Programozás alapjai 3. - házi feladat dokumentáció
Biró Ferenc
HR4VCG

1. Feladat ismertetése:

A “turmesz” egy 2D Turing-gép (angolul “Turmite”), amely általában egy négyzet alapú, két dimenziós rácson helyezkedik el, ahol minden cellának 2 állapota lehet (0 vagy 1), és a hangyának van egy aktuális pozíciója, állapota, és iránya. Minden lépésben a hangya a saját állapotának és a cella tartalmának megfelelően fordul, beleír a cellába, lép egyet, majd átmegy egy másik állapotba.

Ebben a megvalósításban most egy hatszög alapú háló van, így a hangya egy lépésben a következőket csinálja:

1. Fordul 60 fok valamilyen sokszorosával
2. Az aktuális cellába beleír valamit (0 vagy 1)
3. Előre lép egyet
4. Átmegy egy másik állapotba

Ennek szabályait a felhasználó adhatja meg, például egy lehetséges program:

		Cella állapota					
		0			1		
		Beír	Fordul	Köv. állapot	Beír	Fordul	Köv. állapot
Hangya állapota	0	1	1	0	1	1	1
	1	0	0	0	0	0	1

Tehát 0-s állapotban, ha 0-s cellán áll, akkor jobbra fordul 60 fokot, 1-est ír, és marad 0-s állapotban. Ha 1-es cellán áll, akkor jobbra fordul és átmegy 1-es állapotba. Amikor 1-es állapotban van, ha 0-t talál, akkor átmegy 0-s állapotba; ha 1-et talál, akkor azt 0-ra javítja, de marad 1-es állapotban.

2. Az osztályok leírása:

2.1 DirectionHex

A hatszög-rács koordináta rendszerének irányait jelölő enumeráció.

Attribútumok

-

Metódusok

+getVal(): int	Visszaadja az irány relatív értékét a pozitív X-hez képest.

2.2 HexTile

A hatszög-rács egy-egy mezőjét megvalósító osztály.

Attribútumok

-posX	Az X koordináta.
-posY	Az Y koordináta.
-posZ	Az Z koordináta.

Metódusok

+HexTile()	Létrehoz egy mezőt (0,0,0) koordinátákkal.
+HexTile(x: int, y: int, z: int)	Létrehoz egy mezőt a megadott koordinátákkal.
+getX(): int	Visszatér posX értékével.
+getY(): int	Visszatér posY értékével.
+getZ(): int	Visszatér posZ értékével.
+setX(x: int)	Beállítja posX értékét.
+setY(y: int)	Beállítja posY értékét.
+setZ(z: int)	Beállítja posZ értékét.
+equals(Object obj): boolean	Igazzal tér vissza, ha a koordináták páronként megegyeznek.
+hashCode(): int	A koordináták alapján generál egy (relatíve) egyedi számot.

2.3 HexAnt

A Turing hangyát megvalósító osztály.

Attribútumok

-dir: DirectionHex	Az irány, amerre a hangya épp néz.
-tile: HexTile	A mező, amin a hangya épp tartózkodik.
-state: int	A hangya állapota.

Metódusok

+HexAnt()	Létrehoz egy hangyát 0-s állapotban, pozitív X irányba nézve, a (0,0,0) koordinátájú mezőn.
+getDir(): DirectionHex	Visszatér dir értékével.
+getTile(): HexTile	Visszatér tile értékével.
+getState(): int	Visszatér state értékével.
+setDir(d: DirectionHex)	Beállítja dir értékét.
+setTile(t: HexTile)	Beállítja tile értékét.
+setState(s: int)	Beállítja state értékét.

2.4 StateKey

Két állapotból álló elempár, későbbiekben HashMapben való keresés kulcsa. (pl. hangya mezőjének állapota és saját állapota)

Attribútumok

-state1: int	Az egyik állapot.
-state2: int	A másik állapot.

Metódusok

+getState1(): int	Visszatér state1 értékével.
+getState2(): int	Visszatér state2 értékével.
+StateKey(s1: int, s2: int)	Létrehoz egy példányt a beadott állapotokkal.
+StateKey(st: Integer[])	Létrehoz egy példányt a beadott állapotokkal.
+equals(Object obj): boolean	Igazzal tér vissza, ha az állapotok páronként megegyeznek.
+hashCode(): int	Az állapotok alapján generál egy véletlenszámot.

