Министерство образования и науки Российской Федерации

**Уральский федеральный университет**

**имени первого Президента России Б. Н. Ельцина**

**Факультет ускоренного обучения**

Кафедра программных средств и систем

УДК 004.431.42

Оценка проекта

Члены комиссии

**КУРСОВАЯ РАБОТА**

**по дисциплине «Объектно-ориентированное программирование»**

**Тема: Разработка программы «Учёт работ по подготовке данных»**

Студент гр. СП-652 Коробейников К.В.

Руководитель Суханов В.И.

**Екатеринбург 2014**

Уральский федеральный университет

имени первого Президента России Б. Н. Ельцина

Факультет ускоренного обучения

УТВЕРЖДАЮ

Зав.кафедрой ПСС

“ “ 201\_\_ г.

ЗАДАНИЕ

на курсовую работу по дисциплине

«Объектно-ориентированное программирование»

Студент группы СП-6\_\_\_\_ Специальность ПО ВТ и АС

Фамилия\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Имя\_\_\_\_\_\_\_\_\_\_ Отчество\_\_\_\_\_\_\_\_\_\_\_\_

Срок проектирования с “\_\_” \_\_\_\_\_ по “ \_\_ “ \_\_\_\_\_\_\_\_ 200\_\_ г.

1. Тема работы и ее содержание:

**Разработка приложений в среде Java**

Разработать приложение средствами Java для учёта работ по подготовке данных. Предусмотреть использование компонентов: Menu, Label, Edit, Table, Image, Panel (для зонирования форм), Button, OpenDialog, SaveDialog. Исходные данные для работ вводить на первой закладке приложения, по команде меню или кнопки иметь возможность сохранить в файле или прочитать из файла, имя которого запрашивать у пользователя. Записанные данные отправлять по нажатию кнопки в таблицу на второй закладке. Результат учета работ отображать в виде графика на третьей закладке, и предоставить возможность печати графика на принтере. Разработать службу **встроенной помощи**.

2. Дополнительные сведения: назначение приложения можно выбрать по усмотрению студента, связав его с потребностями предприятия. Пояснительная записка должна содержать описание задачи, структуры выходных и входных данных, алгоритм управления приложением, распечатки экранных форм и программных модулей.

3. Курсовое проектирование закончено \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4. Оценка проекта \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_ (Суханов В.И.)

Содержание

[ВВЕДЕНИЕ 4](#_Toc405723984)

[1 Описание программы 5](#_Toc405723985)

[1.1 Закладка «Ввод работы» 5](#_Toc405723986)

[1.2 Закладка «Таблица работ» 8](#_Toc405723987)

[1.3 Закладка «График» 9](#_Toc405723988)

[1.4 Закладка «Неполадки» 10](#_Toc405723989)

[2 Описание алгоритмов 12](#_Toc405723990)

[ЗАКЛЮЧЕНИЕ 14](#_Toc405723991)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 15](#_Toc405723992)

[ПРИЛОЖЕНИЕ А (обязательное) Исходный код программы 16](#_Toc405723993)

ВВЕДЕНИЕ

Программа позволяет вести учёт работ по подготовке данных. Для этого у пользователя есть четыре закладки и меню с разделами и подразделами. Под работой подразумевается ввод данных пользователем о какой-либо организации. Первая закладка служит для непосредственного ввода данных пользователем и записи их в таблице на второй закладке. Вторая закладка служит для хранения данных об организациях в табличном виде. Также здесь есть кнопки для записи табличных данных в файл, и для чтения этих же данных из файла. На третьей закладке формируется график. Этот график формируется на основе табличных данных из второй закладки. График отображает, сколько было работ за конкретную дату. Четвертая закладка служит информирования пользователя об ошибках, которые он мог допустить при заполнении полей на первой закладке. Разделы и подразделы являются командами, которые позволяют выполнить различные действия.

В программе также присутствует справка, где рассказано о каждой закладке, для чего она и что за элементы на ней находятся, а также рассказано о меню программы. Сама справка является разделом меню.

1 Описание программы

Программа состоит из четырех закладок и главного меню. Каждая закладка и раздел меню служат для определенной цели.

1.1 Закладка «Ввод работы»

Данная закладка предназначена для записи данных об организации. На закладке располагается семь полей для ввода и одна кнопка. Каждое поле является одним из реквизитов вводимой организации. Опишем эти поля:

1. Наименование организации – в это поле записывается наименование организации.
2. ИНН – записывается идентификационный номер налогоплательщика.
3. Вид деятельности – записывается, каким видом деятельности занимается организация.
4. Адрес – записывается адрес организации.
5. Контактное лицо – записывается лицо, с которым необходимо связаться.
6. Телефон – записывается телефон контактного лица.
7. Дата – записывается дата, когда была заведена организация.

Кроме полей на закладке есть кнопка «Записать», которая позволяет, после того, как все поля заполнены, записать данные в таблицу на следующей закладке.

С этой закладкой связаны три подраздела меню. Первый подраздел это «Открыть» в разделе «Файлы». Он позволяет открыть нужный файл и передать данные из этого файла в поля на этой закладке. Диалоговое окно этого подраздела показано на рисунке 1.

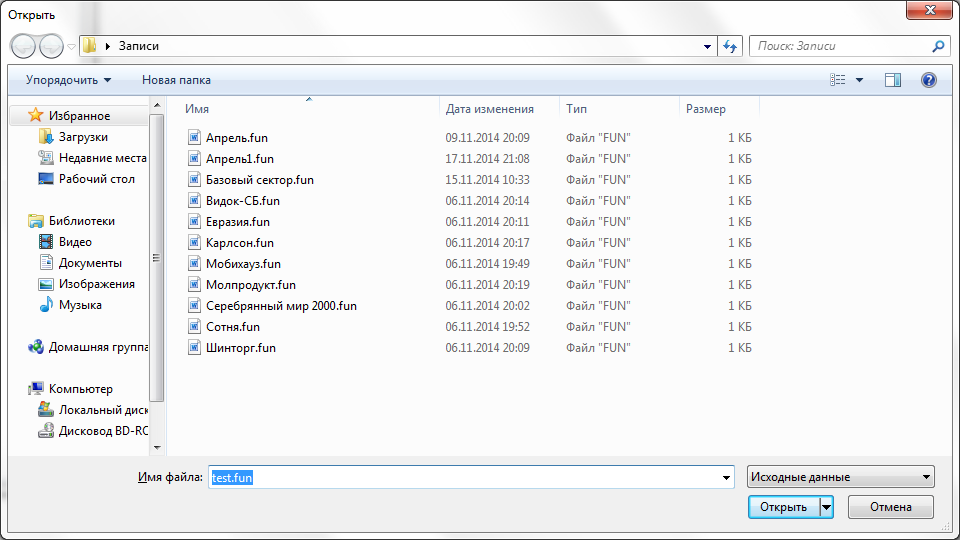


Рисунок 1 – Диалог открытия файла

Второй подраздел этого раздела называется «Сохранить» также связан с этой закладкой. Он позволяет проделать обратную операцию, т.е. передать данные из заполненных полей в выбранный файл. Диалоговое окно этого подраздела показано на рисунке 2.

