Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

**МЕХАНІКО-ТЕХНОЛОГІЧНИЙ ФАКУЛЬТЕТ**

Кафедра кібербезпеки та програмного забезпечення

**Звіт**

з виконаної лабораторної роботи № 6

Дисципліна: Паралельні та розподілені обчислення

на тему

«Розподілені обчислення. Взаємодія паралельних потоків»

Виконав :

студент академічної групи КІ-15

Аннаєв А. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Перевірив :

Викладач

Минайленко Р. М.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кропивницкий- 2018

**Мета:** Одержати навички створення і налагодження додатків для розподілених обчислень.

**Завдання:**

- Використовуючи результати лабораторних робіт № 3 і № 4, 5 здійснити взаємодію   
 між потоками Клієнтської і Серверної програми.

- Реалізувати програмне забезпечення Сервера.

- Реалізувати програмне забезпечення Клієнта.

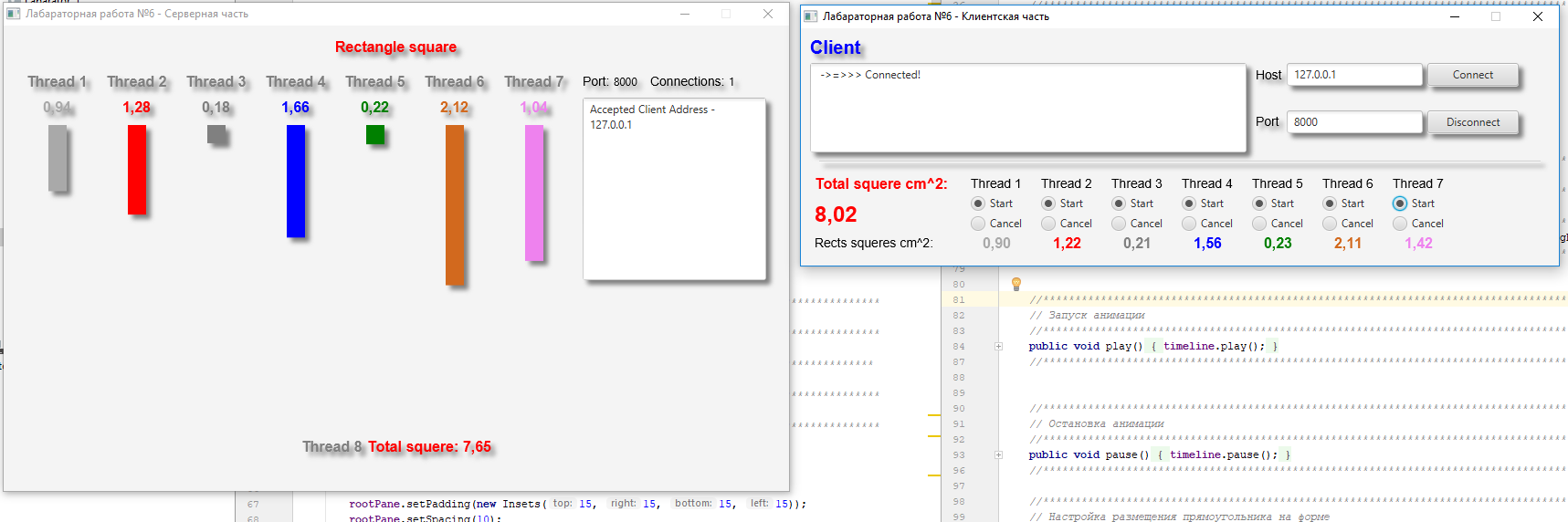
- Здійснити мережне керування потоками розподілених обчислень сторони Сервера з   
 боку Клієнта .