2.5 HexCommand

A hangyának adott utasításokat megvalósító osztály.

Attribútumok

-rotation: int	Mennyit forduljon a hangya, relatívan a jelenlegi irányához.
-nextTileState: int	Mit írjon a hangya a mezőbe.
-nextAntState: int	A hangya következő állapota.

Metódusok

+getRotation(): int	Visszatér rotation értékével.
+getNextTileState(): int	Visszatér nextTileState értékével.
+getNextAntState(): int	Visszatér nextAntState értékével.
+HexCommand(r: int, ts: int, as: int)	Létrehoz egy példányt a beadott paraméterekkel. (rotation, nextTileState, nextAntState)
+HexCommand(cmd: Integer[])	Létrehoz egy példányt a beadott paraméterekkel. (rotation, nextTileState, nextAntState)

2.6 HexAntLogic

A hangyát vezérlő logikáért felelős osztály.

Attribútumok

-logic: ConcurrentHashMap<StateKey, HexCommand>	A hangya-mező állapotpárokhoz tartozó parancsokat tároló hashmap.

Metódusok

+HexAntLogic()	Létrehoz egy példányt, ahol minden állapotpárhoz a (0,0,0) parancsot rendeli.
HexAntLogic(hm: ConcurrentHashMap<StateKey, HexCommand>)	Létrehoz egy példányt a megadott hashmapből.
setLogic(hm: ConcurrentHashMap<StateKey, HexCommand>)	Beállítja a logikát a paraméterként beadott hashmapre.
getLogic(): ConcurrentHashMap<StateKey, HexCommand>	Visszaadja a logikáért felelős hashmapet.
getCommand(sk: StateKey): HexCommand	Visszaadja a paraméterként beadott állapotpárhoz tartozó parancsot.

2.7 Simulation

A szimulációért felelős osztály.

Attribútumok

-tiles: ConcurrentHashMap<HexTile, Integer>	A nem nulla állapotú mezőket és azok állapotát tárolja.
-ant: HexAnt	A szimulációban szereplő hangya.
-logic: HexAntLogic	A szimuláció szabályait tartalmazza.

Metódusok

+Simulation()	Létrehoz egy szimulációt, alapértelmezett szabályokkal. (lásd HexAntLogic paraméter nélküli ctor)
+Simulation(hm: ConcurrentHashMap<StateKey, HexCommand>)	Létrehoz egy szimulációt a beadott szabályokkal.
+Simulation(hal: HexAntLogic)	Létrehoz egy szimulációt a beadott szabályokkal.
+getAnt(): HexAnt	Visszaadja a hangyát.
+getTiles(): ConcurrentHashMap<Tile, Integer>	Visszaadja a nem nulla állapotú mezőket és állapotaikat.
+getTileState(t: Tile): int	Visszaadja egy adott mező állapotát.
+setTileState(t: Tile, s: int)	Beállítja egy adott mező állapotát.
+getLogic(): HexAntLogic	Visszaadja a szimuláció szabályait.
+setLogic(hal: HexAntLogic)	Beállítja a szimuláció szabályait.
+setLogic(hm: ConcurrentHashMap<StateKey, HexCommand>)	Beállítja a szimuláció szabályait.
+iterate()	Egy iterációval lépteti a szimulációt, a megfelelő szabályok szerint.

2.8 SimWindow

A szimulációt megjelenítő ablak, JFrame-től örököl.

Attribútumok

-sWIDTH: int	Az ablak szélessége.
-sHEIGHT: int	Az ablak magassága.
-ZOOM: double	A nagyítás mértéke.
-maxZOOM: double	A maximális nagyítás.
-minZOOM: double	A minimális nagyítás.
-offsetX: int	A kamera elmozdulásának X koordinátája.
-offsetY: int	A kamera elmozdulásának Y koordinátája.
-colorAntState0: Color	A hangya 0-s állapotához tartozó szín.
-colorAntState1: Color	A hangya 1-es állapotához tartozó szín.
-colorTile: Color	A mezők 1-es állapotához tartozó szín.
-colorGrid: Color	A rács színe.
-colorBGR: Color	A háttér/0-s állapotú mezők színe.
-sim: Simulation	A megjelenítendő szimuláció.

