhető egy getter metóduson keresztül.

2.2.3. GameModel

A `GameModel` osztály felel a játék belső logikájáért. Eltárolja magát a játéktáblát a játékosokat illetve az is jelenleg melyik játékos köre van. Konstruktora egy pozitív egész számot vár és ez alapján hoz létre egy $n \times n$-es táblát. A tábla egy adott cellája lekérdezhető a `getTile(int x, int y)` metódussal. Egy adott cella (és annak szomszédai) megpöccintését végzi a `tipTile(int x, int y)` metódus, mely üzenetek egy gyűjteményét adja vissza. Egy üzenet egy három elemű `int[]`, melynek formátuma a következő:



- x: az adott cella sorának indexe
- y: az adott cella oszlopának indexe
- l: 1, ha az (x, y) cella `tipTile(Player player)` metódusa igazat adott vissza, különben nulla

Az üzenet kezelése a játék nézetének (pl. GUI, parancssori kezelőfelület) felelőssége.

Az `getCurrentPlayer()` metódus visszaadja a jelenleg aktív játékost reprezentáló objektum referenciáját `getCurrentPlayerIndex()`, pedig ugyanezen játékos `players` tömbbeli indexét. Az `isOver()` metódus igazat ad vissza ha a játék véget ért azaz a játékosok pontszámának összege megegyezik a tábla méretének négyzetével. A `getWinner()` metódus visszaadja a több pontot elérő `players`-beli indexét. Döntetlen esetén (bár ez csak akkor lenne lehetséges ha páros méretű pályák is lennének) 1-et azaz a második játékos indexét adja vissza.

2.2.4. GameGUI

A `GameGUI` osztály felelős a játék kezelőfelületének megvalósításáért. Annak érdekében hogy a játék könnyen újraindítható legyen az osztály konstruktora csak az állandó komponenseket hozza létre.

A többi komponens és a játék logikájának elindítása a `setupGame(int boardSize)` feladata. A játék végén automatikusan lefut a `reset()` metódus, mely alaphelyzetbe állítja a játékot. A GUI főbb komponensei:

- új játék menü
- tábla (`JButton[][] board`)
- pontszámlálók (`JLabel[] scoreLabels`)
- körjelző (`JLabel turnLabel`)

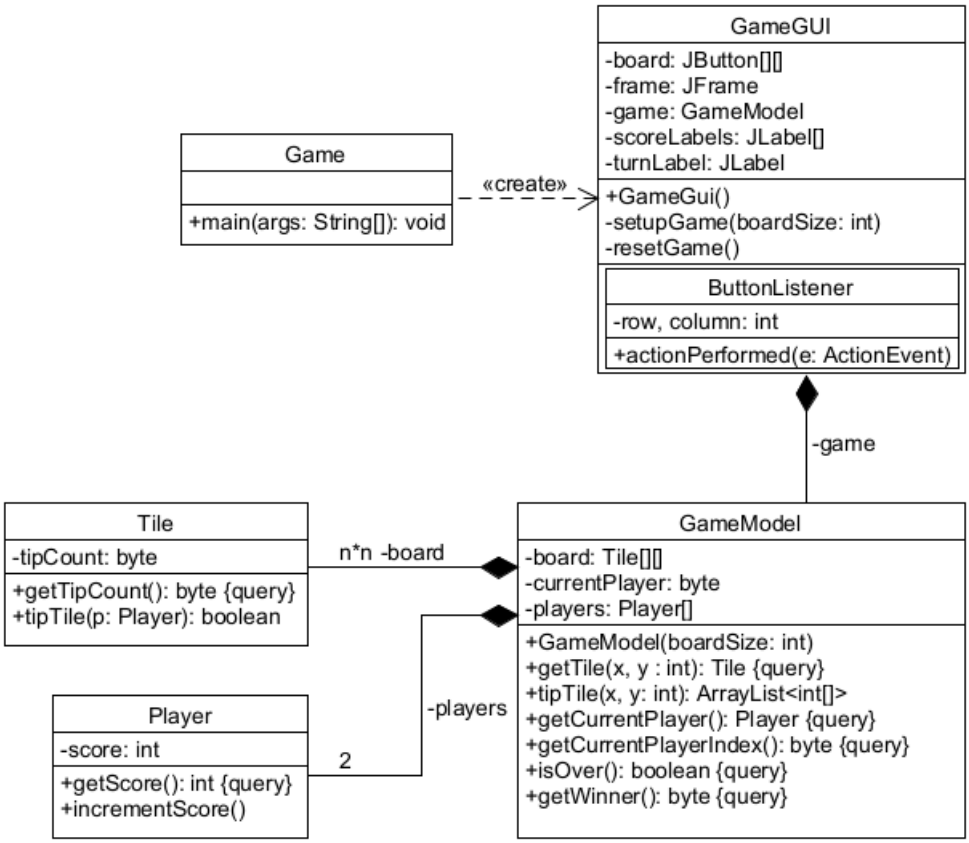
A konstruktorban létrejövő `JButton`-ökhöz mind megtalálható `ButtonListener` eseménykezelőt. Ezen osztály `actionPerformed()` metódusa kommunikál a `GameModel` osztállyal, úgy a következő módon:

1. lekéri a jelenlegi játékost
2. meghívja a cella `tipTile(Player player)` metódusát, és feldolgozza az ebből kapott üzenetet:
   (a) Ha egy cella értéke 4 lett átszínezi a gombot az adott játékos színére, kikapcsolja a gombot és leszedi az eseménykezelőt, frissíti az eredményjelzőt.
   (b) Frissíti a gomb szövegét a cella értékére
3. lépteti a kört a megfelelő metódussal, frissíti a körjelzőt
4. ellenőrzi tart-e még a játék az `isOver()` metódussal.

2.2.5. Game

A program fő osztálya, melynek egyetlen feladata egy `GameGUI` objektum példányosítása.

2.3. UML osztálydiagram



2.4. Implementálás

A terv implementálását a `Tile` és `Player` osztályoktól érdemes kezdeni felfelé haladva.
Java verzió: 17.0.4.1

3. Tesztelés

3.1. Fehérdobozos tesztesetek

Leírás	Tevékenység	Elvárt eredmény
Cella pöccintés cella értéke nem négy lesz	tile.tipTile()	Hamis
Cella pöccintés cella értéke négy lesz	tile.tipTile()	Igaz
Cella pöccintés cella értéke már volt négy	tile.tipTile()	Hamis
Cella értéke nem lehet több mint négy	tile.tipTile()	tile.getTipCount()==4
Illegális tábla méret	GameModel()	IllegalArgumentException

Fehérdobozos tesztesetekhez JUnit 4.13-as tesztkörnyezet: **GameTest.java**

3.2. Feketedobozos tesztesetek

Tevékenység	Elvárt eredmény
Program elindítása	Létrejön ablak a táblaméret választó menüvel
Új játék létrehozása a menüből	Létrejönnek a GUI komponensek, elindul a játék
Kattintás cellára	A cella és környező cellák értéke megváltozik, a játékos köre véget ér
Egy cella értéke 4 lesz	Az átpöccintő játékos pontszámának növelése, gomb kikapcsolása és átszínezése
Új játék létrehozása, úgy hogy, folyamatban van játék	Új GUI komponensek jönnek létre, új játék indul
Játék véget ér	Párbeszédablak jelenik meg, ami kiírja a nyertest
Párbeszédablak bezárása	Új játék indul