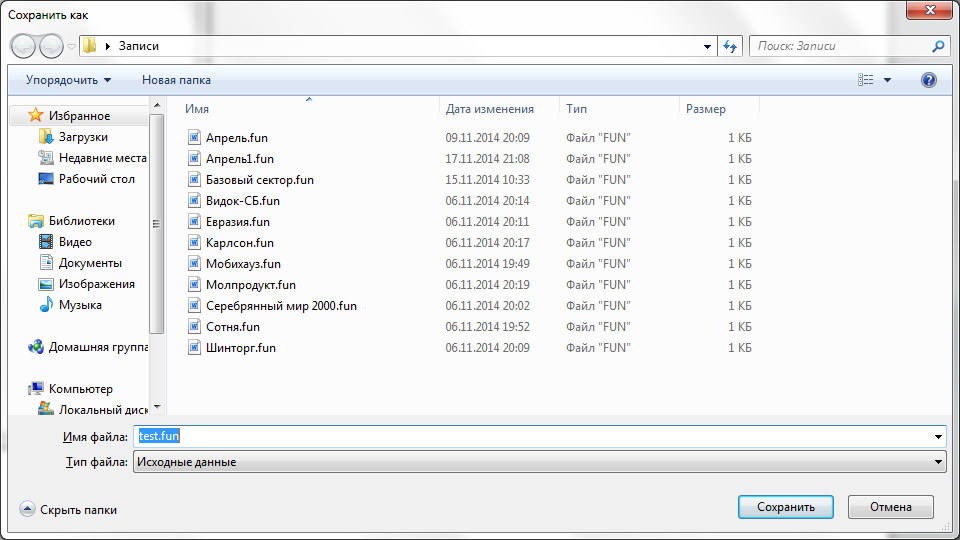


Рисунок 2 – Диалог сохранения файла

Третий подраздел называется «Проверить введенные данные», он находится в разделе «Функции». Данный подраздел позволяет проверить правильность заполнения полей. Правила заполнения полей приведены ниже:

1. Наименование организации – поле не должно быть пустым.
2. ИНН – допускается ввод только цифр и в количестве ровно десять штук, не больше и не меньше.
3. Вид деятельности – допускается ввод только букв и поле не должно быть пустым.
4. Адрес – поле не должно быть пустым.
5. Контактное лицо – допускается ввод только букв и поле не должно быть пустым.
6. Телефон – допускается ввод по определенной маске, т.е. сначала идут три цифры, затем тире, потом еще две цифры, затем еще раз тире и в конце еще две цифры. Поле не должно быть пустым.
7. Дата – заполняется по маске: две цифры, точка, две цифры, точка, четыре цифры. Поле не должно быть пустым.

Если хоть одно из полей будет заполнено не корректно, то внизу формы высветится сообщение «Есть ошибки!». Такой пример показан на рисунке 3.

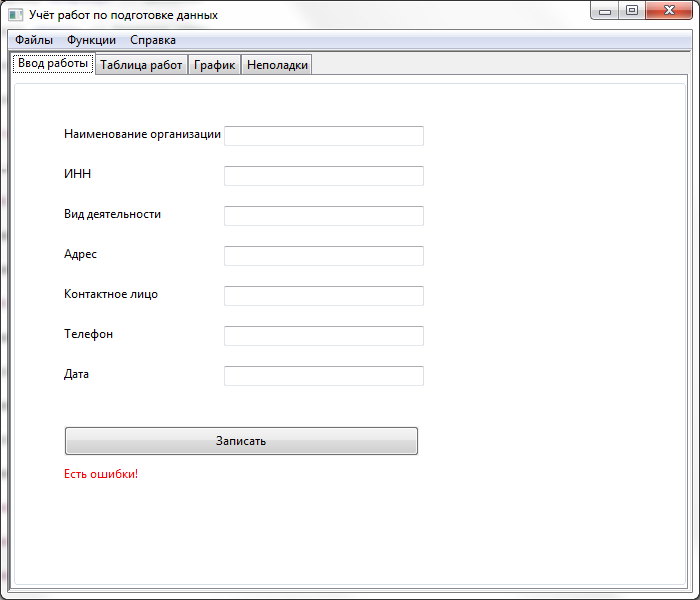


Рисунок 3 – Вывод сообщения об ошибке

Сообщения об ошибках выводятся на четвертой закладке «Неполадки». Если все поля заполнены правильно, то программа выдаст сообщение «Все верно!». Пример показан на рисунке 4.

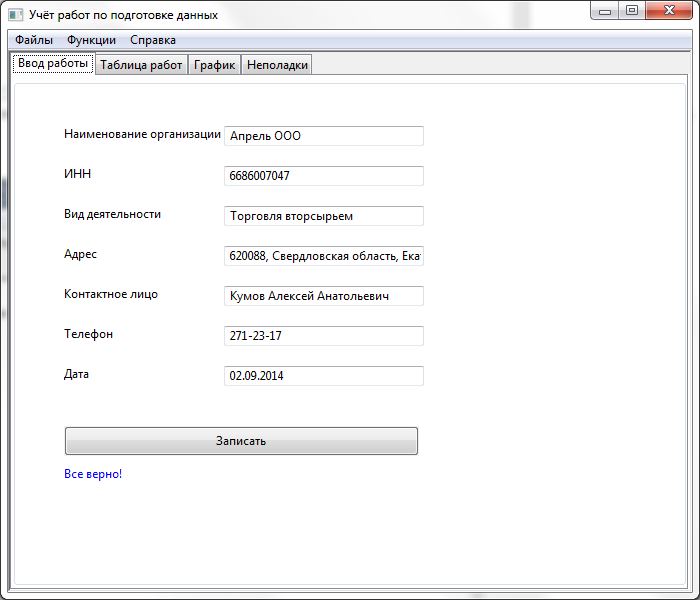


Рисунок 4 – Вывод сообщения о правильном заполнении полей

1.2 Закладка «Таблица работ»

Вторая закладка предназначена для хранения табличных данных об организациях. На ней располагается ода большая таблица, где представлены эти данные и две кнопки, предназначенные для обработки этих данных. Одна строка в таблице соответствует одной организации. Каждое поле этой строки также соответствует одному из полей на предыдущей закладке. Также таблица работ служит для формирования графика на следующей закладке.

Кроме таблицы на закладке есть две кнопки: «Записать таблицу в CSV файл» и «Записать в таблицу из CSV файла». Первая кнопка служит для загрузки табличных данных в файл формата CSV, а вторая служит обратной задачи, т.е. записывает данные в таблицу из CSV файла. Данная закладка показана на рисунке 5.