Metódusok

+SimWindow()	Létrehoz egy új szimulációs ablakot.
getWindowWidth(): int	Visszatér az ablak magasságával.
getWindowHeight(): int	Visszatér az ablak szélességével.
getZoom(): double	Visszaadja a nagyítás mértékét.
getOffsetX(): int	Visszaadja a kamera elmozdulását az X tengelyen.
getOffsetY(): int	Visszaadja a kamera elmozdulását az Y tengelyen.
getSim(): Simulation	Visszaadja a megjelenített szimulációt.
setWindowWidth(w: int)	Beállítja az ablak szélességét.
setWindowHeight(h: int)	Beállítja az ablak magasságát.
setZoom(z: double)	Beállítja a nagyítás mértékét.
setOffsetX(x: int)	Beállítja a kamera elmozdulását az X tengelyen.
setOffsetY(y: int)	Beállítja a kamera elmozdulását az Y tengelyen.
setSim(s: Simulation)	Beállítja a megjelenített szimulációt.

2.9 Canvas

A szimuláció kirajzolásáért felelős osztály, SimWindow belső osztálya, Jpanel-től örököl.

Attribútumok

-img: BufferedImage	A kirajzolt kép.

Metódusok

Canvas(f: javax.swing.JFrame)	Létrehoz egy új Canvas elemet a megadott ablakban.
paintComponent(graphics: Graphics)	Kirajzolja a képet az ablakba.
drawAnt(graphics: Graphics, ant: HexAnt, cmd: HexCommand)	Kirajzolja a hangyát a képre.
drawBackground(graphics: Graphics)	Kirajzolja a hátteret a képre.
drawTile(graphics: Graphics, t: Tile)	Kirajzol egy mezőt a képre.
drawGrid(graphics: Graphics)	Kirajzolja a rácsot a képre.
drawSim(graphics: Graphics)	Kirajzolja a szimulációt a képre, az előbbi segédfüggvényeket meghívva.

2.10 MouseListener

Az egér kezeléséért felelős osztály a szimuláció ablakában. Canvas belső osztálya, MouseInputAdapter-től örököl.

Attribútumok

-mousePt: Point	Az egér előző pozíciója, a kamera mozgatásához szükséges.

Metódusok

mousePressed(e: MouseEvent)	Egérkattintásra eltárolja a kurzor jelenlegi pozícióját.
mouseDragged(e: MouseEvent)	A kamera mozgatását kezeli, ha a felhasználó húzza az egeret. A kurzor előző pozíciójához képest eltolja a kamerát.
mouseWheelMoved(e: MouseWheelEvent)	A nagyítást kezeli, ha a felhasználó a görgőt használja.

2.11 ComponentAdapter

Az ablak átméretezéséért felelős, anonim osztály. Canvas belső osztálya.

Attribútumok

-

Metódusok

ComponentResized(e: ComponentEvent)	Az ablak átméretezésekor beállítja a sWIDTH és sHEIGHT változók értékét.

2.12 GUIWindow

A kezelőfelületért felelős ablak, JFrame-től örököl.

Attribútumok

-gWIDTH: int	Az ablak szélessége.
-gHEIGHT: int	Az ablak magassága.

Metódusok

+GUIWindow()	Létrehoz egy kezelőfelület-ablakot.

2.13 GUI

A felhasználói felületért felelős osztály, GUIWindow belső osztálya, Jpanel-től örököl.

Attribútumok

-colorBGR: Color	A háttér színe.
-start: JButton	A Start gomb, elindítja a szimulációt.
-stop: JButton	A Stop gomb, megállítja a szimulációt.
-step: JButton	A Step gomb, egyel lépteti a szimulációt.
-reset: JButton	A Reset gomb, alaphelyzetbe állítja a szimulációt, és frissíti a szabályokat.
-save: JButton	A Save gomb, elmenti a jelenlegi konfigurációt.
-load: JButton	A Load gomb, betölti a kiválasztott konfigurációt.
-speed: JSlider	A sebességet állító csúszka.
-rules: JTable	A szabályokat tartalmazó táblázat.
-presets: JComboBox	A kiválasztható konfigurációkat tartalmazó doboz.
-timer: Timer	A szimuláció időzítője.
-timerrunning: boolean	Az időzítő állapotjelzője. (fut/nem fut)

Metódusok

getTableLogic(): HexAntLogic	Kiolvassa a szabályok táblázatát, és visszatér egy HexAntLogic objektummal.

