Antoni Pawlak

401480

Laboratorium 7: JAVA

Cel ćwiczenia:

- Opanowanie podstaw tworzenia wątków w Javie.
- Opanowanie podstawowych metod synchronizacji w Javie.
- Dekompozycja pętli

Wykonanie

Przygotowania

1. Pobranie ze strony przedmiotu plików, utworzenie projektu w IntelliJ IDEA, dodanie tam plików Obraz.java oraz Histogram_test.java.
2. Uruchomienie kodu i sprawdzenie poprawności działania na małych tablicach

Wariant 1

3. Utworzenie nowej klasy Watek dziedziczącej po Thread

```
public class Watek extends Thread{}
```

4. Dodanie konstruktora klasy gdzie przekazujemy obraz oraz numer znaku

```
Watek(int charIndex, Obraz o) {
 this.charIndex = charIndex;
 this.o = o;
}
```

5. Zaimplementowanie metody run()

```
public void run(){
 o.calculate_line_histogram(charIndex);
 o.print_histogram_line(charIndex);
}
```

6. Utworzenie tylu wątków ile mamy znaków

```
int num_threads = 94;
Watek[] NewThr = new Watek[num_threads];
```

7. Uruchomienie wątków

```
for (int i = 0; i < num_threads; i++) {
 (NewThr[i] = new Watek(i, obraz_1)).start();
}
```

8. Zaczekanie na wątki

```
for (int i = 0; i < num_threads; i++) {
    try {
     NewThr[i].join();
    } catch (InterruptedException e) {}
}
```

9. Sprawdzenie poprawności programu

```
ystem.out.println("Czy poprawnie? : " + obraz_1.validateCalculations());
```

Wariant 2

10. Utworzenie nowej klasy Watek2 implementującej interfejs Runnable

```
public class Watek2 implements Runnable{}
```

11. Dodanie konstruktora

```
Watek2(int start, int end, Obraz o){
 this.start = start;
 this.end = end;
 this.o = o;
}
```

12. Zaimplementowanie metody run()

```
public void run() {
 for(int i = start; i < end; i++) {
 this.o.calculate_line_histogram(i);
 this.o.print_histogram_line(i);
 }
}
```

13. Wyliczenie skoku blokowego oraz „reszty” z dzielenia

```
int spacer = 94 / num_threads;
int rest = 94 - spacer * num_threads;
```

14. Uruchomienie wątków

```
for (int i = 0; i < num_threads; i++) {
    if(i == num_threads - 1) (NewThr[i] = new Thread(new Watek2(i*spacer,
(i+1)*spacer + rest, obraz_1))).start();

    else (NewThr[i] = new Thread(new Watek2(i*spacer,(i+1)*spacer,
obraz_1))).start();
}
```

15. Zaczekanie na wątki

```
for (int i = 0; i < num_threads; i++) {
 try {
 NewThr[i].join();
 } catch (InterruptedException e) {}
}
```

16. Sprawdzenie poprawności kodu

```
System.out.println("Czy poprawnie? : " + obraz_1.validateCalculations());
```

Wnioski

Thread a Runnable

Wyróżniamy dwa sposoby tworzenia nowych wątków w języku Java

- stworzenie własnej klasy która będzie subklasą Thread
  - możemy **zainstancjonować klasę i wystartować wątek**
- stworzenie klasy implementującej interfejs Runnable
  - aby wystartować wątek należy podać jako **argument do nowej instancji Thread**

Implementacja interfejsu Runnable zajmuje mniej pamięci i daje więcej możliwości, pozwala aby nasza klasa rozszerzała jeszcze inne klasy. Do uruchomienia wątku potrzebuje jednak instancji Thread.

Użycie subklasy Thread zajmuje więcej pamięci, nie pozwala na rozszerzanie innej klasy bo już rozszerza Thread. Pozwala jednak startować bez tworzenia nowej instancji Thread.

