PROGRAMOWANIE WSPÓBIEŻNE 2021/2022

Projekt semestralny pt. „Sieć komputerowa”

Autor: Maciej Kawka

Nr. indexu: 76797

Grupa: WCY20IY2S1

1. Treść zadania.

Obraz zawierający tekst

Opis wygenerowany automatycznie

1. Syntetyczny opis problemu – w tym wszystkie przyjęte założenia.

Zadanie polegało na odpowiednim zsynchronizowaniu wątków(komputerów), które na zmianę robią sprawy własne, po czym proszą o dostęp do jednego z urządzeń(drukarka, skaner itd.). Stacje robocze oraz urządzenia zostały ustawione w grupy dostępu. Zasadnicza część działania programu polega na tym, że nie może się zdarzyć aby dwa komputery były jednocześnie połączone do tego samego urządzenia, gdyż urządzenia obsługują tylko jedną stacje roboczą na raz. To samo tyczy się grup tj. nie może dojść do przypadku, kiedy dwa komputery z tej samej grupy łączą się do urządzeń należących do tej samej grupy urządzeń. W celu wykonania programu przyjąłem założenia:

- Komputery z pracują oraz łączą się z różnymi prędkościami(tak jak jest to w prawdziwej sieci komputerowej)

- Kiedy dwa komputery z jednej grupy proszą o dostęp do urządzeń z tej samej grupy urządzeń, jedna stacja robocze musi czekać.

- Komputerami są szare kwadraciki na górze ekranu. Kiedy komputer się przyłącza do sieci to pojawia się na nim kółko. Podczas wykonywania spraw własnych kółko na komputerze się powiększa, po czym jeżeli komputer podejmie próbę połączenia się z urządzeniem to poprosi o przyzwolenie i kiedy je otrzyma wysyła kółko(pakiet danych do urządzenia). Kiedy komputer pracuje na danym urządzeniu to jest pomiędzy nim, a owym urządzeniem linia. Po skończonej pracy kółko wraca do komputera i znika.

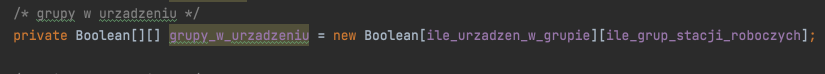
- Komputery łączą się do sieci różną ilość razy.(Tak jak w prawdziwej sieci)

Techniczny aspekt problemu rozwiązałem za pomocą technologii monitorów.

Każdy proces, który jest obiektem klasy StacjaRobocza po wykonaniu spraw własnych tj. utworzenia koła i powiększenia go, uruchamia funkcje uzyskaj\_dostęp(), dostępną z obiektu monitora. Monitorem jest obiekt klasy Server. Po uruchomieniu tej funkcji Server najpierw sprawdza czy w urządzeniu nie ma już wątku z tej samej grupy, poprzez sprawdzenie czy w tablicy grupy\_w\_urzadzeniu pod pozycją wobec wątku [grupa urządzeń, którą chce wziąć][grupa urządzeń, w której jestem]. Jeżeli okaże się że już jakiś wątek z tej samej grupy korzysta z urządzenia z grupy urządzeń, o którą jest zapytanie, to wątek wywołuje grupa[co\_bierzesz][gr].await() i czeka na signal() od stacji roboczej z tej samej grupy stacji roboczych, która opuszcza urządzenie z odpowiedniej grupy urządzeń. Jeżeli ten warunek zostanie spełniony to sprawdzane są urządzenia z oczekiwanej grupy urządzeń. Jeżeli uda się znaleźć wolne urządzenia to wszystkie dostępy są uzyskane i wątek oznacza, że zajmuje dane urządzania. Jeżeli natomiast wszystkie urządzenia są zajęte to jest wykonywana operacja await() na zmiennej warunkowej urządzenie[co\_bierzesz], przez co wątek czeka na pierwszą stacje roboczą która opuści urządzenie. Po czym jeżeli jakiś wątek który opuszcza urządzenie wykona urządzenie[co\_bierzesz].siglen(), sprawdzany jest na nowo stan urządzeń i po wybraniu właściwego urządzenia dostępy są przydzielone.

