[Fénymotor](https://lengyel.web.elte.hu/node/64#comment-245) párbaj : Trongame dokumentáció

Programozási Technológia 1.

Második beadandó

Kuti Bence Gilbert

HS74YU

**A feladat**

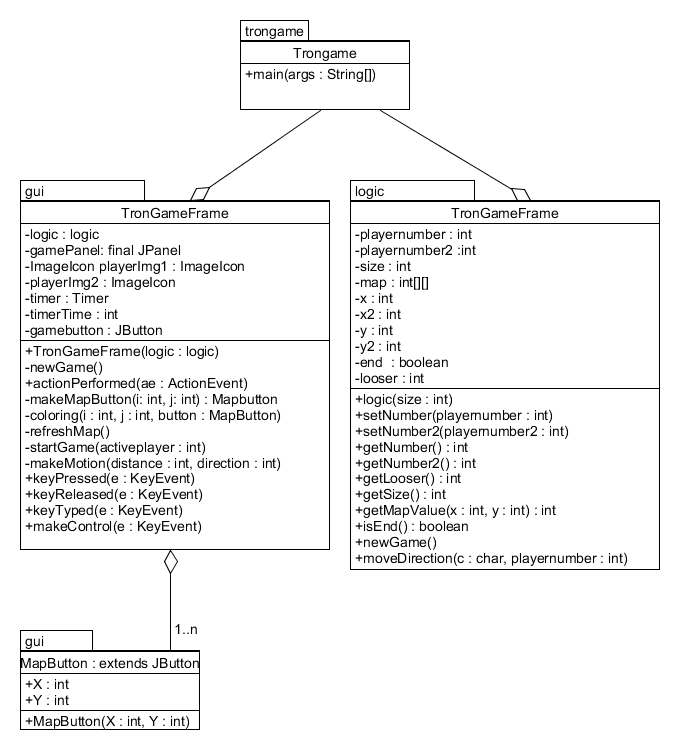
**Fénymotor párbaj : Trongame**

Adott egy n×n elemből álló játékpálya. A két játékos a bal, illetve jobb oldal közepén indul egy-egy fénymotorral, amely egyenesen halad (rögzített időközönként) a legutoljára beállított irányba (függőlegesen, vagy vízszintesen). A motorokkal lehetőség van balra, illetve jobbra fordulni. A fénymotor mozgás közben fénycsíkot húz, ami a játék végéig megmarad.  
Az a játékos veszít, aki előbb nekiütközik a másik játékos motorjának, bármelyikük fénycsíkjának vagy a pálya szélének.  
A program biztosítson lehetőséget új játék kezdésére a pályaméret megadásával (12×12, 24×24, 36×36), valamint a játék szüneteltetésére (ekkor nem telik az idő, és nem mozognak a motorok). Ismerje fel, ha vége a játéknak, és jelenítse meg, melyik játékos győzött.

**Tervezés**

A feladat főbb célja egy két játékos által játszható motoros játék volt ami leginkább a TRON című filmben látott fénymotor párbajra emlékeztet ezért is neveztem el a játékot Trongame-nek. Szükség van legalább 2 csomagra: egy ami a megjelenítést és egy ami a mozgatás megvalósítását végezni. Mivel 2 játékos mód ezért mindenképpen billentyűkkel való irányítást kell implementálni. Konstans irányba mozognak újabb gomb lenyomásáig a motorok megadott időközönként (Timer) Maga a pálya gombokból áll n\*n-es alapból 12\*12 ezt 12-vel léptetve megkapjuk a nagyobb pályaméreteket egészen 36\*36-ig. A játéktéren a gombok 5 állapotot kell tudniuk felvenni: a motor, b motor, a motor csíkja, b motor csíkja (1,2,3,4) ezenkívül a nulla az alapállapot amikor a mezőre mehet tovább a motor. Szükség van függvényre ami a mozgatások után mindig lefrissíti a gombok tartalmát, illetve a győztes megállapításához kellenek még függvények amik képesek eldönteni, hogy ki hibázott (ütközött falnak, magába, másikba) és ezáltal eldönteni, hogy ki nyert. Mivel a játékot meg is kell tudni állítani ezért kell pause és start gomb is ami a Timer segítségével kerül megvalósításra.