2.14 ButtonListener

A kezelőfelület gombjaiért felelős osztály, az ActionListener interfészt valósítja meg, GUI belső osztálya.

Attribútumok

-

Metódusok

actionPerformed(e: ActionEvent)	Megnézi, melyik gombot nyomta meg a felhasználó, majd elvégzi a megfelelő műveletet.

2.15 SliderListener

A kezelőfelület sebesség csúszkájáért felelős osztály, a ChangeListener interfészt valósítja meg, GUI belső osztálya.

Attribútumok

-

Metódusok

stateChanged(e: ChangeEvent)	Beállítja a szimuláció sebességét a csúszka állása alapján.

2.16 RulesTableModel

A szabályokat tartalmazó táblázatot valósítja meg, az AbstractTableModel osztálytól örököl, GUI belső osztálya..

Attribútumok

-columnNames: String[]	Az oszlopneveket tartalmazó tömb.
-data: Object[][]	A táblázat adatait tartalmazó kétdimenziós tömb.

Metódusok

+RulesTableModel()	Létrehoz egy táblázatmodell példányt.
+getColumnClass(i: int): Class	Visszatér egy adott oszlop értékének osztályával.
+getColumnCount(): int	Visszatér az oszlopok számával.
+getRowCount(): int	Visszatér a sorok számával.
+getValueAt(r: int, c: int): Object	Visszatér egy adott cella értékével.
+isCellEditable(r: int, c: int): boolean	Igazzal tér vissza minden nem első vagy második oszlopbéli cellára.
+setData(d: Object[][])	Beállítja a táblázat összes értékét.
setValueAt(o: object, r: int, c: int)	Beállítja egy cella értékét.

2.17 XMLParser

XML dokumentumokat feldolgozó osztály, DefaultHandler-től örököl.

Attribútumok

-def: boolean	A beolvasott konfiguráció alapértelmezett-e.
-data: Object[][]	A konfiguráció tartalma a GUI táblázatának megfelelő formában.
-hal: HexAntLogic	A konfiguráció tartalma a Simulation osztálynak megfelelő formában.
-num: int	A konfigurációban található adatsorok (jelenlegi) száma.
-hm: ConcurrentHashMap<StateKey, HexCommand>	A konfiguráció HashMap formában, belső segédstruktúra.

Metódusok

+startElement(uri: String, localName: String, qName: String, attributes: Attributes)	Feldolgozza a jelenlegi XML címkét.
+endDocument()	Véglegesíti a beolvasott konfigurációt.
+getDefault(): boolean	Visszatér def értékével.
+getData(): Object[][]	Visszatér data értékével.
+getLogic(): HexAntLogic	Visszatér hal értékével.

2.18 FileManager

A fájlkezelésért felelős osztály. Minden metódusa statikus, és nem példányosítható.