1. Wykaz współdzielonych zasobów.

Server.java



1. Wykaz punktów synchronizacji.

Server.java

Obraz zawierający tekst

Opis wygenerowany automatycznie

1. Wykaz obiektów synchronizacji.

Server.java

Obraz zawierający tekst

Opis wygenerowany automatycznie

1. Wykaz procesów sekwencyjnych.

StacjaRobocza.java

package siec.javafxprojektpw9\_2022\_projekt;  
  
import javafx.animation.PathTransition;  
import javafx.animation.ScaleTransition;  
import javafx.application.Platform;  
import javafx.scene.image.ImageView;  
import javafx.scene.layout.AnchorPane;  
import javafx.scene.paint.Color;  
import javafx.scene.shape.\*;  
import javafx.util.Duration;  
  
import java.util.Random;  
  
public class StacjaRobocza extends Thread {  
 private int grupa;  
 private int rodzaj;  
 //String nazwa;  
 private int N;  
 private Server server;  
 private int numer;  
 int LR;  
 int rr;  
 AnchorPane animacja;  
 Rectangle mojaStacja;  
 Rectangle[][] urzadzenia = new Rectangle[3][3];  
 int ile\_urzodzen;  
 int szybciej;  
  
 StacjaRobocza(int numer, int grupa, Server server, int N, AnchorPane animacja, Rectangle mojaStacja, Rectangle[][] urzadzenia, int ile\_urzodzen, int szybciej) {  
 super("W"+numer);  
 this.grupa = grupa;  
 this.server = server;  
 this.N = N;  
 this.animacja = animacja;  
 this.mojaStacja = mojaStacja;  
 this.urzadzenia = urzadzenia;  
 this.ile\_urzodzen = ile\_urzodzen;  
 this.szybciej = szybciej;  
 }  
  
 Random r = new Random();  
 Random rrr = new Random();  
 Random R = new Random();  
 public void run() {  
 /\*--------------Done---------------\*/  
 N = rrr.nextInt(9)+1;  
 for ( int i = 1; i <= N; i++ ) {  
 try {  
 int RR = R.nextInt(5)+1;  
 /\*--------------Tworzenie---------------\*/  
 Circle circle = new Circle();  
 circle.setCenterX(20+mojaStacja.getLayoutX());  
 circle.setCenterY(20+mojaStacja.getLayoutY());  
 circle.setRadius(0);  
 circle.setStroke(Color.*BLACK*);  
 circle.setFill(Color.*RED*);  
  
 Platform.*runLater*(() -> {  
 animacja.getChildren().add(circle);  
 });  
 /\*--------------Sprawy wlasne---------------\*/  
 /\*--------------Powiekszanie---------------\*///0->20  
 ScaleTransition scaleTransition = new ScaleTransition();  
 scaleTransition.setDuration(Duration.*millis*(1000/szybciej));  
 scaleTransition.setNode(circle);  
 scaleTransition.setByX(40);  
 scaleTransition.setByY(40);  
 Platform.*runLater*(()->{  
 scaleTransition.play();  
 });  
  
 try {  
 *sleep*(1000/szybciej);  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 /\*--------------Done---------------\*/  
  
 rr = r.nextInt(ile\_urzodzen);//bylo 2  
 LR = server.uzyskaj\_dostep(getName(), rr, grupa, i); // String nazwa, int co\_bierzesz, int gr, int nr\_powt  
 /\*--------------Linia---------------\*/  
 System.*out*.println(rr+" "+LR);  
 Line line = new Line();  
 line.setStartX(20+mojaStacja.getLayoutX());  
 line.setStartY(20+mojaStacja.getLayoutY());  
 line.setEndX(25+urzadzenia[rr][LR].getLayoutX());  
 line.setEndY(25+urzadzenia[rr][LR].getLayoutY());  
 Platform.*runLater*(() -> {  
 animacja.getChildren().add(line);  
 });  
 // LR jezeli 0 to lewy jezeli 1 to prawy  
 /\*--------------Przesuwanie---------------\*/  
 Path path = new Path();  
 MoveTo moveTo = new MoveTo();  
 moveTo.setX(circle.getCenterX());  
 moveTo.setY(circle.getCenterY());  
 LineTo lineTo = new LineTo();  
 lineTo.setX(25+urzadzenia[rr][LR].getLayoutX());  
 lineTo.setY(25+urzadzenia[rr][LR].getLayoutY());  
 path.getElements().addAll(moveTo,lineTo);  
 PathTransition pathTransition = new PathTransition(Duration.*millis*((2000\*RR)/szybciej), path, circle);  
 System.*out*.println("Puszczam animacje");  
 Platform.*runLater*(() -> {  
 pathTransition.play();  
 });  
  
 try {  
 *sleep*((4000\*RR)/szybciej);  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 //sleep(r.nextInt(5));  
 //System.out.println("Moje urzo to "+LR+" jestem "+getName());  
  
 server.zwolnij\_zasob(getName(), rr, grupa, i, LR); // String nazwa, int co\_bierzesz, int gr, int nr\_powt, int ktore  
 /\*--------------Powrot---------------\*/  
 Path backPath = new Path();  
 MoveTo backMoveToo = new MoveTo();  
 backMoveToo.setX(25+urzadzenia[rr][LR].getLayoutX());  
 backMoveToo.setY(25+urzadzenia[rr][LR].getLayoutY());  
 LineTo BacklineToo = new LineTo();  
 BacklineToo.setX(20+mojaStacja.getLayoutX());  
 BacklineToo.setY(20+mojaStacja.getLayoutY());  
 backPath.getElements().addAll(backMoveToo,BacklineToo);  
 PathTransition backPathTransitionn = new PathTransition(Duration.*millis*((2000\*RR)/szybciej), backPath, circle);  
 System.*out*.println("Puszczam animacje");  
 Platform.*runLater*(() -> {  
 backPathTransitionn.play();  
 });  
  