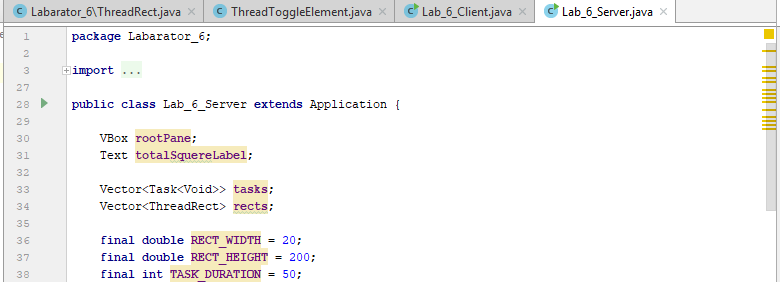
- Забезпечити безперервне обчислення площ прямокутників у паралельних потоках   
 на стороні Сервера.

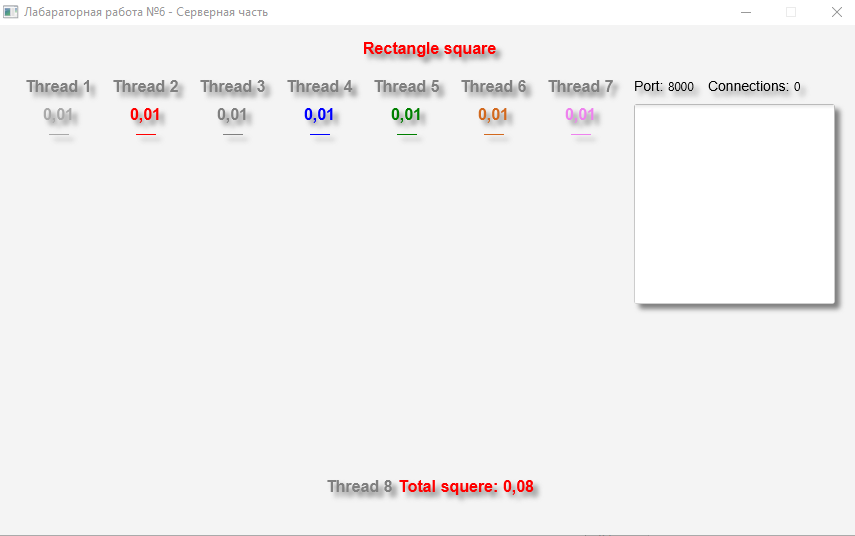
- Забезпечити безперервну передачу результатів обчислень Клієнту

- На стороні Клієнта забезпечити виведення результатів обчислень.

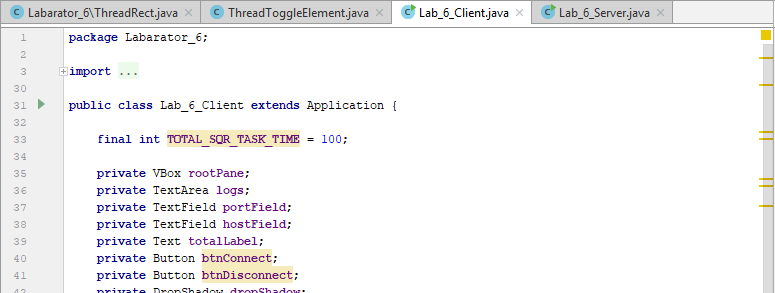
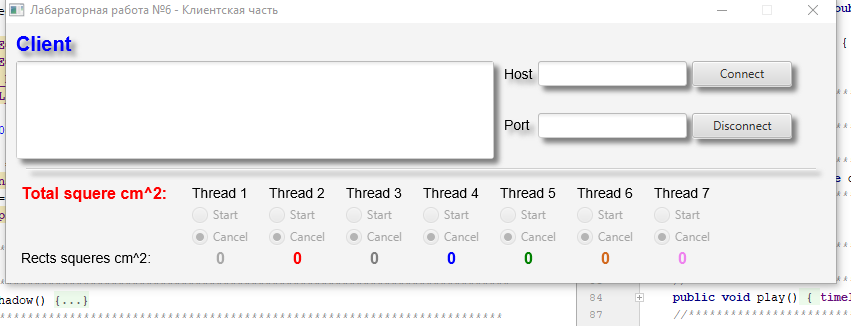
Хід роботи



