Programozás technológia

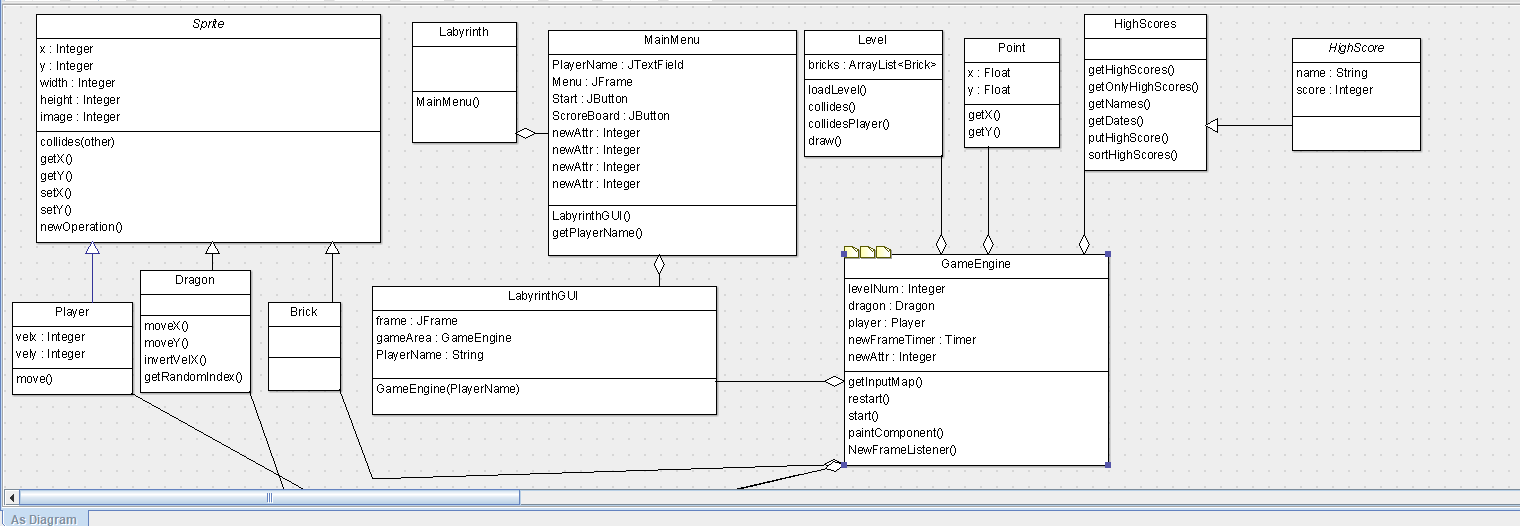
3.beadandó feladat: 3. feladat

Készítette: Madár Bálint MRPBLG

# A feladat leírása:

Készítsünk programot, amellyel egy labirintusból való kijutást játszhatunk. A játékos a labirintus bal alsó sarkában kezd, és a feladata, hogy minél előbb eljusson a jobb felső sarokba úgy, hogy négy irányba (balra, jobbra, fel, vagy le) mozoghat, és elkerüli a labirintus sárkányát. Minden labirintusban van több kijutási útvonal. A sárkány egy véletlenszerű kezdőpozícióból indulva folyamatosan bolyong a pályán úgy, hogy elindul valamilyen irányba, és ha falnak ütközik, akkor elfordul egy véletlenszerűen kiválasztott másik irányba. Ha a sárkány a játékossal szomszédos területre jut, akkor a játékos meghal. Mivel azonban a labirintusban sötét van, a játékos mindig csak 3 sugarú körben látja a labirintus felépítését, távolabb nem. Tartsuk számon, hogy a játékos mennyi labirintuson keresztül jutott túl és amennyiben elveszti az életét, mentsük el az adatbázisba az eredményét. Egy menüpontban legyen lehetőségünk a 10 legjobb eredménnyel rendelkező játékost megtekinteni, az elért pontszámukkal, továbbá lehessen bármikor új játékot indítani egy másik menüből. Ügyeljünk arra, hogy a játékos, vagy a sárkány ne falon kezdjenek. A megoldás terve, az osztályokat és kapcsolataikat bemutató osztálydiagram, a feladat elemzése

UML diagram:



# A program felépítése és elemzése és a megoldáshoz szükséges típusok (osztályok) absztrakt leírása

labyrinth feladat:

**Brick.java a** „kövek” leírását megadó osztály

**Dragon.java** a sárkányt és a random mozgását leíró osztály

**GameEngine.java**, a játék fő motorja itt található, illetve a főbb esemény kezelők

**HighScore.java** A HighScores-ben lesz megvalósítva az adatbázis értékeit tartalmazza

**HighScores.java** az adatbázis lekérdezéséhez módosításához szükséges osztály

**Labyrinth.java** főmetódus itt fut le

**LabyrinthGUI.java** a labirintust hozza létre

**Level.java** a szintek betöltéséért felelős

**MainMenu.java** : a program alapablaka, innen lehet új játékot indítani

**Player.java** a játékos osztály ami a játékos mozgását határozza meg

**Point.java** egy pont osztály

**Sprite.java ebből** az osztályból számaznak az alakzatok

**Ikonok a programnak:** icon.jpg, player.png, dragon.jpg

# Esemény-eseménykezelő párosítások és a tevékenységük rövid leírása

Új játék gomb kattintás: new LabyrinthGUI(PlayerName.getText()); függvény fut le, ekkor létrejön az új játék.

A WASD lenyomás során a lent felsorolt esemény kezelők futnak le.

**Esemény**: this.getInputMap().put(KeyStroke.getKeyStroke("A"), "pressed left");

**Eseménykezelő**: pressed left: Bal oldali irányba mozgatja a játékost

**Esemény**: this.getInputMap().put(KeyStroke.getKeyStroke("D"), "pressed right");

**Eseménykezelő:** pressed right: jobb oldali irányba mozgatja a játékost

**Esemény**: pressed up, pressed down

**Eseménykezelő:** A játékost fel illetve le irányba mozgatja

**Esemény**: this.getInputMap().put(KeyStroke.getKeyStroke(KeyEvent.VK\_A, 0, true), "leftreleased");

**Eseménykezelők**: leftreleased, releasedright, releasedup, realeseddown megállítják a játékos mozgását ha az adott gombot elengedi a felhasználó

**Fontos függvények amelyek lefutnak a játék működése közben:**

* **dragon.moveX();** A sárkány X irányú mozgatásáért felelős függvény.
* **dragon.moveY();** A sárkány Y irányú mozgatásáért felelős függvény.
* **dragon.invertVelX();** A sárkány mozgását invertálja az X tengelyen
* **dragon.invertVelY();** A sárkány mozgását invertálja az Y tengelyen
* **restart();** Újraindítja az adott szintet
* **player.collides(dragon)** Ha a játékos a sárkánnyal ütközik vége van a játéknak és adatbázisba lesz elmentve a pontszám.
* **level.collidesPlayer(player**) Fontos függvény abból a szempontból, hogy a játékos ne tudjon rámenni a téglára.
* **isOver()** Ha a felhasználó megoldja az adott szintet jön a következő szint

# Lényeges importált Java osztályok:

import java.awt

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JPanel;

# Részletes tesztelési terv

Több féle játék szimulálása és lejátszása.