 try {  
 *sleep*((2000\*RR)/szybciej);  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 Platform.*runLater*(() -> {  
 animacja.getChildren().remove(scaleTransition);  
 animacja.getChildren().remove(backPathTransitionn);  
 animacja.getChildren().remove(moveTo);  
 animacja.getChildren().remove(path);  
 animacja.getChildren().remove(lineTo);  
 animacja.getChildren().remove(backPath);  
 animacja.getChildren().remove(backPathTransitionn);  
 animacja.getChildren().remove(backMoveToo);  
 animacja.getChildren().remove(BacklineToo);  
 animacja.getChildren().remove(line);  
 });  
  
 /\*--------------Znikanie---------------\*/  
 ScaleTransition puff = new ScaleTransition();  
 puff.setDuration(Duration.*millis*((1000\*RR)/szybciej));  
 puff.setNode(circle);  
 puff.setByX(-40);  
 puff.setByY(-40);  
 puff.setAutoReverse(true);  
 puff.setCycleCount(1);  
 Platform.*runLater*(() -> {  
 puff.play();  
 });  
 try {  
 *sleep*((1000\*RR)/szybciej);  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
  
 Platform.*runLater*(() -> {  
 animacja.getChildren().remove(circle);  
 animacja.getChildren().remove(puff);  
 });  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
}

1. Listing programu.

Server.java

package siec.javafxprojektpw9\_2022\_projekt;  
  
import java.util.Random;  
import java.util.concurrent.locks.Condition;  
import java.util.concurrent.locks.\*;  
import java.util.concurrent.locks.ReentrantLock;  
  
public class Server {  
 public int ile\_urzadzen\_w\_grupie = 3;  
 public int ile\_grup\_urzadzen = 3;  
 public int ile\_grup\_stacji\_roboczych = 3;  
  
 Server (int ile\_urzadzen\_w\_grupie,int ile\_grup\_urzadzen,int ile\_grup\_stacji\_roboczych) {  
 this.ile\_urzadzen\_w\_grupie = ile\_urzadzen\_w\_grupie;  
 this.ile\_grup\_urzadzen = ile\_grup\_urzadzen;  
 this.ile\_grup\_stacji\_roboczych = ile\_grup\_stacji\_roboczych;  
 }  
 /\* Obiekt zamka \*/  
 private final Lock lock = new ReentrantLock();  
 /\* Tablica do zajmowania urzadzen \*/  
 private Boolean[][] urzadzenia = new Boolean[ile\_grup\_urzadzen][ile\_urzadzen\_w\_grupie];  
 /\* Tablica do blokowania swojej grupy \*/  
 //private int ile\_stacji\_roboczych = 2;  
 private Condition[][] grupa = new Condition[ile\_grup\_urzadzen][ile\_grup\_stacji\_roboczych];  
 /\* Blokady poszczegulnych grup urzadzen \*/  
 private Condition[] urzadzenie = new Condition[ile\_grup\_urzadzen];  
 /\* grupy w urzadzeniu \*/  
 private Boolean[][] grupy\_w\_urzadzeniu = new Boolean[ile\_urzadzen\_w\_grupie][ile\_grup\_stacji\_roboczych];  
  
 /\* Zmienne pomocnicze \*/  
 //Boolean zajete = false;  
 int j;  
 private int[][] moje = new int[ile\_grup\_urzadzen][ile\_urzadzen\_w\_grupie];  
  
 /\* Funkcja do ustalenia parametrow \*/  
 public void init() {  
 /\* Init dla zajetosci urzadzen \*/  
 for ( int i = 0; i < /\*ile\_grup\_urzadzen\*/3; i++ ) {  
 for ( int j = 0; j < ile\_urzadzen\_w\_grupie; j++ ) {  
 urzadzenia[i][j] = false;  
 }  
 }  
 /\* Init dla blokowania grup \*/  
 for ( int i = 0; i < /\*ile\_grup\_urzadzen\*/3; i++ ) {  
 for ( int j = 0; j < ile\_grup\_stacji\_roboczych; j++ ) {  
 grupa[i][j] = lock.newCondition();  
 }  
 }  
 /\* Init dla blokowania grup urzadzen \*/  
 for ( int i = 0; i < /\*ile\_grup\_urzadzen\*/3; i++ ) {  
 urzadzenie[i] = lock.newCondition();  
 }  
  