**package** Labarator\_6;  
  
**public class** Lab\_6\_Server **extends** Application {  
  
 VBox **rootPane**;  
 Text **totalSquereLabel**;  
 Vector<Task<Void>> **tasks**;  
 Vector<ThreadRect> **rects**;  
 **final double RECT\_WIDTH** = 20;  
 **final double RECT\_HEIGHT** = 200;  
 **final int TASK\_DURATION** = 50;  
 **final int TOTAL\_TASK\_DURATION** = 100;  
 **int port** = 8000;  
 Text **portLabel** = **new** Text(**""**);  
 Text **connectionsLabel** = **new** Text(**""**);  
 TextArea **logs** = **new** TextArea();  
 DropShadow **dropShadow**;  
*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Тень  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **private void** shadow() {  
 **dropShadow** = **new** DropShadow();  
 **dropShadow**.setRadius(7.0);  
 **dropShadow**.setOffsetX(5.0); *// Shadow offset (X and Y axis)* **dropShadow**.setOffsetY(5.0);  
 **dropShadow**.setColor(Color.***GRAY***); *// Shadow color* }  
*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Определение отображения контролов  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **private void** createLayout() {  
 **rootPane** = **new** VBox();  
 **totalSquereLabel** = createLabel(**"0.0"**, Color.***RED***);  
 **rootPane**.setPadding(**new** Insets(15, 15, 15, 15));  
 **rootPane**.setSpacing(10);  
 **logs**.setMinHeight(200);  
 **logs**.setWrapText(**true**);  
 HBox headerBox = **new** HBox();  
 HBox centerHBox = **new** HBox();  
 HBox footerHBox = **new** HBox();  
 HBox serverTitleH = **new** HBox();  
 VBox serverV = **new** VBox();  
 Text headerLabel = createLabel(**"Rectangle square"**, Color.***RED***);  
  
 headerBox.setAlignment(Pos.***CENTER***);  
 headerBox.getChildren().addAll(headerLabel);  
addThreadRectangle(centerHBox,  
 **new** ThreadRect(1000, **"Thread 1"**, **RECT\_HEIGHT**, Color.***DARKGREY***),  
 **TASK\_DURATION**);addThreadRectangle(centerHBox,  
 **new** ThreadRect(700, **"Thread 2"**, **RECT\_HEIGHT**, Color.***RED***),  
 **TASK\_DURATION**);addThreadRectangle(centerHBox,  
 **new** ThreadRect(1200, **"Thread 3"**, **RECT\_HEIGHT**, Color.***GREY***),  
 **TASK\_DURATION**);addThreadRectangle(centerHBox,  
 **new** ThreadRect(400, **"Thread 4"**, **RECT\_HEIGHT**, Color.***BLUE***),  
 **TASK\_DURATION**);addThreadRectangle(centerHBox,  
 **new** ThreadRect(2000, **"Thread 5"**, **RECT\_HEIGHT**, Color.***GREEN***),  
 **TASK\_DURATION**);  
  
addThreadRectangle(centerHBox,  
 **new** ThreadRect(3000, **"Thread 6"**, **RECT\_HEIGHT**, Color.***CHOCOLATE***),  
 **TASK\_DURATION**);addThreadRectangle(centerHBox,  
 **new** ThreadRect(100, **"Thread 7"**, **RECT\_HEIGHT**, Color.***VIOLET***),  
 **TASK\_DURATION**);  
  
