FAKULTA
APLIKOVANÝCH VĚD
ZÁPADOČESKÉ
UNIVERZITY
V PLZNI

KIV/PRJ4

Využití knihovny java.util.concurrent

Petr Laštovka
A15B0055K
jokertwo@students.zcu.cz

Obsah

Kapitola 1

Úvod

1.1 Popis zadání

Navrhout a vypracovat vícevláknovou aplikaci využívající knihovnu java.util.concurrent[4] v progarmovacím jazyku Java.

1.2 Popis aplikace

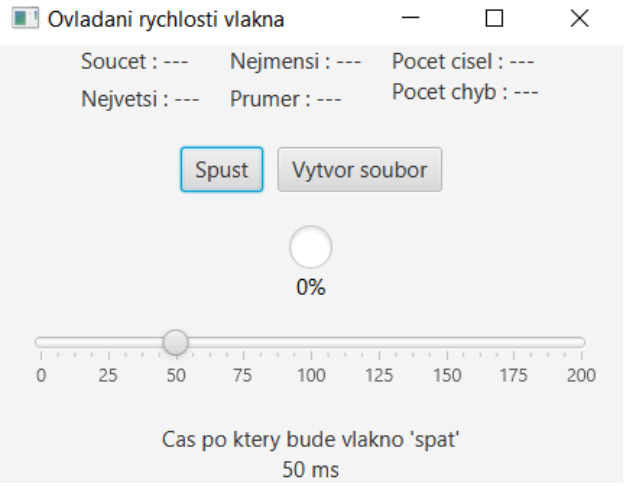
Aplikace pomocí jednoho vlákna načítá data ze souboru a ukládá je do fronty realizované pomocí LinkedBlockingQueue[2]. Další vlákno z této fronty data vybíra a používá. Jelikož aplikace má i GUI rozhraní, další vlákno má na-starosti jeho aktualizaci. Aplikace demostruje běh asynchorních vláken, kdy načítání ze souboru je rychlejší něž následné zpracování dat. Pro vyrovnání rozdílu potřebné doby je využita zmiňovaná fronta.

Kapitola 2

Uživatelská dokumentace

2.1 Spuštění programu

Program se spouští pomocí spustitelného souboru *PRJ4_A15B0055K.jar*. Pro bezproblémové spustění programu je nutné mít nainstalovanou JRE (Java Runtime Environment)[3]. Po spuštění programu se objeví okno na obrázku 2.1.

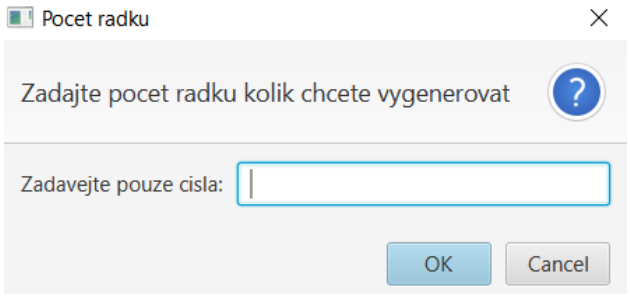


Obrázek 2.1: První spuštění.

2.2 Generování testovacího souboru

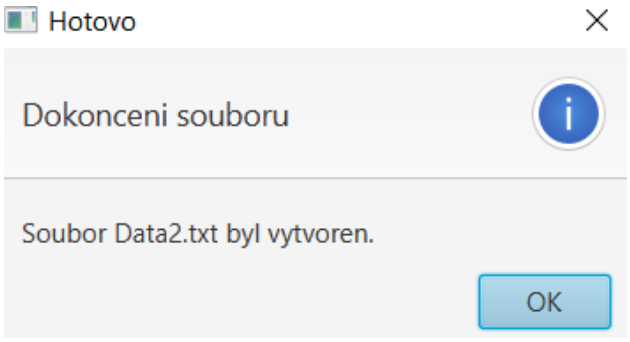
Pro správnou funkci aplikace je třeba vygenerovat testovací soubor. To lze pomocí tlačíka *Vytvor soubor*. Po kliknutí na toto tlačítko se otevře okno s volbou, kam se má generovaný soubor uložit.