 for ( int i = 0; i < /\*ile\_grup\_stacji\_roboczych\*/3; i++ ) {  
 for ( int j = 0; j < /\*ile\_urzadzen\_w\_grupie\*/3; j++ ) {  
 grupy\_w\_urzadzeniu[i][j] = false;  
 }  
 }  
 }  
  
 public int uzyskaj\_dostep( String nazwa, int co\_bierzesz, int gr, int nr\_powt ) throws InterruptedException {  
 lock.lock();  
 try {  
 j = 0;  
 if ( grupy\_w\_urzadzeniu[co\_bierzesz][gr] ) {  
 System.*out*.println("Blok 1");  
 grupa[co\_bierzesz][gr].await();  
 }  
 Boolean zajete = true;  
 for ( int i = 0; i < ile\_urzadzen\_w\_grupie; i++ ) {  
 if ( !urzadzenia[co\_bierzesz][i] ) {  
 zajete = false;  
 j = i;  
 break;  
 }  
 }  
 // Jezeli wszystkie skanery sa zajete to czekanie na odp  
 if ( zajete ) {  
 System.*out*.println("Blok 2");  
 urzadzenie[co\_bierzesz].await();  
 for ( int i = 0; i < ile\_urzadzen\_w\_grupie; i++ ) {  
 if ( !urzadzenia[co\_bierzesz][i] ) {  
 j = i;  
 break;  
 }  
 }  
 }  
 // Teraz przydzielone wszystkie dostepy  
 grupy\_w\_urzadzeniu[co\_bierzesz][gr] = true;  
 urzadzenia[co\_bierzesz][j] = true;  
 System.*out*.println("["+nazwa+","+nr\_powt+"] >> biore "+j+" stan urzadzenia nr {"+co\_bierzesz+"} ["+urzadzenia[co\_bierzesz][0]+","+urzadzenia[co\_bierzesz][1]+"] moja grupa "+gr);  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 } finally {  
 lock.unlock();  
 }  
 return j;  
 }  
  
 public void zwolnij\_zasob( String nazwa, int co\_bierzesz, int gr, int nr\_powt, int ktore ) throws InterruptedException {  
 lock.lock();  
 try{  
 grupy\_w\_urzadzeniu[co\_bierzesz][gr] = false;  
 urzadzenia[co\_bierzesz][ktore] = false;  
 //urzadzenia[co\_bierzesz][moje[gr][jato]] = false;  
 urzadzenie[co\_bierzesz].signal();  
 grupa[co\_bierzesz][gr].signal();  
 System.*out*.println("["+nazwa+","+nr\_powt+"] <<< urzadzenie "+ktore+" stan {"+co\_bierzesz+"} ["+urzadzenia[co\_bierzesz][0]+","+urzadzenia[co\_bierzesz][1]+"] a to ");  
 } finally {  
 lock.unlock();  
 }  
 }  
  
}

StacjaRobocza.java

package siec.javafxprojektpw9\_2022\_projekt;  
  
import java.util.Random;  
import java.util.concurrent.locks.Condition;  
import java.util.concurrent.locks.\*;  
import java.util.concurrent.locks.ReentrantLock;  
  
public class Server {  
 public int ile\_urzadzen\_w\_grupie = 3;  
 public int ile\_grup\_urzadzen = 3;  
 public int ile\_grup\_stacji\_roboczych = 3;  
  
 Server (int ile\_urzadzen\_w\_grupie,int ile\_grup\_urzadzen,int ile\_grup\_stacji\_roboczych) {  
 this.ile\_urzadzen\_w\_grupie = ile\_urzadzen\_w\_grupie;  
 this.ile\_grup\_urzadzen = ile\_grup\_urzadzen;  
 this.ile\_grup\_stacji\_roboczych = ile\_grup\_stacji\_roboczych;  
 }  
 /\* Obiekt zamka \*/  
 private final Lock lock = new ReentrantLock();  
 /\* Tablica do zajmowania urzadzen \*/  
 private Boolean[][] urzadzenia = new Boolean[ile\_grup\_urzadzen][ile\_urzadzen\_w\_grupie];  
 /\* Tablica do blokowania swojej grupy \*/  
 //private int ile\_stacji\_roboczych = 2;  
 private Condition[][] grupa = new Condition[ile\_grup\_urzadzen][ile\_grup\_stacji\_roboczych];  
 /\* Blokady poszczegulnych grup urzadzen \*/  
 private Condition[] urzadzenie = new Condition[ile\_grup\_urzadzen];  
 /\* grupy w urzadzeniu \*/  
 private Boolean[][] grupy\_w\_urzadzeniu = new Boolean[ile\_urzadzen\_w\_grupie][ile\_grup\_stacji\_roboczych];  
  
 /\* Zmienne pomocnicze \*/  
 //Boolean zajete = false;  
 int j;  
 private int[][] moje = new int[ile\_grup\_urzadzen][ile\_urzadzen\_w\_grupie];  
  