 **portLabel**.setText(String.*valueOf*(**port**) + **" "**);  
 **connectionsLabel**.setText(**"0"**);  
 Text portTitle = createLabel(**"Port: "**, Color.***BLACK***, FontWeight.***NORMAL***, 14);  
 Text connectionTitle = createLabel(**"Connections: "**, Color.***BLACK***, FontWeight.***NORMAL***, 14);  
 serverTitleH.getChildren().addAll(portTitle, **portLabel**, connectionTitle, **connectionsLabel**);  
 serverTitleH.setPadding(**new** Insets(0, 10, 10, 0));  
 serverV.setPadding(**new** Insets(10, 10, 10, 10));  
 serverV.getChildren().addAll(serverTitleH, **logs**);  
 Text totalThreadLabel = createLabel(**"Thread 8 "**, Color.***GRAY***);  
 Text footerLabel = createLabel(**"Total squere: "**, Color.***RED***);  
 centerHBox.getChildren().add(serverV);  
 footerHBox.getChildren().addAll(totalThreadLabel, footerLabel, **totalSquereLabel**);  
 footerHBox.setAlignment(Pos.***BOTTOM\_CENTER***);  
shadow();  
 headerBox.setEffect(**dropShadow**);  
 centerHBox.setEffect(**dropShadow**);  
 footerHBox.setEffect(**dropShadow**);  
 centerHBox.setMinHeight(2 \* **RECT\_HEIGHT**);  
 **rootPane**.getChildren().addAll(headerBox, centerHBox, footerHBox);  
 }  
 *//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Определение отображения контролов  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **private void** addThreadRectangle(Pane destPane, ThreadRect rect, **int** taskDurationMsecs) {  
 rect.setWidth(**RECT\_WIDTH**);  
 rect.setHeight(1);  
 VBox v = rect.createLayout();destPane.getChildren().addAll(v);  
 Task<Void> t = rect.createSquereCalcTask(taskDurationMsecs);  
 **tasks**.add(t);  
 **rects**.add(rect);  
Thread th = **new** Thread(t);  
 th.setName(**"Rect Thread"**);  
 th.start();  
 }  
*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Отображение текста  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **private** Text createLabel(String text, Color color) {  
 Text t = **new** Text();  
 t.setText(text);  
 t.setFont(Font.*font*(**"Arial"**, FontWeight.***BOLD***, 16));  
 t.setFill(color);  
 **return** t;  
 }  
 *//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Отображение текста  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **private void** createTotalTaskThread() {  
 Task<Void> task = **new** Task<Void>() {  
 @Override  
 **protected** Void call() **throws** Exception {  
 **while** (**true**) {  
 Platform.*runLater*(**new** Runnable() {  
 @Override  
 **public void** run() {  
 **double** s = 0.0f;  
 **for** (ThreadRect r : **rects**) {  
 s += r.getSquere();  
 }  
 **totalSquereLabel**.setText(String.*format*(**"%1$,.2f"**, s));  
 }  
 });  
 Thread.*sleep*(**TOTAL\_TASK\_DURATION**);  
 }  
 }  
 };  
 **tasks**.add(task);  
 Thread th = **new** Thread(task);  
 th.setName(**"Total thread"**);  
 th.start();  
 }  
*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Отображение текста  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****private** Text createLabel(String text, Color color, FontWeight fontWeight, **int** fontSize) {  
 Text t = **new** Text();  
 t.setText(text);  
 t.setFont(Font.*font*(**"Arial"**, fontWeight, fontSize));  
 t.setFill(color);  
 **return** t;  
 }  
*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Точка входа  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **public static void** main(String[] args) { *launch*(args);  
 }  
*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** @Override  
 **public void** start(Stage primaryStage) **throws** Exception {  
  
 **tasks** = **new** Vector<>();  
 **rects** = **new** Vector<>();  
 primaryStage.setTitle(**"Лабараторная работа №6 - Серверная часть"**);  
 primaryStage.setResizable(**false**);  
  
 createLayout();  
 createTotalTaskThread();  
  
 Scene scene = **new** Scene(**rootPane**, 850, 500, Color.***TRANSPARENT***);  
 primaryStage.setScene(scene);  
 primaryStage.show();  
 Server server = **new** Server(**logs**, **connectionsLabel**, **totalSquereLabel**, **port**, **rects**);  
 server.start();  
 primaryStage.setOnCloseRequest(we -> {  
  