Následně se otevře okno2.2 s volbou kolik řádků s čísly se má v souboru vygenerovat (čím větší číslo tím se bude následně soubor déle zpracovávat).



Obrázek 2.2: Počet řádků.

Po úspěšném generování je o tom uživatel informován patřičným oknem. Například jako na obrázku 2.3



Obrázek 2.3: Počet řádků.

2.3 Hlavní funkce

Pro spuštění hlavní funkce programu je třeba klidnout na tlačítko *Spust*. Po kliknutí bude uživatel vyzván pro výběr testovacího souboru *(Pro správnou funkci programu je třeba používat pouze soubory generované aplikací)*.

Po jeho zvolení se ze souboru začnou načítat data. O průběhu procesu je uživatel informován v horní části okna informacemi jako například o celkovém počtu načtených čísel, jejich součet, největší a nejmenší číslo.

Pod tlačítky se nachází ProgressIndicator, který informuje celkovém průběhu zpracování. Procentuelně vyjadřuje zpracování daných dat.

Ve spodní části okna se nachází *Slider*, pomocí kterého lze ovládat rychlost vlákna zpracovávajícího data ve frontě. Slider určuje po jakou dobu bude vlakno uspáno.

Kapitola 3

Popis implementace

Při návrhu aplikace jsem kladl důraz na oddělení jenotlivých funkcí do samostatných tříd. Zvláště se jedna o oddělení GUI a operace s vlákny na pozadí. Samotné GUI je napsané v JavaFX 8[5]. Ve třídách realizujících jednotlivá vlákna je implementováno rozhraní *runnable*[6]. Vlákna jsou pak spouštěna pomocí *ExecutorService*

Samotná implementace je navržena tak, že o čtení ze souboru se stará třída Read.java, která je rovnou při čtení ukládá do fronty[2]. O výběr dat z fronty se stará třída TakeFromQueue.java.

Generování testovacích souborů má na starosti CreateFile.java.

Všechny výše zmíněné třídy implementují rozhraní *runnable* a tvoří tak samostatná vlákna spouštěná ve třídě ManageOfThread.java pomocí *ExecutorService*. Tato třída se stará i o řádné ukončení každého vlákna, které bylo spuštěno.

Nad touto realizací je třída Gui.java, které obsahuje samotné GUI.

Kapitola 4

Závěr

Samotná aplikace v současném stavu nemá příliš praktických využití. Proto jsem nevěnoval příliš velkou poroznost samotnému zpracování GUI. Věřím ale, že je pro demostraci daného problému dostačující.

Práce na tomto projektu byla pro mne přínosem. Měl jsem díky ní možnost si vyzkoušet vytvoření GUI aplikace, která využívá procesů na pozadí. Díky této možnosti aplikace „nezamrzá" a práce s ní bz pro uživatele měla být příjemnější.

Při vývoji této aplikace se mě osvědčila práce s repozitářem na GitHub[8]. Obzvláště pak funkce verzování byla pro mne velmi přínosná.

Literatura

[1] Package java.util.concurrent :
https://docs.oracle.com/javase/8/docs/api/?java/util/concurrent/-package-summary.html

[2] Class LinkedBlockingQueue<E> :
https://docs.oracle.com/javase/8/docs/api/?java/util/concurrent/-LinkedBlockingQueue.html

[3] Java Runtime Environment :
https://www.java.com/en/download/

[4] Class ProgressIndicator :
https://docs.oracle.com/javase/8/javafx/api/javafx/scene/-control/ProgressIndicator.html

[5] JavaFX 8 :
https://docs.oracle.com/javase/8/javafx/api/toc.htm

[6] Interface Runnable :
https://docs.oracle.com/javase/8/docs/api/java/lang/-Runnable.html

[7] Interface ExecutorSevice :
https://docs.oracle.com/javase/8/docs/api/java/util/-concurrent/ExecutorService.html

[8] GitHub :
https://github.com/Jokertwo/PRJ4