A programozás alapjai 3-Nagy házi feladat dokumentáció

1. Módosított specifikáció

Választásom a Pacman nevű arcade játék megvalósítása Javaban.

Felhasználói specifikáció:

A 2D-s játék lényege, hogy az ikonikus, sárga figurával minél több tallért „együnk meg”. Tudni illik Pacman animációja szerint folyamatosan tátog és próbálja elkapni, megenni az összes arany tallért. Ezeket Inky Blinky Pinky és Clyde őrzik. Három élettel rendelkezik a játékos. A játékos Pacmant fogja irányítani a nyilak segítségével. A legutoljára leütött nyíl irányába fog mozogni Pacman, amíg más parancsot nem kap és tud abba az irányba menni. A játék egy (előreláthatólag) 940x930-as ablakban játszódik.

Programozói specifikáció:

Előreláthatólag Swinget fogok használni grafikus megjelenítőként. Mindenek elött megpróbálom blokkokra osztani a játék részét. Alapvetően 3 megjelenítendő objektumot tervezek:

1. Block: Talán egyedül ez a pont igényel különösebb magyarázatot. Lehet hogy ez egy olyan mező ami nem elérhető a játékosok számára(mondhatni ”fal”). Ezeket egyértelműen megkülönböztethető színekkel fogom ábrázolni. A második típusú field pedig egy játéktér. Azt ezt magábafoglaló/megvalósító osztálynak bool változókkal lesz megkülönböztetve, hogy tallér, szupertallér vagy üres mezőről van szó.(Vagy akár enummal.)

2. Pacman: Pacmant fogja meganimálni. Képek frissítésével fogja mozgatni az objektumot. Elképzeléseim szerint egy tömbböl fogja kiolvasni a képeket és mindig ezen a tömbön iterál végig.

3. Monster: Nagyon hasonlóan mint a Pacman objektumot. Időm engedtével megpróbálom megoldani, hogy küönböző irányú mozgásokra különböző képeket animáljon a játék.

Nem úgy mozog Pacman, hogy amíg nyomva tartják a jobbra nyilat és tud menni addíg jobbra megy, hanem amíg nem nyomnak mást és tud menni. Magyarul: elég egy billentyű bekérése egy időben. Megjegyzem, hogy szerintem élvezhetőbb lenne a játék az első mozgási mehanizmussal, de hagyományőrzés szempontjából a másodikat implementálom.Kell egy Pacmant kirajzoló, animáló, mozgató szál, a 4 szellemnek ugyan ezt a szálat fogom használni. Előre láthatólag az útválaszt algoritmusok is a kirajzolás közben fognak futni. Fájlkezelés: Ha már tanultuk és szerintem meglehetősen hasznos a szerializálást fogom használni. Rekord listát tervezek csinálni, ezt mindenképp. Időm engedtével pedig az aktuális játékállást tervezem még elmenteni. További megjegyzések: Minden szörnynek különböző algoritmust tervezek. Lesz egy teljesen determinisztikus egy olyan ami kopó módjára követi majd Pacmant, egy olyan ami Pacman elé próbál majd kerülni. Ezeket időm engedtével megvalósítom.

A felhasznlót egy loading screen fogadja majd és enterre új játékot kezd.