 **for** (Task<Void> t : **tasks**) {  
 t.cancel();  
 }  
  
 server.stopServer();  
 });  
 }  
}



**package** Labarator\_6;  
**public class** Lab\_6\_Client **extends** Application {  
  
 **final int TOTAL\_SQR\_TASK\_TIME** = 100;  
  
 **private** VBox **rootPane**;  
 **private** TextArea **logs**;  
 **private** TextField **portField**;  
 **private** TextField **hostField**;  
 **private** Text **totalLabel**;  
 **private** Button **btnConnect**;  
 **private** Button **btnDisconnect**;  
 **private** DropShadow **dropShadow**;  
 **private** Socket **clientSocket**;  
 **private** BufferedReader **in**;  
 **private** PrintWriter **out**;  
  
 **private** String **host**;  
 **private int port**;  
 **private** Vector<ThreadToggleElement> **threadToggles**;  
 **private** Task<Void> **totalSqrTask** = **null**;  
 **boolean bConnected** = **false**;  
  
 *//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Отображение текста  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **private** Text createLabel(String text, Color color, FontWeight fontWeight, **int** fontSize) {  
 Text t = **new** Text();  
 t.setText(text);  
 t.setFont(Font.*font*(**"Arial"**, fontWeight, fontSize));  
 t.setFill(color);  
 **return** t;  
 }  
*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Конструктор  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **private** Task<Void> createTotalSquereGetTask(**int** sleepMsec) {  
 Task<Void> task = **new** Task<Void>() {  
 @Override  
 **protected** Void call() **throws** Exception {  
 **while** (**bConnected**) {  
 Platform.*runLater*(**new** Runnable() {  
 @Override  
 **public void** run() {  
 **try** {  
 **out**.println(**"GET\_TOTAL"**);  
 **out**.flush();  
 String totalVal = **in**.readLine();  
 **totalLabel**.setText(totalVal);  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 }  
 });  
 Thread.*sleep*(sleepMsec);  
 }  
 **return null**;}   
 };  
  
 **return** task;  
 }  
*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Подключение  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **private void** connect() {  
 **bConnected** = **false**;  
 **clientSocket** = **null**;  
 **host** = **hostField**.getText();  
 **port** = Integer.*valueOf*(**portField**.getText());  
  
 **try** {  
 **clientSocket** = **new** Socket(**host**, **port**);  
  
 **in** = **new** BufferedReader(**new** InputStreamReader(**clientSocket**.getInputStream()));  
 **out** = **new** PrintWriter(**new** OutputStreamWriter(**clientSocket**.getOutputStream()));  
  
 **totalSqrTask** = createTotalSquereGetTask(**TOTAL\_SQR\_TASK\_TIME**);  
 Thread th = **new** Thread(**totalSqrTask**);  
 th.setName(**"Rect Total Squere Thread"**);  
 th.start();  
  
 **bConnected** = **true**;  
 } **catch** (UnknownHostException uhe) {  
 *// Server Host unreachable* **logs**.appendText(**"Unknown Host :"** + **host**);  
 **clientSocket** = **null**;  
 } **catch** (IOException ioe) {  
 *// Cannot connect to port on given server host* **logs**.appendText(**"Cant connect to server at "** +  
 String.*valueOf*(**port**) +  
 **". Make sure it is running."**);  
 **clientSocket** = **null**;  
 }  
 }  
*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Тень  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **private void** shadow() {  
 **dropShadow** = **new** DropShadow();  
 **dropShadow**.setRadius(7.0);  
 **dropShadow**.setOffsetX(5.0); *// Shadow offset (X and Y axis)* **dropShadow**.setOffsetY(5.0);  
 **dropShadow**.setColor(Color.***GRAY***); *// Shadow color* }  
*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Определение отображения контролов  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **public void** createLayout() {  
 **rootPane** = **new** VBox();  
 **threadToggles** = **new** Vector<ThreadToggleElement>();  
  