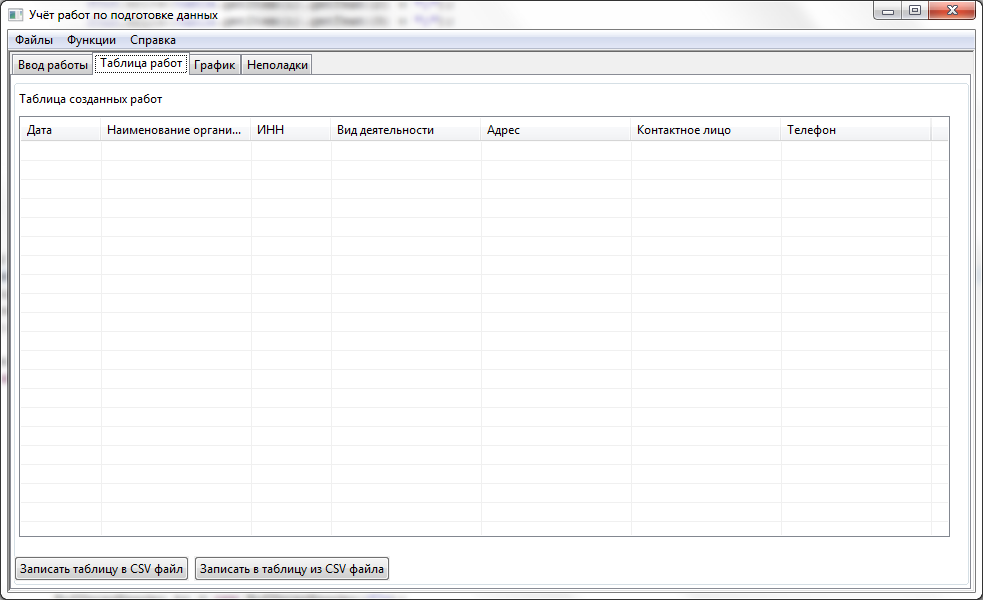


Рисунок 5 – Закладка «Таблица работ»

Диалоги обоих кнопок не имеет смысл показывать, т.к. они похожи на диалоговые окна из предыдущего раздела, т.е. на подразделы «Открыть» и «Сохранить». Различие состоит лишь в формате файла. Добавлять данные в таблицу можно двумя способами: с помощью кнопки «Записать» на предыдущей закладке, в этом случае в таблицу добавляется одна запись или можно воспользоваться кнопкой «Записать в таблицу из CSV файла» и тогда в таблицу может добавиться то количество строк, которое записано в файле. Подразделов из главного меню, связанных непосредственно с этой закладкой нет. Пример заполненной таблицы показан на рисунке 6.

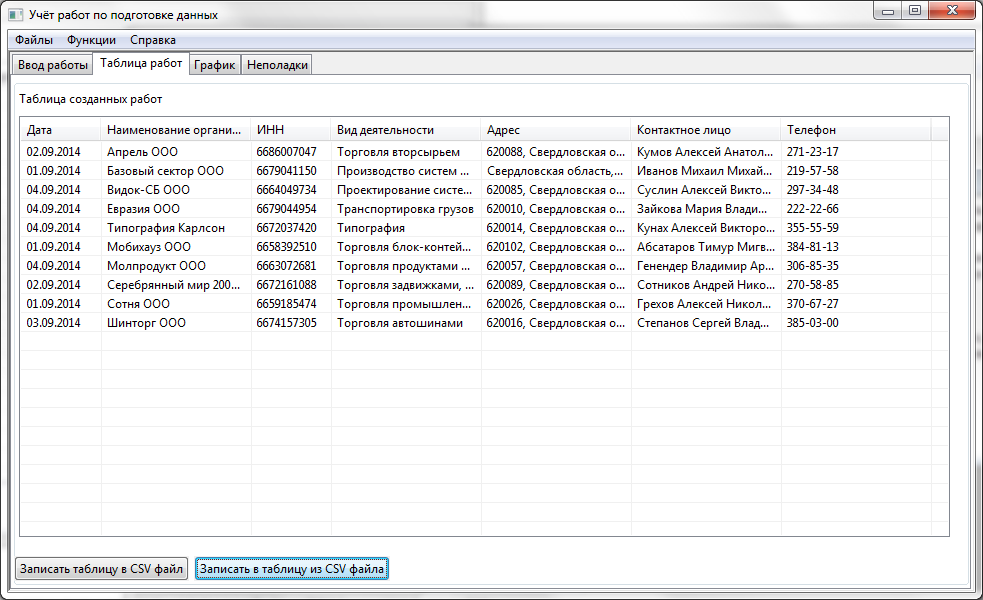


Рисунок 6 – Заполненная таблица работ

1.3 Закладка «График»

Третья закладка предназначена для формирования графика исходя из табличных данных, представленных на предыдущей закладке. Если таблица на второй закладке не заполнена, то график не отобразится, т.е. третья закладка будет пустой, на ней нет элементов кроме графика. Если же в таблице есть несколько записей, то при переходе на третью закладку будет построен график в соответствии с записями. График показывает, сколько было заведено организаций за конкретную дату, т.е. по измерению X отображаются даты, а по измерению Y отображается количество записей за эту дату. Пример построенного графика показан на рисунке 7.

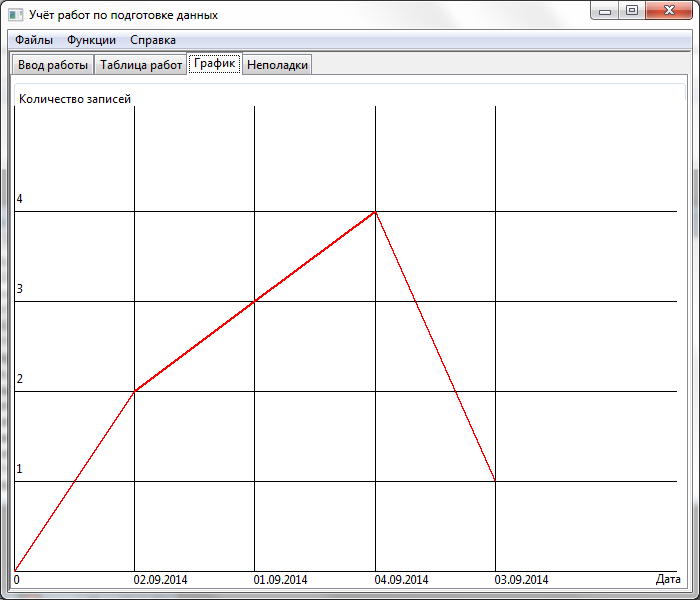


Рисунок 7 – Построенный график

С данной закладкой непосредственно связан лишь подраздел «Напечатать график» из раздела «Функции». Этот подраздел позволяет напечатать получившийся график на принтере, который стоит в системе по умолчанию.

1.4 Закладка «Неполадки»

Четвертая закладка предназначена для отображения пользователю сообщений об ошибке. Она состоит из одного многострочного текстового поля, куда выводятся все сообщения об ошибках. В сообщении указывается, в каком поле была совершена ошибка. Проверка заполнения, как было указано выше, проверяется помощью подраздела «Проверить введенные данные» раздела «Функции». Если все поля заполнены правильно, то многострочное поле будет пусто, в противном случае там будут указаны все ошибки. Пример с большим количеством ошибок показан на рисунке 8.