**UML Diagram:**

****

**Megvalósítás**

***A feladat alapján szükségünk van több csomagra:***

**gui**: vizuális megjelenítést és működést megvalósító osztályokat tartalmazza: MapButton.java, TronGameFrame.java

**logic**: a játék logikájáért felelős osztályt logic.java-t tartalmazza

**trongame**: a Trongame.java-t tartalmazza ami inicializálja a játékot

***Classok és metódusainak működése:***

**Trongame.java**: egy alapjátékot inícializál 12\*12-es játéktérrel (logic példányosítása), létrehozza a játékteret és láthatóvá teszi és futtatja.

**MapButton.java**: a JButton nevű beépített osztályt extendeli és létrehoz hozzá egy olyan konstruktort ami 2 int értéket vár meghíváskor.

**TronGameFrame.java:** a játék vizuális megjelenítéséért felelős metódusokat tartalmazza:

-konstruktorában beállítja az ablak címét, helyét méretét, layoutját. Létrehozza a játéktérhez szükséges gamePanelt. Inicializál egy timert, betölti a motorok képét, ha tudja és meghívja a newGame() metódust.

-newGame: megvalósítja a menüket és azok kinézetét illetve létrehozza a játékteret a gombokkal.

-makeMapButton(): a paraméterben megkapott 25\*25pixel nagyságú gombohoz társítja a tömbben tárolt értéket.

-coloring(): a paraméterben megkapott gombról eldönti, hogy milyen képi megjelenítést tesz lehetővé

1 érték esetén rárakja a gombra az egyes játékos motorját,

2 érték esetén a kettes játékos motorját,

3as érték esetén átszínezi a gombot zöldre az első játékos fénycsíkjának ábrázolásához,

4es érték esetén átszínezi a gombot pirosra a második játékos fénycsíkjának ábrázolásához.

-startGame(): létrehoz egy játék alaphelyzetet. A játék az isEnd() függvény igaz értékéig fut. Elindítja az egyes és kettes játékost és frissíti a pályát, majd lépteti a timert. Ha az isEnd() függvény igaz értékkel tér vissza akkor megállítja a timert és a játékosokat és eldönti a getLooser függvény segítségével, hogy ki nyert 0 érték esetén a zöld játékos, 1 érték esetén a piros játékos.

-makeMotion(): a konstans mozgás megvalósításáért felelős

0 direction érték esetén felfele mozgatja a kiválasztott karaktert

1 direction érték esetén lefele mozgatja a kiválasztott karaktert

2 direction érték esetén balra mozgatja a kiválasztott karaktert

3 direction érték esetén jobbra mozgatja a kiválasztott karaktert

-makeControl(): felelős a lenyomott gombok után való 0,1,2,3 direction érték kiadásáért amely segítségével eldönthető, hogy merre halad tovább a motor. Mozgatás wsad és nyilakkal lehetséges.

-keyPressed(), keyReleased(), keyTyped(): override függvények amelyek a billentyűleütések figyeléséért felelősek.

**logic.java**: a játéklogika megvalósításáért felelős.

-konstruktora: példányosítja egy n\*n-es tömbök mátrixát, elhelyezi a motorok értékét középre 1 és 0.

-setNumber() és setNumber2(): beállítja az aktuális játékos számát

-getNumber() és getNumber2(): lekéri az aktuális játékos számát

-getLooser(): visszatér a vesztő játékos számával

-getSize(): visszatér a játéktér dimenziójával (egyik oldalának méretével mivel n\*n-es)

-getMapValue(): visszatér az aktuális x,y koordinátán levő értékkel 0,1,2,3,4 amely a mozgatáshoz elengedhetetlen

-isEnd(): a játék végének meghatározását segíti egy igaz/hamis értékkel való visszatéréssel

-newGame(): amely 0 értékekkel tölti fel a pályát és a 2 motor helyét 1 és 2 értékkel

-moveDirection(): switch case szerkezettel a hozzá társított karakter érték segítségével u -up, d -down, l -left, r -right eldönti, hogy melyik műveletet kell alkalmazni a paraméterben megkapott játékoson. A case szerkezetek elvégzik a mozgatásokat, ha tud arra mozogni a játékos áthelyezik a játékost és ahol járt mező azt átállítja a hozzá tartozó fénycsík értékére. Ha hiba keletkezik (magába, falba, másikba megy) akkor eldönti, hogy ki a nyertes és visszatér ennek értékével.