Wniosek, należy zastanowić się co jest dla naszego programu ważniejsze, czy możliwość rozszerzania innych klas czy łatwość startowania wątków.

Figure 1: Source: makeinjava.com

Obsługa wątków w Java

Każdy wątek ma nazwę na potrzeby identyfikacji, ale może zdarzyć się tak, że dwa wątki będą nazywać. Jeżeli nie podamy nazwy wątku to jest ona za nas generowana automatycznie.

Cykle życia wątku:

- Narodziny: kiedy tworzymy nową instancję „new”, ale jeszcze nie uruchamiamy „start()”
- Runnable: kiedy thread wystartuje staje się Runnable
- Oczekiwanie: czasem zdarzy się, że wątek musi poczekać na inny
- Zdefiniowne oczekiwanie: ustalamy wątkowi określony czas pauzy
- Zabity: kiedy ukończy swoje zadanie lub zostaje zabity

Prorytety wątków, w java mamy możliwość przekazania do systemy operacyjnego informację o tym jak ważne są dla nas obliczenia danego wątku. Jest to zakres 1-10, gdzie MIN_PRIORITY jest równe 1, a MAX_PRIORITY równe 10. Domyślna wartość to NORM_PRIORITY równa 5. Oczywiście nadanie wątku nie koniecznie gwarantuje chronologię wykonania operacji.

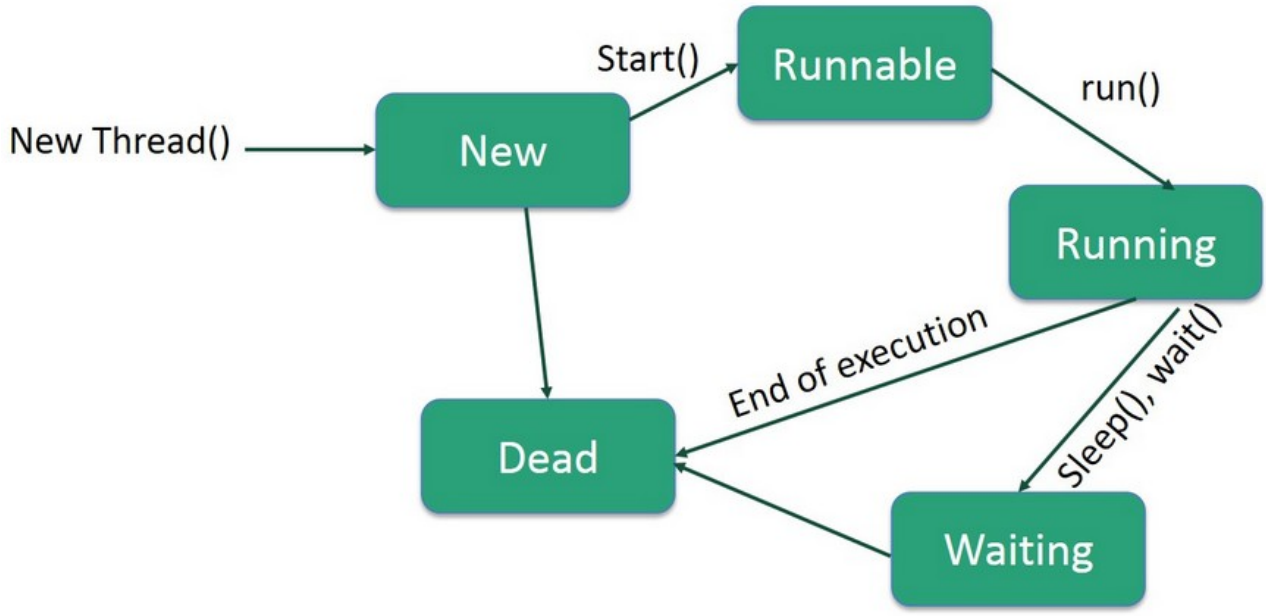

Figure 2: Source: tutorialspoint.com

Dekompozycja