SPŠE Ječná

Informační technologie

Ječná 30, 122 00

Poopie Mafrun

Shit or death

Erik Vaněk

2022

Obsah

1 Cíl práce

Za cíl jsem si dal vytvoření bludišťové hry, kde by se hráč pohyboval šipkami skrz neprobádané území, s tématem „dojít si na toaletu“ pomocí programovacího jazyku Java.

2 Software

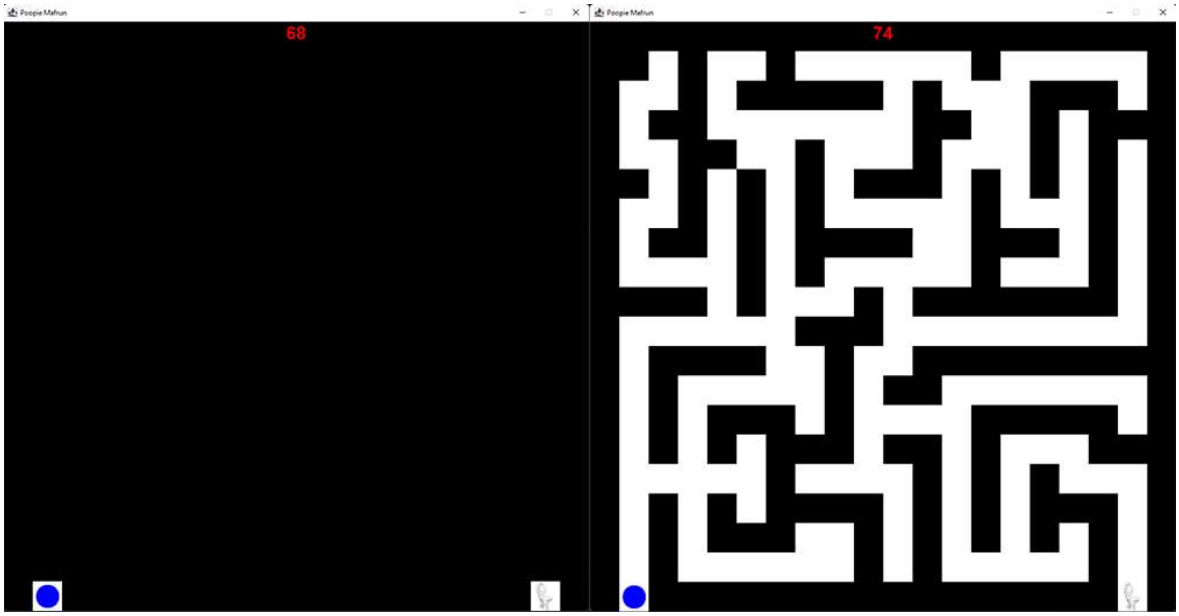
- IntelliJ IDEA 2021.3.2
- Java 16
- Malování - návrh prototypů levelů

3 Program

Ve hře Poopie Mafrun je hráč postaven před herní pole 20x20 kostek, ve kterém se nachází zahalené bludiště, jeho úkolem je pohybovat se skrz bludiště šipkami tak aby došel od startu do konce do 75 sekund. Cesty a stěny jsou zahaleny, není tak možné vidět na první pohled správnou cestu, je tak třeba prozkoumávat možné odbočky a směry.

Lore: Andrew Manký za kterého hrajeme, z ničeho nic akutně potřebuje na toaletu s velkou potřebou, jelikož si neuvědomil, že požívat KFC Chicken Wings™ a jahodovo-mléčný koktejl dohromady, je špatný nápad. Snaží se tak v budově Matematicko-fyzikální fakulty Univerzity Karlovy najít záchody.

Základní kostru programu jsem sestavil za pomocí YouTube tutorialu od kanálu BroCode na snake hru.[1] Odtuď jsem se odrazil a postupně se propracovával k finální verzi mé hry.



Třída Main vytvoří instanci třídy GameFrame.

Třída GameFrame se rozšiřuje o JFrame, vytvoří se tak okno. Ve svém konstruktoru nastavuje základní vlastnosti okna a vytváří instanci třídy GamePanel.