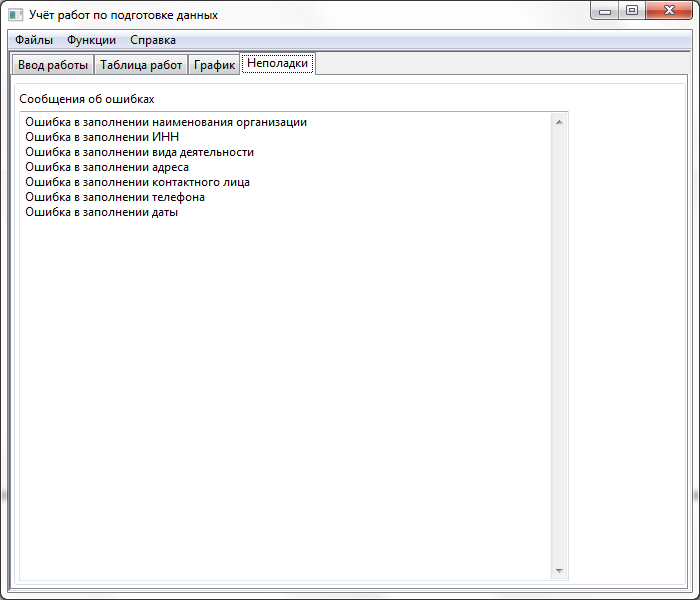


Рисунок 8 – Многострочное поле с ошибками

В главном меню, кроме упомянутых выше разделов и подразделов, есть еще подраздел «Выход» в разделе «Файлы» и раздел «Справка» не содержащий разделов. Подраздел «Выход» служит для выхода из программы. При нажатии на него программа закрывается. Раздел «Справка» служит для отображения пользователю справки по данной программе. Справка содержит краткую информацию обо всех закладках и пунктах главного меню программы. Пример справочного окна показан на рисунке 9.

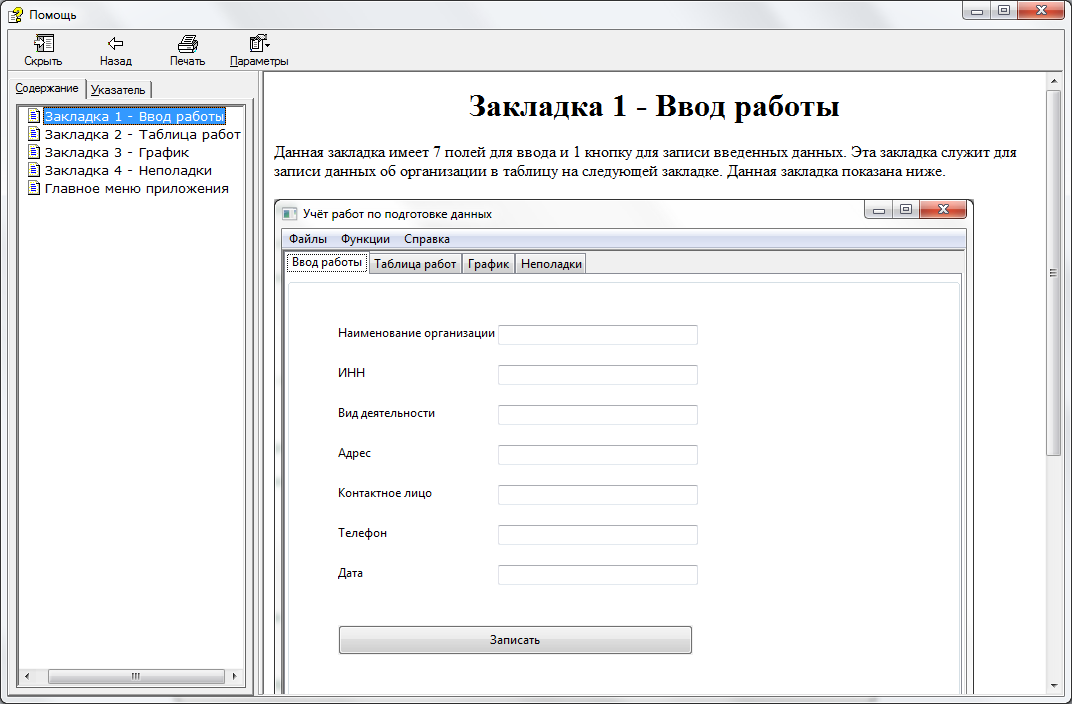


Рисунок 9 – Окно справки

2 Описание алгоритмов

При начале запуска программы создается окно (компонент shell). Для данного окна устанавливается размер по умолчанию, указывается его название. После этого создается главное меню, настраиваются характеристики этого меню. Потом в меню создаются разделы и подразделы. Для каждого подраздела пишется свой обработчик события. Событие в данном случае это нажатие на подраздел или раздел.

Рассмотрим каждый обработчик более подробно. Первый обработчик подраздела «Открыть». В нём создается диалоговое окно для открытия файла. В этом диалоговом окне указываются, какие типы данных может выбрать пользователь, указывается путь файла по умолчанию (обычно это папка user в корневом каталоге) и также указывается имя файла по умолчанию. После того как все данные указаны диалоговое окно открывается, создается поток считывания, после идет попытка считывания строк из файла и записи считанных строк в поля на первой закладке. В конце поток считывания данных закрывается. Если считывание не удалось, то пользователю выводится сообщение об ошибке считывания из файла.

Второй обработчик для подраздела «Сохранить». В нем также создается диалоговое окно, но только для сохранения и также задаются все параметры. Также открывается окно и идет попытка с помощью потока записи записать данные из полей в указанный файл. В конце записи поток закрывается. Если попытка записи не удалась, то пользователю выдается сообщение об ошибке записи в файл.

Третий обработчик для подраздела «Выход». В нем осуществляется выход из программы пользователем.

Четвертый обработчик для подраздела «Проверить введенные данные». Здесь осуществляется ряд проверок. Сначала проверяется поле «Наименование организации» на пустоту строки, если строка пустая, то пользователю отобразится сообщение об ошибке в данном поле. Потом проверяется поле «ИНН» на количество символов, т.е. оно должно быть равно 10, чтобы не было ошибки. Следующее поле «Вид деятельности» проверяется на пустоту и на тип вводимых символов. Если это буквы, то все правильно, иначе будет выдано сообщение об ошибке. Четвертое поле «Адрес» проверяется только на пустоту строки. Пятое поле «Контактное лицо» проверяется точно также как и третье, т.е. идет проверка на пустоту строки и чтобы вводить только буквы. Шестое поле «Телефон» проверяется по специальной маске ###-##-##, где знак # является цифрой. Седьмое поле «Дата» также проверяется по маске ##.##.####, где знак # также является цифрой. В шестом и седьмом полях также проводится проверка на пустоту строки. После всех проверок пользователю высвечивается сообщение: если ошибок нет, то сообщение будет следующим «Все верно!» и будет окрашено в синий цвет, если ошибки есть, то сообщение будет содержать текст «Есть ошибки!» и будет окрашено в красный цвет.

Пятый обработчик для подраздела «Напечатать график». Вначале идет проверка, подключен ли к компьютеру какой-либо принтер. Если подключен, обработчик выполняется дальше, иначе он выводит сообщение об ошибке. После создается объект «Принтер». Если принтер запущен, то дальше идет настройка характеристик графика, т.е. задается цвет фона, цвет линий, устанавливаются границы отображения графика на бумаге. После всех заданных параметров рисуется сам график. Сначала определяются названия по осям X и Y. Позже вычисляется масштаб для каждой оси, определяется положение осей, шаг сетки по этим осям. В конце вычисляется каждая координата и рисуется линия.

Шестой и последний обработчик в меню предназначен для раздела «Справка». Данный обработчик запускает программу справки.

После всех обработчиков идет настройка первой закладки, т.е. создается семь полей для ввода и одна кнопка, указывается их местоположение. При нажатии кнопки создается новая запись в таблице с ранее введенными данными. На второй закладке создается таблица с семью полями и две кнопки. В первой кнопке записан алгоритм записи данных в файл. Во второй алгоритм извлечения данных из файла.