 Text title = createLabel(**"Client"**, Color.***BLUE***, FontWeight.***BOLD***, 20);  
 Text portTitle = createLabel(**"Port"**, Color.***BLACK***, FontWeight.***NORMAL***, 14);  
 Text hostTitle = createLabel(**"Host"**, Color.***BLACK***, FontWeight.***NORMAL***, 14);  
 **logs** = **new** TextArea();  
 **portField** = **new** TextField();  
 **hostField** = **new** TextField();  
 **btnConnect** = **new** Button(**"Connect"**);  
 **btnDisconnect** = **new** Button(**"Disconnect"**);  
  
 **btnConnect**.setMinWidth(100);  
 **btnDisconnect**.setMinWidth(100);  
 **logs**.setMaxHeight(100);  
 **logs**.setWrapText(**true**);  
  
 GridPane topGridPane = **new** GridPane();  
 HBox bottomHPane = **new** HBox();  
 VBox leftVPane = **new** VBox();  
  
 topGridPane.add(title, 0, 0);  
 topGridPane.add(**logs**, 0, 1, 2, 2);  
 topGridPane.add(hostTitle, 3, 1);  
 topGridPane.add(portTitle, 3, 2);  
 topGridPane.add(**hostField**, 4, 1, 2, 1);  
 topGridPane.add(**portField**, 4, 2, 2, 1);  
 topGridPane.add(**btnConnect**, 6, 1, 2, 1);  
 topGridPane.add(**btnDisconnect**, 6, 2, 2, 1);  
  
 Separator sepr = **new** Separator();  
 sepr.setMinHeight(5);  
 sepr.setPadding(**new** Insets(10, 10, 10, 10));  
  
 shadow();  
 topGridPane.setEffect(**dropShadow**);  
 **rootPane**.setEffect(**dropShadow**);  
 sepr.setEffect(**dropShadow**);  
 topGridPane.setHgap(5);  
 topGridPane.setVgap(5);**totalLabel** = createLabel(**""**, Color.***RED***, FontWeight.***BOLD***, 24);  
 Text totalTitle = createLabel(**"Total squere cm^2: "**, Color.***RED***, FontWeight.***BOLD***, 16);  
 Text squeresTitle = createLabel(**"Rects squeres cm^2: "**, Color.***BLACK***, FontWeight.***NORMAL***, 14);  
 leftVPane.getChildren().addAll(totalTitle, **totalLabel**, squeresTitle);  
 leftVPane.setSpacing(10);  
 leftVPane.setPadding(**new** Insets(5, 5, 5, 5));  
 bottomHPane.getChildren().addAll(leftVPane);ThreadToggleElement toggle1 = **new** ThreadToggleElement(**"Thread 1"**, 1, Color.***DARKGRAY***);  
 ThreadToggleElement toggle2 = **new** ThreadToggleElement(**"Thread 2"**, 2, Color.***RED***);  
 ThreadToggleElement toggle3 = **new** ThreadToggleElement(**"Thread 3"**, 3, Color.***GREY***);  
 ThreadToggleElement toggle4 = **new** ThreadToggleElement(**"Thread 4"**, 4, Color.***BLUE***);  
 ThreadToggleElement toggle5 = **new** ThreadToggleElement(**"Thread 5"**, 5, Color.***GREEN***);  
 ThreadToggleElement toggle6 = **new** ThreadToggleElement(**"Thread 6"**, 6, Color.***CHOCOLATE***);  
 ThreadToggleElement toggle7 = **new** ThreadToggleElement(**"Thread 7"**, 7, Color.***VIOLET***);  
  