 /\* Funkcja do ustalenia parametrow \*/  
 public void init() {  
 /\* Init dla zajetosci urzadzen \*/  
 for ( int i = 0; i < /\*ile\_grup\_urzadzen\*/3; i++ ) {  
 for ( int j = 0; j < ile\_urzadzen\_w\_grupie; j++ ) {  
 urzadzenia[i][j] = false;  
 }  
 }  
 /\* Init dla blokowania grup \*/  
 for ( int i = 0; i < /\*ile\_grup\_urzadzen\*/3; i++ ) {  
 for ( int j = 0; j < ile\_grup\_stacji\_roboczych; j++ ) {  
 grupa[i][j] = lock.newCondition();  
 }  
 }  
 /\* Init dla blokowania grup urzadzen \*/  
 for ( int i = 0; i < /\*ile\_grup\_urzadzen\*/3; i++ ) {  
 urzadzenie[i] = lock.newCondition();  
 }  
  
 for ( int i = 0; i < /\*ile\_grup\_stacji\_roboczych\*/3; i++ ) {  
 for ( int j = 0; j < /\*ile\_urzadzen\_w\_grupie\*/3; j++ ) {  
 grupy\_w\_urzadzeniu[i][j] = false;  
 }  
 }  
 }  
  
 public int uzyskaj\_dostep( String nazwa, int co\_bierzesz, int gr, int nr\_powt ) throws InterruptedException {  
 lock.lock();  
 try {  
 j = 0;  
 if ( grupy\_w\_urzadzeniu[co\_bierzesz][gr] ) {  
 System.*out*.println("Blok 1");  
 grupa[co\_bierzesz][gr].await();  
 }  
 Boolean zajete = true;  
 for ( int i = 0; i < ile\_urzadzen\_w\_grupie; i++ ) {  
 if ( !urzadzenia[co\_bierzesz][i] ) {  
 zajete = false;  
 j = i;  
 break;  
 }  
 }  
 // Jezeli wszystkie skanery sa zajete to czekanie na odp  
 if ( zajete ) {  
 System.*out*.println("Blok 2");  
 urzadzenie[co\_bierzesz].await();  
 for ( int i = 0; i < ile\_urzadzen\_w\_grupie; i++ ) {  
 if ( !urzadzenia[co\_bierzesz][i] ) {  
 j = i;  
 break;  
 }  
 }  
 }  
 // Teraz przydzielone wszystkie dostepy  
 grupy\_w\_urzadzeniu[co\_bierzesz][gr] = true;  
 urzadzenia[co\_bierzesz][j] = true;  
 System.*out*.println("["+nazwa+","+nr\_powt+"] >> biore "+j+" stan urzadzenia nr {"+co\_bierzesz+"} ["+urzadzenia[co\_bierzesz][0]+","+urzadzenia[co\_bierzesz][1]+"] moja grupa "+gr);  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 } finally {  
 lock.unlock();  
 }  
 return j;  
 }  
  
 public void zwolnij\_zasob( String nazwa, int co\_bierzesz, int gr, int nr\_powt, int ktore ) throws InterruptedException {  
 lock.lock();  
 try{  
 grupy\_w\_urzadzeniu[co\_bierzesz][gr] = false;  
 urzadzenia[co\_bierzesz][ktore] = false;  
 //urzadzenia[co\_bierzesz][moje[gr][jato]] = false;  
 urzadzenie[co\_bierzesz].signal();  
 grupa[co\_bierzesz][gr].signal();  
 System.*out*.println("["+nazwa+","+nr\_powt+"] <<< urzadzenie "+ktore+" stan {"+co\_bierzesz+"} ["+urzadzenia[co\_bierzesz][0]+","+urzadzenia[co\_bierzesz][1]+"] a to ");  
 } finally {  
 lock.unlock();  
 }  
 }  
  
}

HelloApplication.java

package siec.javafxprojektpw9\_2022\_projekt;  
  
import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Scene;  
import javafx.scene.control.SplitPane;  
import javafx.scene.layout.\*;  
import javafx.stage.Stage;  
import javafx.scene.Group;  
  
import java.io.IOException;  
  
public class HelloApplication extends Application {  
  
 @Override  
 public void start(Stage stage) throws IOException {  
 FXMLLoader fxmlLoader = new FXMLLoader(HelloApplication.class.getResource("hello-view.fxml"));  
 Scene scene = new Scene(fxmlLoader.load(), 750, 500);  
 stage.setTitle("Siec komputerowa!");  
 stage.setScene(scene);  
 stage.show();  
 }  
  
 public static void main(String[] args) {  
 *launch*();  
 }  
}

HelloController.java

package siec.javafxprojektpw9\_2022\_projekt;  
  
