*Dokumentácia – Sokoban Evolučným spôsobom – Matej Pavla (418138)*

**Spustenie projektu:**

Projekt bol vyvíjaný v prostredí NetBeans 7.2. Použitá bola štandardná Java, JDK verzia 1.7.
V Netbeans je spustenie projektu jednoduché:

1. Stačí si projekt („source/SokobanEvo“) načítať do Netbeans

2. A načítať potrebné knižnice, ktoré sú uvedené v priečinku „source/libs“.
V podstate je potrebné načítať: source/libs/slick.jar, source/libs/lwjgl-2.8.5/jar/lwjgl.jar, source/libs/lwjgl-2.8.5/jar/lwjgl\_util.jar, source/libs/lwjgl-2.8.5/jar/jinput.jar
 (Celý návod je dostupný na <http://www.slick2d.org/wiki/index.php/Setting_up_Slick2D_with_NetBeansIDE>)

**Project overview:**

Najdôležitejšie:

1.V sokobanevo.levels.LevelDraw:26:
*final static String mapFilePath = "assets/sasquatch\_s1.txt";***mapFilePath** –cesta k súboru k mapkami
K dispozícii sú zatiaľ dva, je jednoduché pridať ďalšie, stačí sa pozrieť na štruktúru už existujúcich súborov, nie je problém ju reprodukovať.

2.V sokobanevo.SokobanEvo:58:
*baseMap = levelDraw.readMapFromFileById(****ID****);***ID** – id mapky ktorou chceme pracovať

Základná classa je ***sokobanevo.SokobanEvo***:

Classa implementuje metódy pre Slick2D, v nich sa vykonáva príprava, kalkulovanie, rendering..
*Metóda init:* Načítanie mapky, pomocou classy *LevelDraw*, vytvorenie prvej generácie, pomocou statickej metódy classy *EvolutionSokoSolver*.
*Metóda update*: Volá sa prvýkrát po init() a vždy potom pred render(). Po nej sa volá render() a takto sa to opakuje, tzn. fps závisí na rýchlosti vykonania update(). Vypočítame v nej fitness súčasnej generácie. (EvolutionSokoSolver.computeFitnessOfGeneration v skutočnosti vracia penalty, na fitness to prevedieme prevrátením hodnôt). Plus sa tu spočítajú nejaké štatistické premenné ako priemerné fitness. Na konci funkcie ešte vytvoríme novú generáciu na základe fitness. (EvolutionSokoSolver.createNextGeneration)
*Metóda render:* prebieha v nej rendering napočítaných údajov

Classa ***sokobanevo.levels.LevelDraw***:

Classa má funkcie pre parsovanie levelu z TXT súboru (*readMapFromFileById())* a vykreslenie levlu pomocou Slick2D (*drawLvlAt()*, musí byť volaná v *SokobanEvo.render()*). V konštruktore sa inicializujú textúry.

Classa ***sokobanevo.evolution.EvolutionSokoSolver***:

Všetky funkcie sa týkajú samotného evolučného riešenia a sú statické. Všetky dôležité informácie sú popísané v JavaDocu.

Classa ***sokobanevo.evolution.IndividumFitness***:

Funkcia *EvolutionSokoSolver.computeFitnessOfGeneration()* počíta fitness na viacero jadrách. Toto je vlastne implementácia threadu ktorý počíta fitness pre istú (pridelenú) časť populácie. K tomu zase využíva statické funkcie classy EvolutionSokoSolver.

Balík ***sokobanevo.pathfinding:***

Classy v tomto balíčku obsahujú implementáciu A\* algoritmu ktorý využívame v riešení. Bližšie info priamo v JavaDocu.

**Testovanie:**

Súbor mapiek *assets/sasquatch\_s1.txt:*

Zložitejšie mapky, podarilo sa mi vyriešiť len prvé dve, v ostatných vo vyriešení bránia netriviálne detekovateľné deadlocky. (Poznámka: hviezdičky objavujúce sa v TXT týchto mapiek sú pre účely pravdepodobne nejakých hier a slúžia ako bonusové body, ja ich v projekte ignorujem.)

Súbor mapiek *assets/schaul.txt:*

Mapky z diplomovej práce Toma Schaula (<http://www.whatisthought.com/schaulthesis.pdf>). Podarilo sa mi vyriešiť 6/7.

Viac mapiek dostupných na: <http://users.bentonrea.com/~sasquatch/sokoban/>