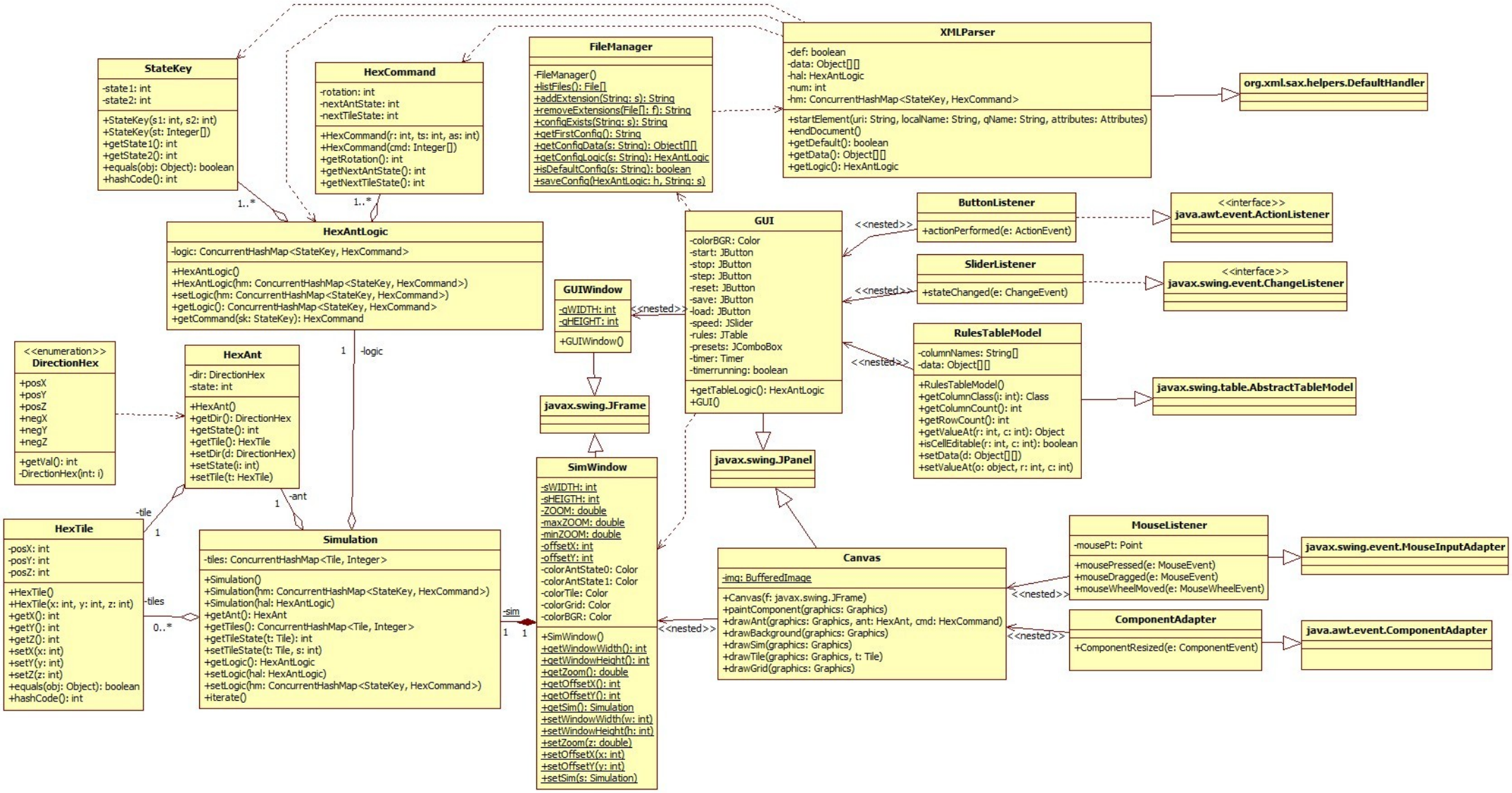
Attribútumok

-

Metódusok

-FileManager()	Privát konstruktor, hogy ne lehessen példányosítani.
+listFiles(): File[]	Kilistázza az összes .xml kiterjesztésű fájlt a gyökérkönyvtárban.
+addExtension(String: s): String	Hozzáadja a .xml kiterjesztés a beadott Stringhez.
+removeExtensions(File[]: f): String	Eltávolítja az .xml kiterjesztést a beadott tömb elemeiből.
+configExists(String: s): String	Megnézi, hogy létezik-e az adott konfiguráció.
+getFirstConfig(): String	Visszatér az első konfiguráció nevével.
+getConfigData(s: String): Object[][]	Visszatér a beadott konfigurációval, a GUI táblázatának megfelelő formában.
+getConfigLogic(s: String): HexAntLogic	Visszatér a beadott konfigurációval, a Simulation szabályának megfelelő formában.
+isDefaultConfig(s: String): boolean	Igazzal tér vissza, ha a beadott konfiguráció alapértelmezett.
+saveConfig(HexAntLogic: h, String: s)	Lementi a beadott konfigurációt beadott néven.

2. AZ OSZTÁLYDIAGRAM:



4. Tesztelés menete:

A tesztelést JUnit4 segítségével hajtottam végre, összesen 4 osztály 12 metódusa lett tesztelve. (ctor és getter/setter függvényeket nem számolva)

4.1 HexAntTest

A HexAnt osztály alapértelmezett konstruktorának és getterjeinek tesztje.

4.2 HexTileTest

A HexTile osztály getterjeinek, setterjeinek, az equals és hashcode függvényeinek tesztje.

4.3 SimulationTest

A Simulation osztály alapértelmezett konstruktorának, és egy iterációjának (iterate függvény) tesztje, több paraméterrel.

4.4 FileManagerTest

A FileManager osztály függvényeinek tesztjei. A két felugró „File not found” párbeszédablak a rendes működés része.

