Semestrální práce – Vesmír, část 1: Animace

Simulace pohybu vesmírných objektů

Úkolem bylo vytvořit simulaci pohybu vesmírných objektů. Základní třídy pro tuto simulaci jsou:

SpaceObj – třída, jejíž instance reprezentuje jeden vesmírný objekt.

Space – třída, jejíž instance reprezentuje vesmír, kde se vyskytují všechny vesmírné objekty.

DrawingPanel – třída, která zajišťuje vykreslení vesmíru a jeho vesmírných objektů.

CSVLoader – třída, která nám načte data z csv souboru a vytvoří z nich instanci vesmíru.

Coord2D – třída, která reprezentuje XY souřadnice. Využívá se návrhového vzoru přepravka.

Galaxy_SP2022 – hlavní třída, která zajistí inicializaci programu, ošetření uživatelských vstupů a opakované překreslení instance třídy DrawingPanel

Třída CSVLoader

Před začátkem simulace se nejdříve načtou data pomocí třídy CSVLoader. Tato třída pomocí metody parseDataToSpace načte data z poskytnutých souborů, které jsou ve formátu CSV a z nich vytvoří instanci třídy Space, která bude obsahovat její gravitační konstantu, krok v čase a všechny vesmírné objekty popsané v načteném souboru.

Třída Space

Třída Space obsahuje simulační čas, který se dá získat metodou getSimulationTime. Jeden z funkčních požadavků je pozastavení simulace. Ve třídě Galaxy_SP2022 je nastavený KeyboardFocusManager, který detekuje stisknutí mezerníku. Při stisknutí se vyvolá metoda startPause/stopPause, která nám zastaví/obnoví simulaci.

Pozastavení nám zajišťují metody startPause, getCurrentTime, stopPause. Jelikož je simulační čas závislý na metodě System.currentTimeMillis, musíme začít počítat uběhnutou dobu od pozastavení simulace. Tuto dobu odečteme od simulačního času a takto „pozastavíme“ čas. Aby pozastavení fungovalo vícekrát, musíme odečítat od simulačního času sumu všech „uběhnutých dob od pozastavení simulace“. Například: první pauza trvala 3 sekundy, druhá pauza trvala 7 sekund, třetí pauza trvala 4 sekundy… Suma těchto dob je 14 sekund a toto odečítáme od simulačního času.

Simulační čas je také ovlivněn krokem v času, který určuje kolik simulačních sekund proběhne za sekundu. Metoda getSimulationTime() toto zajišťuje vynásobením simulačního času v sekundách tímto krokem v času.

V této třídě je také implementovaná simulace pohybů vesmírných objektů. Metoda updateSystem si načte uběhnutý čas od poslední doby, kdy byla tato metoda vyvolána a provede kalkulaci všech pozic a rychlostí za tento čas. Předtím než se vypočítá budoucí pozice a rychlosti, tak se vypočítá zrychlení všech vesmírných objektů pomocí metody getAcceleration.

Třída DrawingPanel

Třída DrawingPanel si pomocí konstruktoru uloží referenci na instanci třídy Space. Zde zajišťuje vykreslování planet metoda drawPlanets. Úplnou viditelnost objektů v každém čase a maximální vyplnění dostupného prostoru okna nám zajišťují metody getScale a getMinMaxBounds (které se ale již vyvolává v metodě getScale). Metoda getMinMaxBounds nám vypočítá levý horní roh a pravý dolní roh, které ohraničují náš vesmír. Obdélník stvořený z těchto dvou bodů reprezentuje nejmenší možný obdélník, který v sobě obsahuje všechny vesmírné objekty. Tato metoda využívá i velikosti vesmírných objektů pro vypočítání těchto bodů.

Metoda getScale nám vypočítá vhodnou hodnotu pro metodu Graphics.Scale se kterou můžeme vhodně vyplnit dostupný prostor okna se zachováním poměru stran. K vycentrování vesmíru využijeme metodu Graphics.translate a atributy offsetX a offsetY jako parametry této metody, které jsme vhodně vypočítali v metodě paint. Pomocí getScale a attributů offsetX a offsetY jsme si také zajistili responzivitu našeho okna.

V pravém horním rohu okna se vypisuje aktuální simulační čas pomocí metody drawTime, která využívá metodu getSimulationTime z instance třídy Space.

Třída SpaceObj

Jedna instance této třídy si v sobě uchovává velikost, hmotnost, pozici a rychlost reprezentovanou instancí Coord2D, typ a název vesmírného objektu. Velikost objektu není určen parametrem z konstruktoru, ale je vypočítán vnitřně pomocí metody getRadius. K těmto atributům byly stvořeny gettery a settery.

Poloměr objektů určuje metoda getRadius ze třídy SpaceObj. Předpokládá se jednotková hustota všech objektů a využívá se vzorce pro objem koule.

Třída Coord2D

Jedna instance této třídy v sobě uchovává x-ové a y-ové souřadnice reprezentované datovým typem double. Tato třída existuje pro ulehčení práce s vektory a souřadnicemi. Je zde využit návrhový vzor Přepravka.

GalaxySP_2022

Ve třídě GalaxySP_2022 byl také přidán MouseListener na instanci třídy DrawingPanel. Při stisknutí levého tlačítka na myši se vyvolá metoda getSelected(Coord2D coord), kde parametry jsou relativní souřadnice (vůči panelu) myši při stisknutí reprezentovány instancí třídy Coord2D. Metoda getSelected detekuje, zda tyto souřadnice nejsou obsažené v jednom z našich vesmírných objektů. Pokud ano, do atributu DrawingPanel.selected se uloží reference na instanci této třídy. Metodou drawSelected se pak vypíšou informace o tomto objektu v prostřední dolní časti okna.