 bottomHPane.setSpacing(10);  
 bottomHPane.getChildren().addAll(toggle1.getLayout(),  
 toggle2.getLayout(),  
 toggle3.getLayout(),  
 toggle4.getLayout(),  
 toggle5.getLayout(),  
 toggle6.getLayout(),  
 toggle7.getLayout());  
 **threadToggles**.add(toggle1);  
 **threadToggles**.add(toggle2);  
 **threadToggles**.add(toggle3);  
 **threadToggles**.add(toggle4);  
 **threadToggles**.add(toggle5);  
 **threadToggles**.add(toggle6);  
 **threadToggles**.add(toggle7);  
 togglesEnabled(**false**);  
 **rootPane**.setPadding(**new** Insets(10, 10, 10, 10));  
 **rootPane**.getChildren().addAll(topGridPane, sepr, bottomHPane);**btnConnect**.setOnAction(**new** EventHandler<ActionEvent>() {  
 @Override  
 **public void** handle(ActionEvent event) {  
 connect();  
 **if** (**clientSocket** == **null**) {  
 togglesEnabled(**false**);  
 **return**;  
 }  
 togglesEnabled(**true**);  
 toggle1.connect(**clientSocket**);  
 toggle2.connect(**clientSocket**);  
 toggle3.connect(**clientSocket**);  
 toggle4.connect(**clientSocket**);  
 toggle5.connect(**clientSocket**);  
 toggle6.connect(**clientSocket**);  
 toggle7.connect(**clientSocket**);  
 **logs**.appendText(**" ->=>>> Connected!\n"**);  
 }  
 });  
 **btnDisconnect**.setOnAction(**new** EventHandler<ActionEvent>() {  
 @Override  
 **public void** handle(ActionEvent event) {  
 **try** {  
 togglesEnabled(**false**);  
 toggle1.disconnect();  
 toggle2.disconnect();  
 toggle3.disconnect();  
 toggle4.disconnect();  
 toggle5.disconnect();  
 toggle6.disconnect();  
 toggle7.disconnect();  
 **bConnected** = **false**;  
 **logs**.appendText(**" ->=>>> Disconnected!\n"**);  
  
 BufferedReader in = **new** BufferedReader(**new** InputStreamReader(**clientSocket**.getInputStream()));  
 PrintWriter out = **new** PrintWriter(**new** OutputStreamWriter(**clientSocket**.getOutputStream()));  
 **if** (out != **null**) {  
 out.println(**"QUIT"**);  
 out.flush();  
 out.close();  
 }  
 **if** (in != **null**)  
 in.close();  
 **clientSocket**.close();  
 } **catch** (IOException exp) {  
 exp.printStackTrace();  
 } } }); } *//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Отключение  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **private void** togglesEnabled(**boolean** m) {  
 **int** n = **threadToggles**.size();  
 **for** (**int** i = 0; i < n; ++i) {  
 **threadToggles**.elementAt(i).getLayout().setDisable(!m);  
 }  
 }  
*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Точка входа  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **public static void** main(String[] args) { *launch*(args); }  
*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** @Override  
 **public void** start(Stage primaryStage) **throws** Exception {  
  
 primaryStage.setTitle(**"Лабараторная работа №6 - Клиентская часть"**);  
 primaryStage.setResizable(**false**);  
 createLayout();  
 Scene scene = **new** Scene(**rootPane**, 820, 250, Color.***TRANSPARENT***);  
 primaryStage.setScene(scene);  
 primaryStage.setOnCloseRequest(**new** EventHandler<WindowEvent>() {  
 @Override  
 **public void** handle(WindowEvent event) {  
 **if** (**totalSqrTask** != **null**)  
 **totalSqrTask**.cancel();  
 }  
 });  
 primaryStage.show();  
 }}



**package** Labarator\_6;  
  
**public class** ThreadToggleElement {  
  
 **final int TASK\_TIME** = 100;  
  
 VBox **main**;  
 Text **titleLabel**;  
 Text **valueLabel**;  
 **int threadID** = -1;  
 BufferedReader **in**;  
 PrintWriter **out**;  
 RadioButton **rb\_1**;  
 RadioButton **rb\_2**;  
 **boolean bConnected** = **false**;  
 Task<Void> **squereTask** = **null**;  
  