5. FELHASZNÁLÓI KÉZIKÖNYV:

A program két ablakban indul el, az egyik a szimulációs ablak, a másik a kezelőfelület, ahol a felhasználó a szimuláció szabályait állíthatja be, míg a szimulációs képernyőn az egérgombot letartva mozgathatja a kamerát, illetve az egér görgője segítségével nagyíthatja, vagy kicsinyítheti a képet. A programot bármelyik ablak bezárásával lehet leállítani.

A szimuláció alapértelmezetten egy üres játéktéren indul, a hangyával középen.

A kezelőpanelen a Start gomb elindítja a szimulációt a táblázatban szereplő szabályok szerint, a csúszkán kiválasztott sebességgel, vagy ha már fut, akkor nem csinál semmit.

A Stop gomb megállítja a szimulációt, illetve ha már áll, akkor nem csinál semmit.

A Step gomb egyel lépteti a szimulációt, ezt a felhasználó akár futás közben is megnyomhatja.

A Reset gomb alaphelyzetbe állítja a szimulációt, a kamerát a középpontba állítja és a nagyítás mértékét is alapértékre teszi. A gomb továbbá frissíti a szimuláció szabályait is, ha változás történt a legutóbbi Start gombnyomás óta. Ha a szimuláció fut, akkor a gomb megnyomása után megint elindul, ha nem, akkor a Start gomb nyomásáig nem indul el.

A Save gombbal lehet elmenteni a jelenlegi szabályokat, a táblázat feletti Presets doboz szerinti néven. Ha alapértelmezett konfigurációt akarnánk felülírni, akkor a program felugró párbeszédablakban ezt jelzi, és arra kéri a felhasználót, hogy másik nevet adjon meg a mentésnek. A Presets szövegdobozba gépelheti a felhasználó a mentés kívánt nevét. Hogyha érvénytelen (üres) nevet akar megadni a felhasználó, akkor a program ezt szintén párbeszédablakban jelzi. Amennyiben már létezik ilyen mentés, és nem alapértelmezett, akkor az felülírásra kerül.

A Load gombbal be lehet tölteni a Presets legördülő menüből kiválasztott, illetve begépelt konfigurációt. Amennyiben nem létező konfigurációt akar betölteni a felhasználó, a program egy felugró párbeszédpanelen jelzi ezt. Hogyha sikeres a betöltés, akkor a szabályok táblázat frissül az új értékekkel, és a szimuláció alapértelmezett helyzetbe kerül. Hogyha futott a szimuláció, akkor nem indul újra az esetben.

A Presets menüpont alatt induláskor megjelenik az összes már mentett konfiguráció. Ha újat hozunk létre, akkor az hozzáadódik a listához.

Legalul található a szabályok táblázata, ahol a felhasználó begépelheti a kívánt adatokat. Helytelen adat esetén a program nem engedi a felhasználónak jóváhagyni a módosítást addig, ameddig helyes adatot nem ad meg. Ha a felhasználó befejezte a tábla megírását, akkor a Reset gombbal beállíthatja az új szabályt a szimulációban. Az első két oszlopot nem lehet módosítani, mert ezek a hangya és mező lehetséges állapotai, az ezekhez tartozó parancsokat adhatja meg a felhasználó.

A szabályok táblázat oszlopneveinek jelentése:

- TST: Jelenlegi mező állapota (**T**ile **St**ate
- AST: Hangya jelenlegi állapota (**A**nt **St**ate)
- ROT: A hangya elfordulásának mértéke, ennyiszer 60°. (**Rot**ation)
- TNS: A mező következő állapota (**T**ile **N**ext **S**tate)
- ANS: A hangya következő állapota (**A**nt **N**ext **S**tate)

A szimuláció sebességét a Save/Load gombok alatt található csúszka segítségével lehet állítani.

A program induláskor betölti név szerinti sorrendben az első konfigurációt, vagy ha nincsen egy sem, akkor egy felugró párbeszédablakban értesíti erről a felhasználót.

Felhasznált források:

- koordináta-rendszer (Cube coordinates):

https://math.stackexchange.com/questions/2254655/hexagon-grid-coordinate-system/2643016#2643016

- hatszögrács kirajzolás (csak az ötlet, megvalósításban végül mást használtam):

https://stackoverflow.com/questions/20734438/algorithm-to-generate-a-hexagonal-grid-with-coordinate-system