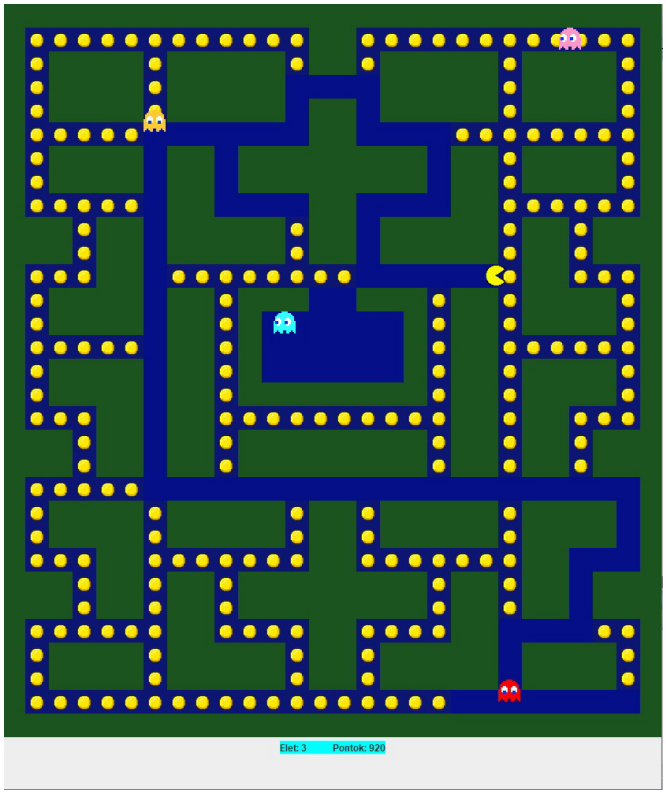
Somogyi Bence

Q79IBL

2019. október 30.

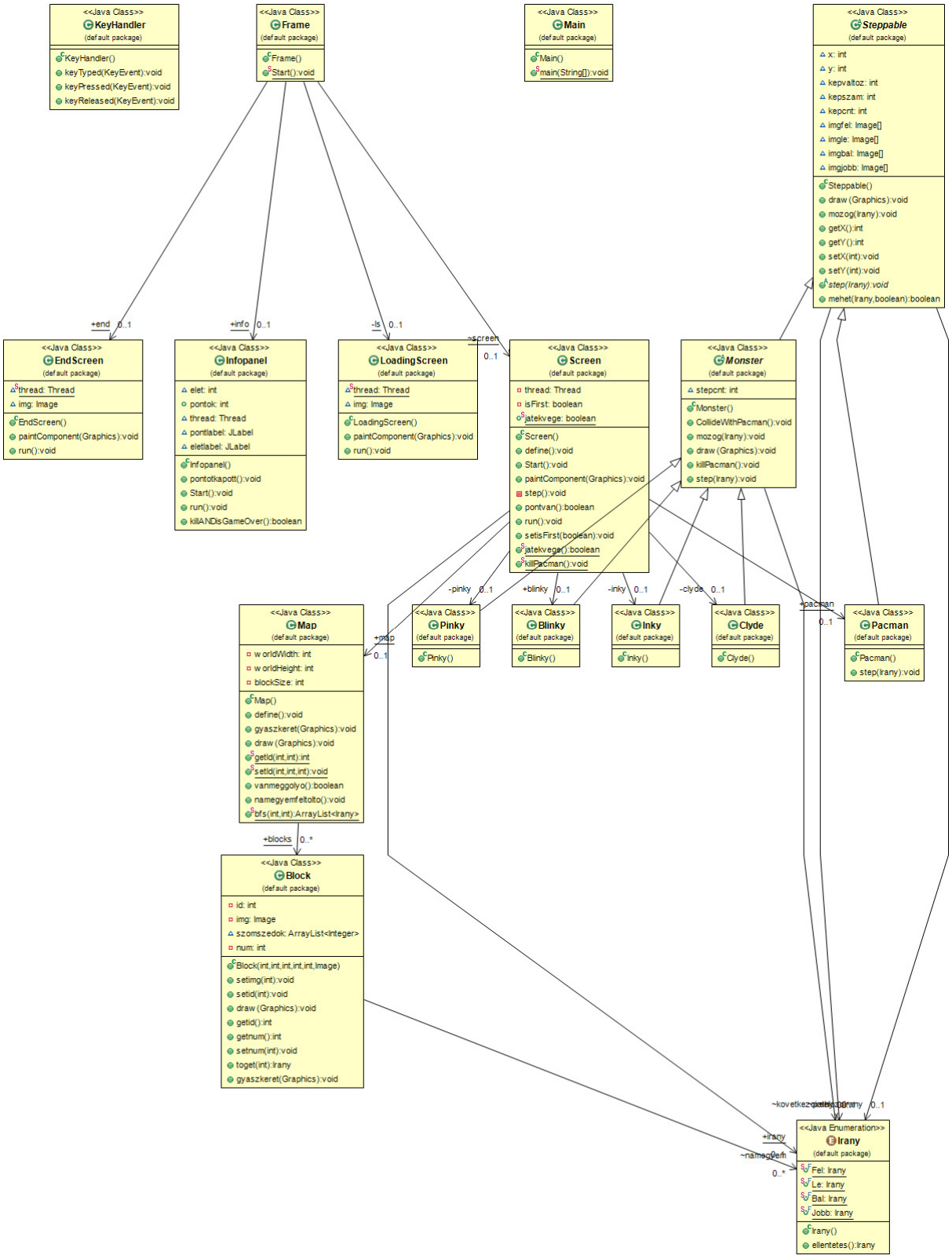
2. Felhasználói dokumentáció:

A Pacman nevű Arcade játékban egy sárga figurát kell irányítani a nyilakkal és minél több tallért összeszedni. Amikor a játékos megnyom egy gombot, akkor abba az irányba fog mozogni A pályán vannak falak (zöld) és mezők (kék). A szellemek és Pacman csak a mezőkön mozoghat. Lehet hogy ezeken van arany tallér, lehet hogy nincs. A nyilakkal irányítva az a játékos dolga, hogy összeszedjen minél több tallért. Ha a pályán minden tallért összegyűjtött akkor újratöltődik a pálya. Pacmannek három élete van, ha egy szellek elkapja akkor elveszít egy életet. A játék addig megy amíg nem vesztette el mind három életét. Ezeket az információkat a képernyő alján lehet látni, hogy hány élete van és hány pontja van a játékosnak. Ha a pályán minden tallért összeszedett a játékos akkor automatikusan újra generálódik a pálya. Játék közben a játékos escape-et nyoymva kilép a játékból.



3. Programozói dokumentáció

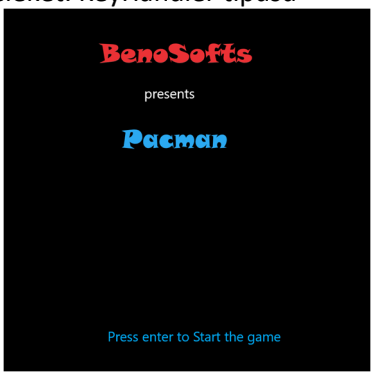
Az osztálydiagramm:



Ez egy generált osztálydiagramm. Az Eclipseben letölthető *objectaid* bővítményt használtam.

Frame class:

A JFrame leszármazottja, a megjelenítéshez használom. Négy fontos adattagja a LoadingScreen, Screen, Infopanel, EndScreen. Ezekre később kitérek. Rendelkezik egy konstruktorral és egy Start metódussal, ami elindítja a játékot és láthatóvá teszi a megfelelő Jpaneleket. KeyHandler típusú KeyListener van hozzáadva.



LoadingScreen class:

Az osztály kirajzolja a legelső képet és egy enter leütést vár amire elindul a játék. A JPanel leszármazottja és implementálja a Runnable interfacet.

Screen class:

Ez az osztály felelős az egész játékmenetért, mondhatni kontroller és view funkciókat lát el. Implementálja a Runnable interfacet.

Fontosabb adattagjai a Thread, ami a játék szálát kezeli. A játékége bool változó. Az isFirst bool változó. Map típusú adattagja tárol minden a pályáról. Illetve maga a játékban szereplő karakterek: Pacman, Inky, Blinky, Pinky és Clyde. Rendelkezik egy konstruktorral ami csak a méreteket álítja be. Define metódusban inicializálom az adattagokat, ez két helyen is kell. Mind a konstruktorban, mind akkor amikor hamarabb indult volna el a szál mint lefutott volna a konstruktor. Start metódus elindítja ezt a szálat és a Frame Infopanel típusú szálát, mert ez a kettő csak együtt futhat a játék hibátlan futásában. paintComponent metódusa meghívja minden mező, minden szellem és Pacman draw függvényét, itt rajzolódnak ki a játékelemek. Step metódusában mindig léptet egyet midnen szellemen és Pacmanen és meghívja a Pontvan metódust. Pontvan metódusa figyeli azt ha Pacman felvett egy tallért, ekkor meghívja az Infopanel megfelelő metódusát és az adott blokk idját és képét megváltoztatja. A run metódus maga az ún. game loop. Ez ugye a Runnable interface run metódusának felülírása. Addig fut amíg nem hallt meg pacman háromszor (!jatekvege) Az első körben inicializál, majd utána megnézi hogy nem kell e frissíteni a térképet (ha kell, újra feltölti tallérokkal) és minden kkörben meghívja a már említett step és paintComponent metódusokat. Killpacman metódusában a kezdőpozíciójába helyezi Pacmant, meghívja az Infopanel killANDisGameOver metódusát ami vissza ad egy bool értéket miszerint van- e még élete Pacmannek vagy már elfogyott mind a három. Ha vége a játéknak leállítja a két szálat és a Screen panelt és az Infopanel panelt láthatatlanná teszi. Ellenben az EndScreent láthatóvá teszi.