Далее идет формирование третьей закладки. Здесь установлен обработчик перерисовки графика. Данный обработчик повторяет почти полностью обработчик подраздела «Напечатать график», за исключением определения границ для печати на бумаге и создания объекта «Принтер». Потом идет формирование четвертой, последней закладки. Здесь указывается создание многострочного текстового поля, его размеры и и его координаты.

После формирования всех вкладок окно открывается, а если оно закрывается, то освобождаются ресурсы операционной системы.

ЗАКЛЮЧЕНИЕ

Программа предназначена для ведения учета работ, а точнее созданных пользователем организаций. Программа будет использоваться операторами отдела телемаркетинга для своих основных задач.

В результате проделанной работы была разработана программа в соответствии с требованиями задания на курсовую работу. В процессе работы были разработаны и реализованы интерфейсы, необходимые алгоритмы, главная программа и обработчики событий, позволяющие сделать программу быстрой и эффективной в работе. Интерфейсы приложения позволяют неподготовленному пользователю решать стоящие перед ним задачи без дополнительного обучения. Программа имеет встроенную помощь, которая легко включается в самой программе, по основным функциям и интерфейсам.

Данная программа является несовершенной и поэтому может быть доработана по следующим пунктам:

1. Необходимо добавить возможность блокировки определенных символов для полей ввода на первой закладке, к примеру, для поля «ИНН» запретить ввод букв.
2. Реализовать заполнение номера телефона и даты по специальной маске.
3. Реализовать в таблице возможность изменения, удаления и сортировки данных самим пользователем.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Суханов В.И. Программирование графических интерфейсов на языке Java с использованием библиотеки SWT [Электронный ресурс] –Екатеринбург: Изд.УГТУ-УПИ, 2007. –272 с.
2. 2 Тимошенко С.И. Лекции по языку Java [Электронный ресурс] –Екатеринбург: Изд.УГТУ-УПИ, 2007. –213 с.
3. 3 Оформление курсовых и дипломных проектов: Методические указания [Текст] / В.Н.Кичигин, И.У.Мясников, С.И.Тимошенко. –Екатеринбург: Изд. ИПК УГТУ, 2004. – 80 с.
4. 4 Разработка встроенной помощи: Методические указания [Текст] / В.И.Суханов. –Екатеринбург: Изд. ИПК УГТУ, 2001. – 32 с.

ПРИЛОЖЕНИЕ А

(обязательное)

Исходный код программы

Код программы:

**package** K2;

**import** org.eclipse.swt.\*;

**import** org.eclipse.swt.graphics.\*;

**import** org.eclipse.swt.widgets.\*;

**import** org.eclipse.swt.layout.\*;

**import** org.eclipse.swt.program.\*;

**import** org.eclipse.swt.printing.\*;

**import** java.util.\*;

**import** java.io.\*;

**public** **class** MainProg {

**static** Display *display*; // Дисплей

**static** Shell *shell*; // Главное окно

**static** Text *nameOrg*; // Имя организации

**static** Text *INN*; // ИНН

**static** **long** *INNTest*;

**static** Text *Deyatelnost*; // Вид деятельности

**static** **int** *DeyatelnostTest*;

**static** Text *Address*; // Адрес организации

**static** Text *Contact*; // Контактное лицо

**static** **int** *ContactTest*;

**static** Text *Tel*; // Телефон

**static** String *TelTest*;

**static** Text *Data*; // Дата записи

**static** String *DataTest*;

**static** Text *errText*; // Для сообщений

**static** **boolean** *isError* = **false**;

**static** **boolean** *isVerify* = **false**;

**static** **boolean** *isTable* = **false**;

**static** **boolean** *isGraph* = **false**;

**static** Label *errLabel*;

**static** Table *table*; // Таблица значений полинома

**static** TableColumn *columnOrg*; // Столбцы таблицы

**static** TableColumn *columnINN*;

**static** TableColumn *columnDeyatelnost*;

**static** TableColumn *columnAddress*;

**static** TableColumn *columnContact*;

**static** TableColumn *columnTel*;

**static** TableColumn *columnData*;

**static** String *fileName*; // Имя файла для сохранения

**static** FileWriter *fOut*; // Файл для записи

**static** FileReader *fIn*; // Файл для чтения

**static** Group *grpGr*; // Группа элементов для построения графика

**static** **int** *penW* = 1; // Параметры пера для рисования

**static** **int** *penG* = 2;

**static** Listener *onHelp* = **new** Listener() {

**public** **void** handleEvent(Event e) {

// Загрузить файл справки приложением по расширению html

// Program.launch("C:\\eclipse\\karjer.hlp");

Program.*launch*("HelpKursovaya2.chm");

}

};

**static** Listener *onPrint* = **new** Listener() {

// Обработка команды меню "Печать графика"

**public** **void** handleEvent(Event e) {

PrinterData data = Printer.*getDefaultPrinterData*();

**if** (data == **null**) {

MessageBox msg = **new** MessageBox(*shell*, SWT.*ICON\_QUESTION*

| SWT.*OK*);

msg

.setMessage("Предупреждение: Не определен принтер по умолчанию");

msg.open();

**return**;

}

// Создать объект - принтер

Printer printer = **new** Printer(data);

**if** (printer.startJob("Graphic\_1")) {

Color black = printer.getSystemColor(SWT.*COLOR\_BLACK*);

Color white = printer.getSystemColor(SWT.*COLOR\_WHITE*);

Color red = printer.getSystemColor(SWT.*COLOR\_RED*);

Rectangle bnd = printer.getBounds();

Point dpi = printer.getDPI();

**int** leftMargin = dpi.x / 2 + bnd.x; // 1/2 дюйма слева от

// границы бумаги

**int** topMargin = dpi.y / 2 + bnd.y; // 1/2 дюйма сверху от

// границы бумаги

GC gc = **new** GC(printer); // Получили графический контекст 130

**if** (printer.startPage()) { // Принтер готов печатать страницу

gc.setBackground(white); // 136

gc.setForeground(black);

// Определение количества дат и количества записей за

// конкретную дату

**int** k = 0;

String[] S = **new** String[*table*.getItemCount()];

**boolean** pravda = **false**;

**int**[] Kolichestvo = **new** **int**[*table*.getItemCount()];

**for** (**int** i = 0; i < *table*.getItemCount(); i++) {

**for** (**int** j = 0; j < S.length; j++) {

**if** (*table*.getItem(i).getText(0).equals(S[j])) {

pravda = **true**;

**break**;

} **else**

pravda = **false**;

}

**if** (pravda == **false**) {

S[k] = *table*.getItem(i).getText(0);

k++;

}

}

**for** (**int** j = 0; S[j] != **null**; j++)

**for** (**int** i = 0; i < *table*.getItemCount(); i++) {

**if** (*table*.getItem(i).getText(0).equals(S[j]))

Kolichestvo[j] += 1;