Momentální omezení a zjednodušení simulace, rozšíření do budoucna

Simulace využívá vzorce N-objektů pro výpočet pozice v daném časovém úseku. Nejdřív se vypočítá zrychlení všech planet, všechna zrychlení se uloží do pole. Pak se vypočítají pozice a rychlosti planet využitím tohoto pole.

Simulace momentálně nemusí fungovat správně, pokud nastane kolize mezi planetami (Například data ze souborů random100.csv, random500.csv a collision.csv fungují zvláštně). Řešením je implementovat kód, který sloučí planety, pokud jsou si moc blízko sebe. Kolize mezi planetami bude ošetřena a implementována v části 2 semestrální práce, jelikož část 1 toto nevyžaduje.

Pro vypočítání poloměru se předpokládá jednotková hustota naší planety. Jelikož je známá hustota a hmotnost planety, využije se vzorec pro vypočítání hustoty:

$$\rho = \frac{m}{V}$$

Kde m je hmotnost planety a V je objem planety. Jeden z možných problémů je, pokud objem interpretujeme jako obsah kružnice. V tomto případě bude poloměr Slunce větší než jakákoliv vzdálenost libovolné planety od Slunce. Pak se planety nemusí vykreslit ideálně, protože Slunce bude překrývat všechny naše planety. Z tohoto důvodu byl v programu využit vzorec pro objem koule místo vzorce pro obsah kružnice.

Popis vytvoření, instalace a spuštění aplikace

Stáhněte si celý projekt, spusťte Build.cmd respektive Build.sh. Teď je program vytvořený. Aplikace se spustí pomocí příkazového řádku, který je otevřený ve složce projektu. Příkaz na spuštění simulace je:

Run „cesta k dat“

Semestrální práce – Vesmír, část 2: Grafy a další rozšíření

Základní funkční požadavky 2. části semestrální práce

Kolize

K ošetření kolizi se vytvořila metoda collide(SpaceObj other) ve třídě SpaceObj. Tato metoda spojí dva vesmírné objekty, pokud instance, u které je tato metoda vyvolávaná, překrývá střed objektu z parametru a její váha je větší než váhá objektu z parametru. Sloučení je implementováno tak, že se sečte váha obou objektů a přepočítá se rychlost využitím zákonu zachování hybnosti. Při úspěšném sloučení metoda vrací true. Objekt v parametru je tedy absorbován objektem, u kterého se vyvolává tato metoda, ale objekt v parametru není nijak změněn nebo ovlivňen. V opačném případě se objekty nesloučí a metoda vrací false.

Vzorec pro vypočtení rychlosti využitím zákona zachování hybnosti, kde v_F je rychlost spojeného objektu:

$$m_1 v_1 + m_2 \cdot v_2 = (m_1 + m_2) \cdot v_F$$

Další metoda k ošetření kolize je metoda checkCollision() ve třídě Space. Tato metoda vyvolává metodu collide(SpaceObj other) a kontroluje zda nenastala kolize mezi některými objektami. Pokud ano, metoda collide(SpaceObj other) nám zajistí sloučení objektů a vymaže ze seznamu objektů ve Vesmíru ten objekt, který byl předán metodě collide.

Metoda checkCollision() se pravidělně vyvolává ve hlavní třídě Galaxy_SP2022 a kontroluje zda nenastala kolize.

Graf rychlosti

Graf byl implementován pomocí knihovny JFreeChart. Jelikož chceme vykreslit graf rychlosti, budeme rychlost vykreslovat jako spojnicový graf. Nejdříve se ale musí zajistit data na x-ové souřadnici (čas) a y-ové souřadnici (rychlost). Metoda trackPlanetVel(SpaceObj planet) ve třídě Space nám toto zajistí. Tato metoda do atributu trackTime uloží veškeré časy a do atributu trackVel uloží veškeré rychlosti objektů. Oba tyto atributy jsou mezi sebou provázány. K těmto atributům byly vytvořeny gettery getTrackTime() a getTrackVel().

Ve hlavní třídě se vytvoří graf pomocí statické továrny ChartFactory, pomocí metody getDataset(List<Double> cas, List<Double> rychlost) se získají data, která můžeme vložit do našeho grafu.

Graf zobrazuje rychlost vybraného vesmírného objektu za posledních 30 sekund v reálném času.

Trajektorie pohybu objektů

Trajektorie pohybu objektů je implementovaná metodou drawTrails(Graphics2D g2) ve třídě DrawingPanel. Tato metoda si uchovává pozice všech objektů za poslední 1 sekundu ve frontě trails. Využitím trails získáme všechny potřebné pozice za poslední 1 sekundu. Tato metoda pak vykreslí na těchto pozicích vyplněné kružnice. První zaznamenaná pozice má nejmenší kružnici, nejvyšší zaznamenaná pozice má největší kružnici, která je velikostně stejně velká jako objekt, které patří tato historie pozic. V případě, že nastane kolize se všechny trajektorie resetují.

Omezení a problémy

V případě kolize se trajektorie vždycky musí resetovat. Chyba je v tom, že kolize maže objekty a ošetření tohoto problému by vyžadovalo přepis celého programu.

Při kolizi se někdy v metodě drawPlanets(Graphics g2) vyhází výjimka ConcurrentModificationException. V tomto případě se nevykreslí jeden snímek. Tento problém jsem nedokázal vyřešit, pravděpodobně v tomto roli hraje mazaní z pole a kolize, naštěstí tato výjimka nastává výjimečně, takže vizuálně program funguje normálně.