Infopanel class:

Az osztály felelőssége a pontok és az életek számának nyilvántartása, változásának menedzselése és monitorozása. A JPanel osztály leszármazottja és implementálja a Runnable interfacet.

Attribútumai az életeket és pontokat számoló integer, a pontokat és az életek számát monitorozó label-ek illetve a Thread típusú változó. Az osztály konstruktora nem igényel különösebb magyarázatot. A pontotkapott metódust hívja meg a Screen osztály, hozzáad 10-et a pontszámhoz. Start metódusa elindítja a szálat. Run metódusa monitorozza a két Jlabel értékét. KillANDisGameOver metódusa csökkenti eggyel az életek számát és visszaadja hogy vége-e a játéknak vagy nem.

EndScreen class:

Nagyon hasonlóan a LoadingScreen osztályhoz, egy darab képet rajzol ki az után hogy a játékos kiesett. A JPanel leszármazottja és a Runnable interfacet implementálja. Bármely billentyű leütését várja, ezután az alkalmazás leáll.

Block class:

Az osztály felelős azért, hogy milyen típusú mező van az adott pozícióban és hogy a megfelelő képet rajzolja ki mint mező. A Rectangle osztály leszármazottja, mert a setbounds metódusával nagyon egyszerű beállítani a blokkok méretét, rengeteget egyszerűsödik a kód.

Attribútumai az id integer az img Image amik használhatóak. A többi attribútum a BFS algoritmus működéséhez kell (fejlesztés alatt).

A gettereken és settereken kívül a draw metódust érdemes talán megmlíteni, kirajzolja az eltárolt képet a megelelő pozícióba.

Map class:

Az osztály felelős a pálya blokkjainak struktúrájáért azok szinkronbantartásáért a játékkal.

Fontosabb adattagjai: Block[][]blocks mátrix ami a pálya minden blokkját veszi kollekcióba.

define metódusa az ami tartalmazza a filekezelését a szoftvernek. Beolvassa a res/Map/Map.txt tartalmát és feldarabolja spacenként. Kinyeri a mátrix megfelelő elemeinek idját és az alapján álítja be a képet. (Fal arany vagy üres.) A pozíció beállítása triviális: x=30*belsőciklusfutóindex; y=30*külsőciklusfutóindex. Vannak továbbá getterek setterek. draw metódus meghívja minden blokk draw metódusát, mondhatni kiadja a parancsot minden blokknak hogy rajzolja ki magát. vanmeggolyo metódus visszaadja hogy van e még olyan mező ami van arany, ha false értéket kaptunk vissza frissíteni kell a pályát, lefut a define metódus. Ez egyébként a Screen osztályban történik.

Steppable class:

Egy absztrakt osztály ami összegyűjti a mozgatható objektumok közös tulajdonságait.
Leszármazottjai: Monster class és Pacman class.