import javafx.animation.PathTransition;  
import javafx.animation.RotateTransition;  
import javafx.animation.ScaleTransition;  
import javafx.animation.TranslateTransition;  
import javafx.application.Platform;  
import javafx.fxml.FXML;  
//import javafx.scene.control.Label;  
import javafx.scene.control.\*;  
import javafx.scene.image.Image;  
import javafx.scene.image.ImageView;  
import javafx.scene.layout.AnchorPane;  
import javafx.scene.layout.Pane;  
import javafx.scene.paint.Color;  
import javafx.scene.shape.\*;  
import javafx.scene.text.\*;  
import javafx.util.Duration;  
  
  
public class HelloController {  
  
 //in the Controller class  
 @FXML  
 private Slider ile\_stacji\_roboczych\_w\_grupie;  
 @FXML  
 private Slider ile\_grup\_stacji\_roboczych;  
 @FXML  
 private Slider ile\_urzadzen\_w\_grupie;  
 @FXML  
 private Slider ile\_grup\_urzadzen;  
 @FXML  
 private Text textpierw;  
 @FXML  
 private Text textdwa;  
 @FXML  
 private Text texttrzy;  
 @FXML  
 private Text textcztery;  
 @FXML  
 private AnchorPane animacja = new AnchorPane();  
  
 int fasterr = 1;  
  
  
 int ile\_stacji\_roboczych\_w\_gr;  
 int ile\_gr\_stacji\_roboczych;  
 int ile\_urzadzen\_w\_gr;  
 int ile\_gr\_urzadzen;  
  
 private int[][][] kolaX = new int[3][3][2]; //0-x 1-y  
  
 @FXML  
 private Rectangle[][] Urzadzenia = new Rectangle[3][3];  
 @FXML  
 private Rectangle[][] StacjeRobocze = new Rectangle[3][3];  
  
 Thread[][] sta = new StacjaRobocza[3][3];  
  
 public void initialize() {  
 ile\_stacji\_roboczych\_w\_gr = 2;  
 ile\_gr\_urzadzen = 2;  
 ile\_urzadzen\_w\_gr = 2;  
 ile\_gr\_stacji\_roboczych = 2;  
 }  
  
 @FXML  
 public void drawing() {  
 int xStacjeDzielenie = 500/(ile\_gr\_stacji\_roboczych+1);  
 int xUrzadzeniaDzielenie = 500/(ile\_gr\_urzadzen+1);  
 int yStacje = 100;  
 int yUrzadzenia = 400;  
 int xlayoutStacje = -40;  
 int xlayoutUrzadzenia = -50;  
 int stala = xStacjeDzielenie;  
 for (int i = 0; i < ile\_gr\_stacji\_roboczych; i++) {  
 for (int j = 0; j < ile\_stacji\_roboczych\_w\_gr; j++) {  
 StacjeRobocze[i][j] = new Rectangle();  
 StacjeRobocze[i][j].setHeight(40);  
 StacjeRobocze[i][j].setWidth(40);  
 StacjeRobocze[i][j].setFill(Color.*LIGHTGRAY*);  
 StacjeRobocze[i][j].setStroke(Color.*BLACK*);  
 StacjeRobocze[i][j].setLayoutX(stala+xlayoutStacje);  
 StacjeRobocze[i][j].setLayoutY(yStacje);  
 animacja.getChildren().add(StacjeRobocze[i][j]);  
 xlayoutStacje+=40;  
 }  
 //xStacjeDzielenie\*=2;  
 stala+=xStacjeDzielenie;  
 }  
 stala=xUrzadzeniaDzielenie;  
  
 for (int i = 0; i < ile\_gr\_urzadzen; i++) {  
 for (int j = 0; j < ile\_urzadzen\_w\_gr; j++) {  
 Urzadzenia[i][j] = new Rectangle();  
 Urzadzenia[i][j].setWidth(50);  
 Urzadzenia[i][j].setHeight(50);  
 Urzadzenia[i][j].setStroke(Color.*BLACK*);  
 Urzadzenia[i][j].setFill(Color.*GRAY*);  
 Urzadzenia[i][j].setLayoutX(stala+xlayoutUrzadzenia);  
 Urzadzenia[i][j].setLayoutY(yUrzadzenia);  
 animacja.getChildren().add(Urzadzenia[i][j]);  
 xlayoutUrzadzenia+=50;  
 }  
 stala+=xUrzadzeniaDzielenie;  
 }  
 }  
  
 @FXML  
 public void onSliderStacjeRoboczeChanged() {  
 int sliderValue = (int) ile\_stacji\_roboczych\_w\_grupie.getValue();  
 textpierw.setText("Ile stacji roboczych: "+sliderValue);  
 ile\_stacji\_roboczych\_w\_gr = sliderValue;  
 System.*out*.println(sliderValue + " "); // zbieranie informacji ze slajdera 1  
 //txtOut.setText(escritoresQuant.getValue()+" ");  
 }  
  