Třída GamePanel se rozšiřuje o JPanel a implementuje ActionListener. Drží také několik následujících důležitých proměnných:

- Integer
  - SCR_WIDTH – (static) šířka okna
  - SCR_HEIGHT – (static) výška okna
  - UNIT – (static) velikost jednotlivých kostek v okně
  - playerX – pozice hráče na ose X
  - playerY – pozice hráče na ose Y
  - levelSelection – zvolený level
  - countdown – odpočet
- Boolean
  - alive – zajišťuje aby hráč prohrál po vypršení času
  - win – kontroluje pokud se hráč dostal do cíle
- Enum
  - STATE – 4 stavy: MAINMENU, LEVELMENU, GAME, GAMEOVER
  - gameState – drží informaci v jakém stavu se hra právě nachází

Do konstruktoru přidává KeyListener a MouseListener ze třídy GameInput.
K tomu navíc vytváří instance tříd GameRender a Countdown pro přenos dat.

Metody:

- gameStart(); - start hry: zavolá metodu readLevels();, nastaví startovní: pozici hráče, odpočet, booleany a timer.
- paintComponent(); – na základě *gameState* si program volí co se bude v okně renderovat
- actionPerformed(); - při každé interakci hráčem či timeru spustí repaint(); pro obnovu renderu, zároveň sleduje jestli hráč došel do cíle

```
//region render
public void paintComponent(Graphics g) { //chooses what to render based on enum state
    super.paintComponent(g);

    switch (gameState) {
        case MAINMENU -> gR.renderMainMenu(g);
        case LEVELMENU -> gR.renderLevelSelection(g);
        case GAME -> {
            if (alive) gR.renderGame(g);
            else gameState = STATE.GAMEOVER;
        }
        case GAMEOVER -> gR.renderGameOver(g);
    }
}
//endregion
```

Třída GameLevel slouží k přečtení rozložení kostek levelu ze souboru .csv.

Metody:

- getMapLayout(); vrátí arraylist arraylistů *levels*
- readLevels(); díky arrayi *levelPath* přečte všechny soubory a vloží je jako arraylist integerů do arraylistu *levels* (použil jsem nabyté znalosti z předchozího úkolu z IT o načítaní souborů)
- blockCoords(int x, int y); po zavolání a vložení hodnot vrátí pořadové číslo dané kostky ve svém arraylistu
- delete(); smaže všechny načtené levely (používáno pro „try again" - rozhodně existuje lepší řešení, tohle mi přišlo (psychicky) úspornější)

Obsahuje statický arraylist arraylistů levels, a statický array levelPath do kterého se zadávají jednotlivé umístění souborů s levely.

```
//read map layout from files
public static void readLevels() {
    delete();
    for (int i = 0; i < levelPath.length; i++) {
        ArrayList<Integer> tmp = new ArrayList<~>();
        try {
            BufferedReader br = new BufferedReader(new FileReader(levelPath[i]));
            String line;
            while ((line = br.readLine()) != null) {
                var input :String[] = line.split( regex: ",");

                for (int j = 0; j < input.length; j++) {
                    tmp.add(Integer.parseInt(input[j]));
                }
            }
            levels.add(tmp);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

//get "block" order in arraylist
public static int blockCoords(int x, int y) {
    int pos = 0;

    for (int i = 1; i < (y / 50) + 1; i++) {
        pos += 20;
    }
    pos += (x / 50);

    return pos;
}
```

Třída GameRender má za účel renderovat jednotlivé části hry podle *gameState*.

Metody:

- renderGame(); renderuje samotný level vyvoláním metody *renderLevel();*, hráčovi objevené kostky na bílou, pohyb hráče a odpočtu.
- rednerLevel(); renderuje level z arraylistu *levels* podobně jako staré CRT monitory, řádek po řádku (pomocí forcyklů), vše krom objevených cest přebarvuje na černo.
- renderGameOver(); pokud hráč došel do cíle vyrenderuje výherní text nebo opak, také renderuje možnosti kterými si uživatel vybere co bude dál dělat: Menu, Try again, Next level.
- renderMainMenu(); renderuje hlavní menu s jeho možnostmi, a má zajímavé pozadí.[2][3]
- renderLevelMenu(); renderuje menu s možností výběru levelu.[2][3]

```
public void renderLevel(Graphics g) { //render of map layout
    int x = 0;
    int y = 0;
    int block = 0;
    for (int i = 0; i < SCR_HEIGHT / UNIT; i++) {
        for (int j = 0; j < SCR_WIDTH / UNIT; j++) {
            g.setColor(Color.BLACK);
            if (getMapLayout().get(gP.getLevelSelection()).get(block).equals(0) || getMapLayout().get(gP.getLevelSelection()).get(block).equals(1))
                g.fillRect( x: x * UNIT,  y: y * UNIT, UNIT, UNIT);
            else {
                g.setColor(Color.WHITE);
                g.fillRect( x: x * 50,  y: y * 50, UNIT, UNIT);
            }
            x++;
            block++;
        }
        y++;
        x = 0;
    }
    g.setColor(Color.white);
    g.fillRect( x: 1 * UNIT,  y: 19 * UNIT, UNIT, UNIT); //start
    g.drawImage(Toolkit.getDefaultToolkit().getImage( filename: "gamefiles/images/toilet.png"),  x: 18 * UNIT,  y: 19 * UNIT,  observer: null); //finish
}
```

Třída GameInput registruje input od uživatele a aplikuje jej na základě herního stavu *gameState*.

Obsahuje konstruktory aby mohl využívat dat z ostatních tříd, přesněji z: GamePanel, GameLevel a Countdown.

Metody:

- mouseClicked(); na základě kam přesně hráč klikne spustí určitou metodu či nastaví proměnou. [4]
- keyPressed(); podle šipek mění pozici hráče na osách X a Y, taky sleduje pokud je kostka jiné hodnoty než nula aby no ní mohl hráč vstoupit.

Třída Countdown má pouze jeden účel, a to se starat o odpočet který hráč vidí nad mapou. Použil jsem Timer a actionPerformed. Timer má nastavený delay 1000ms a actionPerformed zajišťuje že se odečte 1 z integeru countdown.[5]

Obsahuje konstruktor pro získání informací z GamePanel třídy a vytváří instanci třídy GameInput pomocí konstruktoru.

```
public Timer countdownTimer = new Timer( delay: 1000,  listener: this);

@Override
public void actionPerformed(ActionEvent e) {
    gP.setCountdown(gP.getCountdown() - 1);
    if (gP.getCountdown() <= 0) gameState = GamePanel.STATE.GAMEOVER;
}
```

4 Závěr

Mým původním záměrem bylo udělat hru, kde se místo hledání cest pohybováním, bude hledat cesta odpovídáním na matematické rovnice, zda jsou pravdivé či ne (např. pokud se 6+9 = 42 pokud ne tím směrem nelze jít), od toho jsem rychle opustil, když jsem zjistil, že by to bylo těžší udělat. 😊

Na pár problémů jsem narazil, některé jsem vyřešil rychle, ale jiné vyžadovali trochu více pozornosti. Jedním z větší zádrhelů pro mě byl, když jsem zjistil že není úplně dobré mít téměř všechny proměnné statické, problém jsem vyřešil zhruba do dne, když mi kamarád poradil nějaké možnosti, jak problém vyřešit.

Z celkové práce jsem jinak spokojený, dělá to to, co to má dělat, a to je hlavní. Práce probíhala bez problémů, obzvlášť když jsem u toho poslouchal nějaké pořádné „bengry“.

5 Zdroje

[1] Java snake game . In: *YouTube* [online]. 20.6.2020 [cit. 2022-06-02]. Dostupné z: https://youtu.be/bI6e6qjJ8JQ

[2] Java Game Development #19 - Starting Menu System. YouTube [online]. 8.7.2013 [cit. 2022-06-02]. Dostupné z: https://youtu.be/FZWX5WoGW00

[3] Simplest way to set image as JPanel background. Stackoverflow [online]. 1.10.2013 [cit. 2022-06-02]. Dostupné z: https://stackoverflow.com/questions/19125707/simplest-way-to-set-image-as-jpanel-background

[4] Java Game Development #20 - Mouse Input. YouTube [online]. 26.7.2013 [cit. 2022-06-02]. Dostupné z: https://youtu.be/qfjxLRrHS0c

[5] [Java Code Sample] Create timer (normal/countdown/two digits). *RYISNOW'S PROGRAMMING BLOG* [online]. 14.4.2021 [cit. 2022-06-02]. Dostupné z: https://www.ryisnow.online/2021/04/java-beginner-code-sample-create-timer.html