Продолжение приложения А

} catch (SQLException ex) {

Logger.getLogger(Database.class.getName()).log(Level.SEVERE, null, ex);

}

}

public void update(String sqlString) {

try {

statemnt.executeUpdate(sqlString);

} catch (SQLException ex) {

Logger.getLogger(Database.class.getName()).log(Level.SEVERE, null, ex);

}

}

public ResultSet select(String sqlString) {

ResultSet rs = null;

try {

rs = statemnt.executeQuery(sqlString);

} catch (SQLException ex) {

Logger.getLogger(Database.class.getName()).log(Level.SEVERE, null, ex);

}

return rs;

}

public void close() {

try {

connect.close();

statemnt.close();

} catch (SQLException ex) {

Logger.getLogger(Database.class.getName()).log(Level.SEVERE, null, ex);

}

}

}

Метод для построения графика:

private void ButtonGraphActionPerformed(java.awt.event.ActionEvent evt){

BarChartDialog dlg = new BarChartDialog(this, true);

dlg.setLocationRelativeTo(null);

dlg.setVisible(true);

Stream server = new Stream();

String executeStr = "SELECT `ГодЗаключения`, `Депозит`.`Наименование`, COUNT(`ГодЗаключения`) AS Количество FROM `Договор` "

+ "INNER JOIN `Депозит` ON `Договор`.`КодВклада`=`Депозит`.`Код`";

if (!dlg.getDeposite().equals("")) {

executeStr += " WHERE Наименование='" + dlg.getDeposite() + "'";

Продолжение приложения А

}

if (!dlg.getSince().equals("")) {

executeStr += " AND ГодЗаключения >= " + dlg.getSince();

}

if (!dlg.getFor().equals("")) {

executeStr += " AND ГодЗаключения <= " + dlg.getFor();

}

executeStr += " GROUP BY `ГодЗаключения` ORDER BY `ГодЗаключения`;";

server.sendInt(14);

server.sendString(executeStr);

try {

int count = server.getInt();

DefaultCategoryDataset dataSet = new DefaultCategoryDataset();

for (int i = 0; i < count; i++) {

String[] data = server.getString().split(":");

dataSet.setValue(Integer.parseInt(data[1]), data[0], data[0]);

}

JFreeChart chart = ChartFactory.createBarChart3D("Диаграмма о количестве заключенных договоров", "Год", "Количество",

dataSet, PlotOrientation.VERTICAL, true, true, true);

CategoryPlot P = (CategoryPlot) chart.getPlot();

ChartFrame frame = new ChartFrame("Количество по годам", chart);

frame.setVisible(true);

frame.setSize(700, 700);

} catch (IOException ex) {

Logger.getLogger(MainAdminForm.class.getName()).log(Level.SEVERE, null, ex);

}

}

Метод рассылки писем на E-mail:

private void SendToEmailButtonActionPerformed(java.awt.event.ActionEvent evt){

EmailSendDialog dlg = new EmailSendDialog(this, true);

dlg.setLocationRelativeTo(null);

dlg.setVisible(true);

Sender tlsSender = new Sender("snopok.lika081@gmail.com", "Vjhb\_Jufq17.02");

tlsSender.send(dlg.getSubject(),dlg.getText(), "snopok.lika081@gmail.com", dlg.getRecipients());

}

Класс фильтра для строк:

public class DigitFilter extends DocumentFilter {

private static final String DIGITS = "\\d+";

Продолжение приложения А

@Override

public void insertString(FilterBypass fb, int offset, String string, AttributeSet attr) throws BadLocationException {

if (string.matches(DIGITS)) {

super.insertString(fb, offset, string, attr);

}

}

@Override

public void replace(FilterBypass fb, int offset, int length, String string, AttributeSet attrs) throws BadLocationException {

if (string.matches(DIGITS)) {

super.replace(fb, offset, length, string, attrs);

}

}

}

Класс паттерна Фабрика:

public class TableModelFactory {

public MyTableModels createModel(TypeOfModel type, List<Object> list) {

MyTableModels model = null;

switch (type) {

case CLIENT\_TABLE:

model = new ClientTableModel(list);

break;

case DEPOSITE\_TABLE:

model = new DepositeTableModel(list);

break;

case CONTRACT\_TABLE:

model = new ContractTableModel(list);

break;

case CURRENCY\_TABLE:

model = new CurrencyTableModel(list);

break;

case USER\_TABLE:

model = new UserTableModel(list);

break;

}

return model;

}

}