 *//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Геттеры и Сеттеры  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **public** VBox getLayout() { **return main**; }  
 **public** Text getValueLabel() { **return valueLabel**; }  
 *//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Конструктор  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **public** ThreadToggleElement(String title, **int** threadID, Color valColor) {  
 *//this.socket = clientSocket;* **this**.**titleLabel** = createLabel(title, Color.***BLACK***, FontWeight.***NORMAL***, 14);  
 **this**.**valueLabel** = createLabel(**"0"**, valColor, FontWeight.***BOLD***, 16);  
 **this**.**threadID** = threadID;  
 createLayout();  
 }  
*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Настройка размещения  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **private void** createLayout() {  
 **main** = **new** VBox();  
 ToggleGroup tg = **new** ToggleGroup();  
 **rb\_1** = **new** RadioButton(**"Start"**);  
 **rb\_2** = **new** RadioButton(**"Cancel"**);  
 **rb\_1**.setToggleGroup(tg);  
 **rb\_2**.setToggleGroup(tg);  
 tg.selectToggle(**rb\_2**);  
 **main**.setSpacing(5);  
 **main**.setPadding(**new** Insets(5, 5, 5, 5));  
 HBox h = **new** HBox();  
 h.getChildren().add(**valueLabel**);  
 h.setAlignment(Pos.***CENTER***);  
 **main**.getChildren().addAll(**titleLabel**, **rb\_1**, **rb\_2**, h);  
 }  
*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Подключение  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **public void** connect(Socket socket) {  
 **bConnected** = **true**;  
 **try** {  
 **in** = **new** BufferedReader(**new** InputStreamReader(socket.getInputStream()));  
 **out** = **new** PrintWriter(**new** OutputStreamWriter(socket.getOutputStream()));  
 } **catch** (IOException e) {  
 System.***out***.println(**"Cannot connect to server."**);  
 }  
  
 **if** (**rb\_1** == **null** || **rb\_2** == **null**)  
 **return**;  
  
 **rb\_1**.setOnAction((ActionEvent evn) -> {  
 **out**.println(**"ON"**);  
 **out**.println(**threadID**);  
 **out**.flush();  
 });  
  
 **rb\_2**.setOnAction((ActionEvent evn) -> {  
 **out**.println(**"OFF"**);  
 **out**.println(**threadID**);  
 **out**.flush();  
 });  
  
**squereTask** = createSquereGetTask(**TASK\_TIME**);  
 Thread th = **new** Thread(**squereTask**);  
 th.setName(**"Rect Squere Thread"**);  
 th.start();  
 }  
*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Отключение  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **public void** disconnect() {  
 **bConnected** = **false**;  
 **if** (**squereTask** != **null**)  
 **squereTask**.cancel();  
 }  
*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Отображение текста  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **private** Text createLabel(String text, Color color, FontWeight fontWeight, **int** fontSize) {  
 Text t = **new** Text();  
 t.setText(text);  
 t.setFont(Font.*font*(**"Arial"**, fontWeight, fontSize));  
 t.setFill(color);  
 **return** t;  
 }  
*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
 //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
// Настройка размещения  
//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\** **private** Task<Void> createSquereGetTask(**int** sleepMsec) {  
 Task<Void> task = **new** Task<Void>() {  
 @Override  
 **protected** Void call() **throws** Exception {  
 **while** (**bConnected**) {  
 Platform.*runLater*(**new** Runnable() {  
 @Override  
 **public void** run() {  
 **try** {  
 **out**.println(**"GET"**);  
 **out**.println(**threadID**);  
 **out**.flush();  
 String val = **in**.readLine();  
 **valueLabel**.setText(val);  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 **bConnected** = **false**;  
 } }  
 });  
 Thread.*sleep*(sleepMsec);  
 }  
 **return null**; }  
 }; **return** task;  
 }  
*//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**}