Dokumentace k semestrální práci

Jan Pelikán

Obsah

Úvod ... 3
Fungování programu ... 3
Spuštění programu ... 3
Třídy a jejich metody ... 3
Prostor ke zlepšení, případné nedostatky ... 4
Šipky a jména ... 4
Vykreslování ... 4

Úvod

Než jsem začal vypracovávat zadání samostatné práce, popral jsem se s Gitlabem o stažení „kostry“ programu. Potom co jsem vyhrál bitvu, prostudoval jsem si Java dokumentaci k tomu, co již bylo naprogramované, abych věděl, co a jak dělá celý program. Když jsem nevěděl, jak dále pokračovat v seminární práci, dost mi pomáhalo povídat si o tom s někým, kdo tomu vůbec nerozumí. To mi pomohlo abych si věci ujasnil a napadlo mě řešení problému.

Fungování programu

Spuštění programu

Program se spouští souborem Run.cmd, při spuštění programu můžete navíc zadat jako parametr číslo od 0 do 3, podle toho, která chcete spustit scénář. 0 spustí první scénář. Vstup je ošetřen tak, aby při zadání jiného vstupu, než jsou čísla od 0-3 se spustil první scénář a vypsala se hláška do konzole. Vstup je ověřen pomocí if-else bloku.

Velikost okna se nastaví na velikost potřebnou pro přesné vykreslení scénáře.

Třídy a jejich metody

Program má pouze dvě třídy: „L01_SpusteniSimulatoru“ a „DrawingPanel“. První třída má v sobě Main, stará se pouze o spuštění DrawingPanel a nastavuje správnou velikost okna. Zajišťuje také aby se vykreslení obnovovalo každých 100ms. Třída DrawingPanel je zajímavější. Má spoustu metod.

První metoda ve třídě DrawingPanel je „*paint*“. V této metodě se neděje nic zajímavého, vykresluje celý program a stará se o správné škálování, které ovšem nepočítá. První zajímavá metoda je až *drawWaterLayer*.

V této metodě pomocí dvou cyklů „for“ vykresluji vodu. První cyklus je y-souřadnice a druhý cyklus počítá x-souřadnici. Celé to stojí na metodě *isDry*, která vrací jestli daný bod je mokrý nebo suchý. Pokud jsou buňky vedle sebe mokré, vykreslí se voda. Na konci metody se ptám, jestli zde mám i nějaké vodní zdroje, které případně vykreslím.

V předchozí metodě jsem zmiňoval vykreslení vodních zdrojů, o které se stará další metoda *drawWaterSources*. Tato metoda využívá metodu *drawWaterFlowLabel*, která má na starost vykreslit jméno řeky a směr toku řeky. Informace o pozicích mi dávat *WaterSourceUpdater*, pozici šipky zjistím přes getIndex() % maxX a getIndex()/maxX. V prvním případě mám operátor „modulo“, aby mi vrátil x-ovou souřadnici v řádce, řádku zjistím poté pomocí dělení, které mi vrátí řádek, na kterém se bod nachází.

Metoda *drawWaterFlowLabel* používá a přesouvá pozici přijmutou z parametru (aby šipky a jména nevedli z řeky, ale byli vedle řeky). Otáčí jména, aby byli vodorovně se šipkou a vykresluje jméno řeky a šipku.

Metoda na vykreslení šipek je hodně inspirována šipkami ze cvičení 3.

Další zajímavou metodou je *computeModel2Windowtansformation*. Tato metoda počítá škálování, aby se vykreslování nedeformovalo se změnou velikosti okna. A potom také počítá posun souřadnic. Tato metoda je hodně inspirována cvičením 6.

Prostor ke zlepšení, případné nedostatky

Šipky a jména

Vykreslování šipek -> šipky stále mění směr a drobet i velikost, vzhledově to není úplně hezké, to samé je i se jmény. Ve výsledky bych toto chování rád odstranil, ovšem mě teď nenapadá žádný způsob jak na to.

Vykreslování

Všiml jsem si, že při spuštění vykreslování se více než dvojnásobně zvedne práce CPU. Dost mě překvapilo, když jsem si všiml že na mém PC, běžel procesor na 95% a z toho 65% bylo vykreslování programu. Mám Intel Core i5 osmé generace 3,2 GHZ. Ovšem s tímto chováním také nevím co udělat.