 @FXML  
 public void onSliderGrupyStacjiRoboczychChanged() {  
 int sliderValue = (int) ile\_grup\_stacji\_roboczych.getValue();  
 ile\_gr\_stacji\_roboczych = sliderValue;  
 textdwa.setText("Ile grup stacji roboczych: "+sliderValue);  
 System.*out*.println(sliderValue + " ");//zbieranie informacji ze slajdera 2  
 }  
  
 @FXML  
 public void onSliderUrzadzeniaChanged() {  
 int sliderValue = (int) ile\_urzadzen\_w\_grupie.getValue();  
 ile\_urzadzen\_w\_gr = sliderValue;  
 texttrzy.setText("Ile urzadzen: "+sliderValue);  
 System.*out*.println(sliderValue + " ");  
 }  
  
 @FXML  
 public void onSliderGrupyUrzadzenChanged() {  
 int sliderValue = (int) ile\_grup\_urzadzen.getValue();  
 ile\_gr\_urzadzen = sliderValue;  
 textcztery.setText("Ile grop urzadzen: "+sliderValue);  
 System.*out*.println(sliderValue + " ");  
 }  
  
 @FXML  
 public void onHelloButtonClick() throws InterruptedException {  
 System.*out*.println("Hello");// zaslepka na start animacji  
 //myThread thr = new myThread(animacja, StacjeRobocze[0][0], Urzadzenia[0][0]);  
 //myThread thrr = new myThread(animacja, StacjeRobocze[0][1], Urzadzenia[0][1]);  
 //thr.start();  
 //thrr.start();  
 Server server = new Server(ile\_urzadzen\_w\_gr, ile\_gr\_urzadzen, ile\_gr\_stacji\_roboczych);//int ile\_urzadzen\_w\_grupie,int ile\_grup\_urzadzen,int ile\_grup\_stacji\_roboczych  
 server.init();  
 /\*Thread w1 = new StacjaRobocza(1,0,server,3, animacja, StacjeRobocze[0][0], Urzadzenia);//int numer, int grupa, Server server, int N, AnchorPane animacja, Rectangle mojaStacja, Rectangle[][] urzadzenia  
 Thread w2 = new StacjaRobocza(2,0,server,3, animacja, StacjeRobocze[0][1], Urzadzenia);  
 Thread w3 = new StacjaRobocza(3,1,server,3, animacja, StacjeRobocze[1][0], Urzadzenia);  
 Thread w4 = new StacjaRobocza(4,1,server,3, animacja, StacjeRobocze[1][1], Urzadzenia);\*/  
 int numer = 1;  
 for (int i = 0; i < ile\_gr\_stacji\_roboczych; i++) {  
 for (int j = 0; j < ile\_stacji\_roboczych\_w\_gr; j++) {  
 sta[i][j] = new StacjaRobocza(numer, i, server, 3, animacja, StacjeRobocze[i][j], Urzadzenia, ile\_gr\_urzadzen, fasterr);  
 numer++;  
 }  
 }  
  
 /\*w1.start();  
 w2.start();  
 w3.start();  
 w4.start();\*/  
  
 for (int i = 0; i < ile\_gr\_stacji\_roboczych; i++) {  
 for (int j = 0; j < ile\_stacji\_roboczych\_w\_gr; j++) {  
 sta[i][j].start();  
 }  
 }  
  
 /\*try {  
 w1.join();  
 w2.join();  
 w3.join();  
 w4.join();  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }\*/  
 System.*out*.println("Koniec\n");  
 }  
  
 int predkosc = 1;  
  
 @FXML  
 public void faster() {  
 if (predkosc%2 == 1) {  
 fasterr = 10;  
 predkosc++;  
 }  
 else  
 {  
 fasterr = 4;  
 predkosc++;  
 }  
 }  
  
 @FXML  
 public void szybciej() {  
 System.*out*.println("Gotowe");  
 ile\_stacji\_roboczych\_w\_grupie.setDisable(true);  
 ile\_grup\_stacji\_roboczych.setDisable(true);  
 ile\_urzadzen\_w\_grupie.setDisable(true);  
 ile\_grup\_urzadzen.setDisable(true);  
 drawing();  
 }  
  