}

**double** minY = 0;

**double** maxY = Kolichestvo.length;

**double** minX = 1;

**double** maxX = S.length;

**int** w = bnd.width - 10 - dpi.x;

**int** h = bnd.height - 15 - dpi.y;

**double** scaleX = w / ((maxX - minX) \* 1.10); // Масштаб по Х

**double** scaleY = h / ((maxY - minY) \* 1.10); // Масштаб по Y

**int** x0 = (**int**) Math.*floor*((0 - minX) \* scaleX) + leftMargin; // Положение

// оси

// Y

**int** y0 = (**int**) Math.*floor*(h - (0 - minY) \* scaleY)

+ topMargin; // Положение оси Х

gc.setLineWidth(*penW*);

gc

.drawText("Дата", w - 20 + leftMargin, h - 10

+ topMargin); // Имя оси Х

gc.drawText("Количество записей", 5 + leftMargin,

15 + topMargin); // Имя оси Y

**double** dx = (maxX - minX) / 5.0; // Шаг сетки по Х

**double** dy = (maxY - minY) / 5.0; // Шаг сетки по Y

**for** (**int** i = 0; i < k + 1; i++) { // Рисуем шкалы и сетку

**double** x = Math.*floor*(minX + dx \* i); // Координата X

// исходная

**int** x1 = (**int**) Math.*floor*((x - minX) \* scaleX)

+ leftMargin; // Координата X на экране

**if** (i == 0)

gc.drawText("0", x1, h + 1 + topMargin); // Шкала

// по Х

**else**

gc.drawText(S[i - 1], x1, h + 1 + topMargin);

gc.drawLine(x1, topMargin, x1, h + topMargin); // Сетка

// по Х

**double** y = Math.*floor*(minY + dy \* i); // Координата Y

// исходная

**int** y1 = (**int**) Math.*floor*(h - (y - minY) \* scaleY)

+ topMargin; // Координата Y на экране

**if** (i == 0)

gc.drawText(" ", 3 + leftMargin, y1 - 20); // Шкала

// по Y

gc.drawText(Integer.*toString*(i), 3 + leftMargin,

y1 - 20);

gc.drawLine(leftMargin, y1, leftMargin + w, y1); // Сетка

// по Y

}

;

// Рисуем график

**double** x2 = Math.*floor*(minX + dx \* 0);

x0 = (**int**) Math.*floor*((x2 - minX) \* scaleX) + leftMargin;// Задаем

// начальную

// точку

**double** y2 = Math.*floor*(minY + dy \* 0);

y0 = (**int**) Math.*floor*(h - (y2 - minY) \* scaleY) + topMargin;

gc.setLineWidth(*penG*); // Перо для графика

gc.setForeground(red); // Цвет для графика

**for** (**int** i = 1; i < k + 1; i++) { // По точкам графика

**double** x = Math.*floor*(minX + dx \* (i));

**int** x1 = (**int**) Math.*floor*((x - minX) \* scaleX)

+ leftMargin;

**double** y = Math.*floor*(minY + dy \* Kolichestvo[i - 1]); // Координата

// исходная

**int** y1 = (**int**) Math.*floor*(h - (y - minY) \* scaleY)

+ topMargin;

gc.drawLine(x0, y0, x1, y1);

x0 = x1;

y0 = y1; // Меняем начальную точку

}

;

printer.endPage();

}

gc.dispose(); // Освободили графический контекст

printer.endJob(); // Закрыли зададание на печать

}

printer.dispose();

}

};

**static** Listener *onExit* = **new** Listener() {

// Обработка команды меню "Выход"

**public** **void** handleEvent(Event e) {

*shell*.close();

}

};

**static** Listener *onOpen* = **new** Listener() {

// Обработка команды меню "Открыть файл"

**public** **void** handleEvent(Event e) {

FileDialog dialog = **new** FileDialog(*shell*, SWT.*OPEN*); // 85

dialog.setFilterNames(**new** String[] { "Исходные данные",

"Все файлы (\*.\*)" });

dialog.setFilterExtensions(**new** String[] { "\*.fun", "\*.\*" }); // Windows

// wild

// cards

dialog.setFilterPath("c:\\user\\"); // Windows path

dialog.setFileName("test.fun");

*fileName* = dialog.open();

**try** {

*fIn* = **new** FileReader(*fileName*);

BufferedReader br = **new** BufferedReader(*fIn*);

*nameOrg*.setText(br.readLine());

*INN*.setText(br.readLine());

*Deyatelnost*.setText(br.readLine());

*Address*.setText(br.readLine());

*Contact*.setText(br.readLine());

*Tel*.setText(br.readLine());

*Data*.setText(br.readLine());

*fIn*.close();

} **catch** (IOException eio) {

*errText*.append("Ошибка при чтении из файла " + *fileName* + "\n");

*isError* = **true**;

}

}

};

**static** Listener *onSave* = **new** Listener() {

// Обработка команды меню "Сохранить файл"

**public** **void** handleEvent(Event e) {

FileDialog dialog = **new** FileDialog(*shell*, SWT.*SAVE*); // 85

dialog.setFilterNames(**new** String[] { "Исходные данные",

"Все файлы (\*.\*)" });

dialog.setFilterExtensions(**new** String[] { "\*.fun", "\*.\*" }); // Windows

// wild

// cards

dialog.setFilterPath("c:\\user\\"); // Windows path

dialog.setFileName("test.fun");

*fileName* = dialog.open();

**try** {

*fOut* = **new** FileWriter(*fileName*);

*fOut*.write(*nameOrg*.getText() + "\n");

*fOut*.write(*INN*.getText() + "\n");

*fOut*.write(*Deyatelnost*.getText() + "\n");

*fOut*.write(*Address*.getText() + "\n");

*fOut*.write(*Contact*.getText() + "\n");

*fOut*.write(*Tel*.getText() + "\n");

*fOut*.write(*Data*.getText() + "\n");

*fOut*.close();

} **catch** (IOException eio) {

*errText*.append("Ошибка при записи в файл " + *fileName* + "\n");

*isError* = **true**;

}

}

};

**static** Listener *onRevise* = **new** Listener() {

// Обработка команды меню "Проверить данные"

**public** **void** handleEvent(Event e) {

*errText*.setText(""); // 45

*isError* = **false**;

**if** (*nameOrg*.getText().length() == 0) {

*errText*

.append("Ошибка в заполнении наименования организации \n"); // 42

*isError* = **true**;

}

**try** {

*INNTest* = Long.*parseLong*(*INN*.getText());

**if** (*INN*.getText().length() != 10) {

*errText*.append("Ошибка в заполнении ИНН\n");

*isError* = **true**;

}

} **catch** (RuntimeException e1) {

*errText*.append("Ошибка в заполнении ИНН\n"); // 42

*isError* = **true**;

}

;

**try** {

**if** (*Deyatelnost*.getText().length() == 0) {

*errText*.append("Ошибка в заполнении вида деятельности \n"); // 42

*isError* = **true**;

}

*DeyatelnostTest* = Integer.*parseInt*(*Deyatelnost*.getText());

*errText*.append("Ошибка в заполнении вида деятельности \n"); // 42

*isError* = **true**;

} **catch** (RuntimeException e1) {

}

;

**if** (*Address*.getText().length() == 0) {

*errText*.append("Ошибка в заполнении адреса \n"); // 42

*isError* = **true**;

}

**try** {

**if** (*Contact*.getText().length() == 0) {

*errText*.append("Ошибка в заполнении контактного лица \n"); // 42

*isError* = **true**;