Attribútumai: x és y pozíciót tároló integer, Irany „típusú” kovetkezoirany és elozoirany változók, az animáláshoz szükséges integer változók illetve négy darab Image tömb. Az Irany változók a mozgatás kivitelezéséhez szükségesek a tömbök pedig ahoz hogy különböző képet tudjon megjeleníteni a draw metódus annak megfelelően hogy melyik irányba mozog.

draw metódusa megvalósítja az animációt és kirajzolja a megfelelő képet. Animáció menete: mindenekelött kiválasztja hogy melyik irányba mozog az objektum és kirajzolja a megfelelő tömb képcnt-edik elemét. kepvaltoz egy futóindex, minden step-ben növeljük az értékét. kepcnt-nek vesszük a kepszam-mal(tömbök egységes hossza) való osztási maradékát(így biztosan nem kapunk túlindexelés exception-t).Ha a kepvaltoz elérte a hatot akkor kinullázzuk, magyarul, hat pixelenként rajzol ki más képet az algoritmus.

mozog metódusa egy irányt vár paraméterül, a megfelelő irányba mozgatja el az objektumot 1 pixellel.

Tartalmazza az abstract step metódust ami egy irányt vár paraméterül. Ebben lesznek definiálva a mozgatható objektumok mozgási logikái.

Fontos a mehet(Irany i, bool osztvaharminc) metódus. Ez adja vissza egy mozgatható objektumnak, hogy mehet e az adott irányba akkor ha pont illeszkedik egy blokkra és ha pont nem. Azért kell különvenni mert csak akkor tud fordulni ha pontosan illeszkedik egy blokkra. Miután eldöntötte a etódus hogy pontosan illeszkedik e az objektum a blokkra megvizsgálja a megfelelő irányba következő blokkról hogy fal vagy nem. Bool értéket ad vissza a metódus.

Monster class:

A Steppable osztály leszármazottja, leszármazottjai a **Clyde, Inky, Blinky és Pinky osztályok** .

draw és mozog metódusát egy az egyben örökli ősétől. CollidewithPacman metódusa ellenőrzi a Pacmannel való ütközést minden egyes mozgás után. Egy nagyon egyszerű aritmetikai összehasonlítást végez: |Monseter.x-Pacman.x |<30&&| Monseter.y-Pacman.y |<30 Ha ez igaz akkor meghívja a killpacman metódust. Ez pedíg jelez a Screen osztálynak, hogy meghalt pacman.

Step metódus tartalmazza a random mozgásos algoritmust. Minden mozgás elött megvizsgálja milyen irányokba mehet, ebből csinál egy Irany típusú ArrayListet. Ha egy irányba mehet akkor triviális, ha kettőbe akkor megy tovább abba az irányba amiből jött.(Itt veszem hasznát az Irany enum ellentetes() metódusának.) Ha három vagy több irányba mehet akkor sorsol egyet az arraylistből és arra megy. Fontos: iránytváltani csak akkor tud amikor pontosan illeszkedik a blokkra.

A különböző szörnyeket sajnos felesleges dokumentálni mert mindegyik ezt a mozgási algoritmust használja és annyiban különböznek hogy más képekkel van feltöltve a tömb.

Pacman class:

A Steppable osztály leszármazottja. A játékos ezt a figurát irányítja.

Egyedül a Step metódus szorul magyarázatra. Paraméterként kap egy irányt.

Ha pontosan illeszkedik egy blokkra: Menézi hogy tud e a paraméterként kapott irányba menni. Ha tud akkor beállítja a kovetkezoirany értékét erre az irányra, ha nem tud menni akkor elozoiranyba mozog ha mehet.(mehet= mehet metódus meghívása).

Ha nem illeszkedik pontosan akkor csak megy az előző irányba hisz fordulni ott ugyse tud (két blokk közt).

KeyHandler class:

Implementálja a KeyListener interfacet. A Frame osztályhoz van hozzáadva.

Egyedül a keypressed metódus szorult felüldefiniálásra. Figyeli azt amikor entert kapott és nincs vége a játéknak: ekkor a loadingscreenről a játékba lép az alkalmazás fázisa. Amennyiben a nyilakat kapott a program, akkor átálítja a Screen osztály irany metódusát a megfelelőre. Továbbá ha bármely billentyűt megnyomott a felhasználó és vége a játéknak kilép az alkalmazásból.

Somogyi Bence

Q79IBL

2019.12.02.