 @FXML  
 public void kasowanie() {  
 for (int i = 0; i < ile\_gr\_stacji\_roboczych; i++) {  
 for (int j = 0; j < ile\_stacji\_roboczych\_w\_gr; j++) {  
 animacja.getChildren().remove(StacjeRobocze[i][j]);  
 }  
 }  
  
 for (int i = 0; i < ile\_gr\_stacji\_roboczych; i++) {  
 for (int j = 0; j < ile\_stacji\_roboczych\_w\_gr; j++) {  
 animacja.getChildren().remove(Urzadzenia[i][j]);  
 }  
 }  
  
 ile\_stacji\_roboczych\_w\_grupie.setDisable(false);  
 ile\_grup\_stacji\_roboczych.setDisable(false);  
 ile\_urzadzen\_w\_grupie.setDisable(false);  
 ile\_grup\_urzadzen.setDisable(false);  
 }  
}

hello-view.fxml

<?xml version="1.0" encoding="UTF-8"?>  
  
<?import javafx.scene.control.Button?>  
<?import javafx.scene.control.RadioButton?>  
<?import javafx.scene.control.Slider?>  
<?import javafx.scene.control.SplitPane?>  
<?import javafx.scene.layout.AnchorPane?>  
<?import javafx.scene.text.Text?>  
  
<SplitPane dividerPositions="0.29797979797979796" maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-Infinity" minWidth="-Infinity" prefHeight="400.0" prefWidth="600.0" xmlns="http://javafx.com/javafx/17" xmlns:fx="http://javafx.com/fxml/1" fx:controller="siec.javafxprojektpw9\_2022\_projekt.HelloController">  
 <items>  
 <AnchorPane fx:id="animacja" maxHeight="1000" maxWidth="1000" minHeight="500" minWidth="500" prefHeight="500" prefWidth="500" />  
 <AnchorPane maxHeight="1000.0" maxWidth="300.0" minHeight="0.0" minWidth="0.0" prefHeight="500.0" prefWidth="90.0">  
 <children>  
 <Button fx:id="startButton" layoutX="45.0" layoutY="416.0" mnemonicParsing="false" onAction="#onHelloButtonClick" prefHeight="30.0" prefWidth="144.0" text="Start" />  
 <Slider fx:id="ile\_stacji\_roboczych\_w\_grupie" blockIncrement="1" layoutX="58.0" layoutY="90.0" majorTickUnit="1" max="3" min="1" minorTickCount="1" onMouseReleased="#onSliderStacjeRoboczeChanged" showTickLabels="true" showTickMarks="true" snapToTicks="true" value="2" /><!-- ile stacji w grupie -->  
 <Slider fx:id="ile\_grup\_stacji\_roboczych" blockIncrement="1" layoutX="58.0" layoutY="162.0" majorTickUnit="1" max="3" min="1" minorTickCount="1" onMouseReleased="#onSliderGrupyStacjiRoboczychChanged" showTickLabels="true" showTickMarks="true" snapToTicks="true" value="2" /><!-- ile grup stacji roboczych 4-->  
 <Slider fx:id="ile\_urzadzen\_w\_grupie" blockIncrement="1" layoutX="58.0" layoutY="223.0" majorTickUnit="1" max="3" min="1" minorTickCount="1" onMouseReleased="#onSliderUrzadzeniaChanged" showTickLabels="true" showTickMarks="true" snapToTicks="true" value="2" /><!-- Ile urzadzen w grupie 4-->  
 <Slider fx:id="ile\_grup\_urzadzen" blockIncrement="1" layoutX="58.0" layoutY="285.0" majorTickUnit="1" max="3" min="1" minorTickCount="1" onMouseReleased="#onSliderGrupyUrzadzenChanged" showTickLabels="true" showTickMarks="true" snapToTicks="true" value="2" /><!-- ile grup urzadzen 4-->  
 <Text fx:id="textpierw" layoutX="45.0" layoutY="73.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Ile stacji roboczych: 2" />  
 <Text fx:id="textdwa" layoutX="45.0" layoutY="139.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Ile grup stacji roboczych: 2" />  
 <Text fx:id="texttrzy" layoutX="45.0" layoutY="207.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Ile urzadzen: 2" />  
 <Text fx:id="textcztery" layoutX="45.0" layoutY="272.0" strokeType="OUTSIDE" strokeWidth="0.0" text="Ile grup urzadzen: 2" />  
 <Button fx:id="szybko" layoutX="49.0" layoutY="350.0" mnemonicParsing="false" onAction="#szybciej" text="Dodaj" />  
 <Button fx:id="kasowanie" layoutX="120" layoutY="350.0" mnemonicParsing="false" onAction="#kasowanie" text="Usun" />  
 <RadioButton fx:id="faster" layoutX="63.0" layoutY="19.0" mnemonicParsing="false" onAction="#faster" text="Szybko" />  
 </children>  
 </AnchorPane>  
 </items>  
</SplitPane>

1. Uwagi końcowe.

HelloApplication.java jest klasą uruchomieniową.

HelloController.java i hello-view.fxml są odpowiedzialne za graficzną część projektu.

Server.java jest monitorem procesów, odpowiedzialnym za synchronizację wątków.

StacjaRobocza.java jest klasą wątków (komputerów) która jest odpowiedzialna za tworzenie animacji i to obiekty tej klasy proszą o dostęp klasę monitora.