}

*ContactTest* = Integer.*parseInt*(*Contact*.getText());

*errText*.append("Ошибка в заполнении контактного лица \n"); // 42

*isError* = **true**;

} **catch** (RuntimeException e1) {

}

;

**try** {

*TelTest* = *Tel*.getText();

**if** (*TelTest*.charAt(3) != '-' | *TelTest*.charAt(6) != '-'

| *TelTest*.length() > 9) {

*errText*.append("Ошибка в заполнении телефона \n"); // 42

*isError* = **true**;

}

String a = *TelTest*.substring(0, 3);

String b = *TelTest*.substring(4, 6);

String c = *TelTest*.substring(7, 9);

Integer.*parseInt*(a);

Integer.*parseInt*(b);

Integer.*parseInt*(c);

} **catch** (RuntimeException e1) {

*errText*.append("Ошибка в заполнении телефона \n"); // 42

*isError* = **true**;

}

;

**try** {

*DataTest* = *Data*.getText();

**if** (*DataTest*.charAt(2) != '.' | *DataTest*.charAt(5) != '.'

| *DataTest*.length() > 10) {

*errText*.append("Ошибка в заполнении даты \n"); // 42

*isError* = **true**;

}

String a = *DataTest*.substring(0, 2);

String b = *DataTest*.substring(3, 5);

String c = *DataTest*.substring(6, 9);

Integer.*parseInt*(a);

Integer.*parseInt*(b);

Integer.*parseInt*(c);

} **catch** (RuntimeException e1) {

*errText*.append("Ошибка в заполнении даты \n"); // 42

*isError* = **true**;

}

**if** (!*isError*) {

*errLabel*.setForeground(*display*.getSystemColor(SWT.*COLOR\_BLUE*));

*errLabel*.setText("Все верно!");

*isVerify* = **true**;

} **else** {

*errLabel*.setForeground(*display*.getSystemColor(SWT.*COLOR\_RED*));

*errLabel*.setText("Есть ошибки!");

*isVerify* = **false**;

}

;

*isTable* = **false**;

*isGraph* = **false**;

}

};

**static** Listener *onGraph* = **new** Listener() {

// Обработка запроса на перерисовку графика

**public** **void** handleEvent(Event e) {

**if** ((*isGraph*) & (e.type == 9)) {// Был ли график построен

GC gc;

**if** (e.type != 9) {

gc = **new** GC(*grpGr*);

}// Для полотна графической группы блокнота

**else** {

gc = (GC) e.gc;

}

;// Для области перерисовки

Color black = *display*.getSystemColor(SWT.*COLOR\_BLACK*);

Color white = *display*.getSystemColor(SWT.*COLOR\_WHITE*);

Color red = *display*.getSystemColor(SWT.*COLOR\_RED*);

gc.setBackground(white);

gc.setForeground(black);

gc.fillRectangle(*grpGr*.getBounds());

**int** k = 0;

String[] S = **new** String[*table*.getItemCount()];

**boolean** pravda = **false**;

**int**[] Kolichestvo = **new** **int**[*table*.getItemCount()];

**for** (**int** i = 0; i < *table*.getItemCount(); i++) {

**for** (**int** j = 0; j < S.length; j++) {

**if** (*table*.getItem(i).getText(0).equals(S[j])) {

pravda = **true**;

**break**;

} **else**

pravda = **false**;

}

**if** (pravda == **false**) {

S[k] = *table*.getItem(i).getText(0);

k++;

}

}

**for** (**int** j = 0; S[j] != **null**; j++)

**for** (**int** i = 0; i < *table*.getItemCount(); i++) {

**if** (*table*.getItem(i).getText(0).equals(S[j]))

Kolichestvo[j] += 1;

}

**double** minY = 0;

**double** maxY = Kolichestvo.length;

**double** minX = 0;

**double** maxX = S.length;

**int** w = *grpGr*.getBounds().width - 10;

**int** h = *grpGr*.getBounds().height - 15;

**double** scaleX = w / ((maxX - minX) \* 1.10); // Масштаб по Х

**double** scaleY = h / ((maxY - minY) \* 1.10); // Масштаб по Y

**int** x0 = (**int**) Math.*floor*((0 - minX) \* scaleX); // Положение оси

// Y

**int** y0 = (**int**) Math.*floor*(h - (0 - minY) \* scaleY); // Положение

// оси Х

gc.setLineWidth(*penW*);

gc.drawText("Дата", w - 20, h);// Имя оси Х

gc.drawText("Количество записей", 5, 15); // Имя оси Y

**double** dx = (maxX - minX) / 5.0; // Шаг сетки по Х

**double** dy = (maxY - minY) / 5.0; // Шаг сетки по Y

**for** (**int** i = 0; i < k + 1; i++) {// Рисуем шкалы и сетку

**double** x = Math.*floor*(minX + dx \* i); // Координата

// исходная

**int** x1 = (**int**) Math.*floor*((x - minX) \* scaleX); // Координата

// на экране

**if** (i == 0)

gc.drawText("0", x1, h + 1); // Шкала по Х

**else**

gc.drawText(S[i - 1], x1, h + 1);

gc.drawLine(x1, 30, x1, h); // Сетка по Х

**double** y = Math.*floor*(minY + dy \* i); // Координата

// исходная

**int** y1 = (**int**) Math.*floor*(h - (y - minY) \* scaleY);// Координата

// на

// экране

**if** (i == 0)

gc.drawText(" ", 3, y1 - 20); // Шкала по Y

**else**

gc.drawText(Integer.*toString*(i), 3, y1 - 20);

gc.drawLine(0, y1, w, y1); // Сетка по Y

}

;

// Рисуем график

**double** x2 = Math.*floor*(minX + dx \* 0);

x0 = (**int**) Math.*floor*((x2 - minX) \* scaleX);// Задаем начальную

// точку

**double** y2 = Math.*floor*(minY + dy \* 0);

y0 = (**int**) Math.*floor*(h - (y2 - minY) \* scaleY);

gc.setLineWidth(*penG*); // Перо для графика

gc.setForeground(red); // Цвет для графика

**for** (**int** i = 1; i < k + 1; i++) {// По точкам графика

**double** x = Math.*floor*(minX + dx \* (i));

**int** x1 = (**int**) Math.*floor*((x - minX) \* scaleX);

**double** y = Math.*floor*(minY + dy \* Kolichestvo[i - 1]); // Координата

// исходная

**int** y1 = (**int**) Math.*floor*(h - (y - minY) \* scaleY);

gc.drawLine(x0, y0, x1, y1);

x0 = x1;

y0 = y1;// Меняем начальную точку

}

;

**if** (e.type == 13) {

gc.dispose();

}

;

}

;

}

};

**public** **static** **void** main(String[] args) {

// Главный метод приложения

*display* = **new** Display();

*shell* = **new** Shell(*display*);

*shell*.setLayout(**new** FillLayout());

*shell*.setText("Учёт работ по подготовке данных");

// Размеры окна приложения

*shell*.setSize(700, 600);

// Главное меню приложения

Menu bar = **new** Menu(*shell*, SWT.*BAR*); // Область Главного меню

*shell*.setMenuBar(bar);

MenuItem fileItem = **new** MenuItem(bar, SWT.*CASCADE*);

fileItem.setText("Файлы"); // Группа меню

Menu subFile = **new** Menu(*shell*, SWT.*DROP\_DOWN*);

fileItem.setMenu(subFile);

MenuItem itemOpen = **new** MenuItem(subFile, SWT.*PUSH*);

itemOpen.setText("Открыть");

itemOpen.addListener(SWT.*Selection*, *onOpen*);

MenuItem itemSave = **new** MenuItem(subFile, SWT.*PUSH*);

itemSave.setText("Сохранить");

itemSave.addListener(SWT.*Selection*, *onSave*);

MenuItem itemExit = **new** MenuItem(subFile, SWT.*PUSH*);

itemExit.setText("Выход");

itemExit.addListener(SWT.*Selection*, *onExit*);

MenuItem funItem = **new** MenuItem(bar, SWT.*CASCADE*);

funItem.setText("Функции"); // Группа

Menu subFun = **new** Menu(*shell*, SWT.*DROP\_DOWN*);

funItem.setMenu(subFun);

MenuItem itemRevise = **new** MenuItem(subFun, SWT.*PUSH*);

itemRevise.setText("Проверить введенные данные");

itemRevise.addListener(SWT.*Selection*, *onRevise*);

MenuItem itemGraph = **new** MenuItem(subFun, SWT.*PUSH*);

itemGraph.setText("Напечатать график");

itemGraph.addListener(SWT.*Selection*, *onPrint*);

MenuItem itemHelp = **new** MenuItem(bar, SWT.*PUSH*);

itemHelp.setText("Справка");

itemHelp.addListener(SWT.*Selection*, *onHelp*);

// Блокнот с закладками

**final** TabFolder tabFolder = **new** TabFolder(*shell*, SWT.*BORDER*);

// Закладка "Исходные данные"

TabItem tabInp = **new** TabItem(tabFolder, SWT.*NONE*);

tabInp.setText("Ввод работы");

// Создали группу для элементов на закладке

Group grpInp = **new** Group(tabFolder, SWT.*SHADOW\_IN*);

// Константы для размещения элементов ввода исходных данных

**final** **int** inp\_leftLabel = 50;

**final** **int** inp\_wLabel = 160;

**final** **int** inp\_leftText = 210;

**final** **int** inp\_wText = 200;

**final** **int** inp\_topLabel = 50;

**final** **int** inp\_hText = 20;

**final** **int** inp\_vertStep = 40;

// Поле ввода названия организации

Label lName = **new** Label(grpInp, SWT.*LEFT*);

lName.setText("Наименование организации");

lName.setLocation(inp\_leftLabel, inp\_topLabel);

lName.setSize(inp\_wLabel, inp\_hText);

*nameOrg* = **new** Text(grpInp, SWT.*BORDER* | SWT.*SINGLE*);

*nameOrg*.setLocation(inp\_leftText, inp\_topLabel);

*nameOrg*.setSize(inp\_wText, inp\_hText);

// Поле ввода ИНН

Label lINN = **new** Label(grpInp, SWT.*LEFT*);

lINN.setText("ИНН");

lINN.setLocation(inp\_leftLabel, inp\_topLabel + 1 \* inp\_vertStep);

lINN.setSize(inp\_wLabel, inp\_hText);

*INN* = **new** Text(grpInp, SWT.*BORDER* | SWT.*SINGLE*);

*INN*.setLocation(inp\_leftText, inp\_topLabel + 1 \* inp\_vertStep);

*INN*.setSize(inp\_wText, inp\_hText);

// Поле ввода вида деятельности

Label lDeyatelnost = **new** Label(grpInp, SWT.*LEFT*);

lDeyatelnost.setText("Вид деятельности");

lDeyatelnost

.setLocation(inp\_leftLabel, inp\_topLabel + 2 \* inp\_vertStep);

lDeyatelnost.setSize(inp\_wLabel, inp\_hText);

*Deyatelnost* = **new** Text(grpInp, SWT.*BORDER* | SWT.*SINGLE*);

*Deyatelnost*.setLocation(inp\_leftText, inp\_topLabel + 2 \* inp\_vertStep);

*Deyatelnost*.setSize(inp\_wText, inp\_hText);

// Поле ввода адреса

Label lAddress = **new** Label(grpInp, SWT.*LEFT*);

lAddress.setText("Адрес");

lAddress.setLocation(inp\_leftLabel, inp\_topLabel + 3 \* inp\_vertStep);

lAddress.setSize(inp\_wLabel, inp\_hText);

*Address* = **new** Text(grpInp, SWT.*BORDER* | SWT.*SINGLE*);

*Address*.setLocation(inp\_leftText, inp\_topLabel + 3 \* inp\_vertStep);

*Address*.setSize(inp\_wText, inp\_hText);

// Поле ввода контактного лица

Label lContact = **new** Label(grpInp, SWT.*LEFT*);

lContact.setText("Контактное лицо");

lContact.setLocation(inp\_leftLabel, inp\_topLabel + 4 \* inp\_vertStep);

lContact.setSize(inp\_wLabel, inp\_hText);

*Contact* = **new** Text(grpInp, SWT.*BORDER* | SWT.*SINGLE*);

*Contact*.setLocation(inp\_leftText, inp\_topLabel + 4 \* inp\_vertStep);

*Contact*.setSize(inp\_wText, inp\_hText);

// Поле ввода телефона

Label lTel = **new** Label(grpInp, SWT.*LEFT*);

lTel.setText("Телефон");

lTel.setLocation(inp\_leftLabel, inp\_topLabel + 5 \* inp\_vertStep);

lTel.setSize(inp\_wLabel, inp\_hText);

*Tel* = **new** Text(grpInp, SWT.*BORDER* | SWT.*SINGLE*);

*Tel*.setLocation(inp\_leftText, inp\_topLabel + 5 \* inp\_vertStep);

*Tel*.setSize(inp\_wText, inp\_hText);

// Поле ввода даты

Label lData = **new** Label(grpInp, SWT.*LEFT*);

lData.setText("Дата");

lData.setLocation(inp\_leftLabel, inp\_topLabel + 6 \* inp\_vertStep);

lData.setSize(inp\_wLabel, inp\_hText);

*Data* = **new** Text(grpInp, SWT.*BORDER* | SWT.*SINGLE*);

*Data*.setLocation(inp\_leftText, inp\_topLabel + 6 \* inp\_vertStep);

*Data*.setSize(inp\_wText, inp\_hText);

// Создание кнопки Записать

Button button = **new** Button(grpInp, SWT.*LEFT*);

button.setText("Записать");

button.setAlignment(SWT.*CENTER*);

button.setLocation(inp\_leftLabel, inp\_topLabel + 7 \* inp\_vertStep + 20);

button.setSize(355, 30);

button

.addSelectionListener(**new** org.eclipse.swt.events.SelectionAdapter() {

**public** **void** widgetSelected(

org.eclipse.swt.events.SelectionEvent e) {

TableItem item = **new** TableItem(*table*, SWT.*NONE*);

item.setText(0, *Data*.getText());

item.setText(1, *nameOrg*.getText());

item.setText(2, *INN*.getText());

item.setText(3, *Deyatelnost*.getText());

item.setText(4, *Address*.getText());

item.setText(5, *Contact*.getText());

item.setText(6, *Tel*.getText());

*isTable* = **true**;

*isGraph* = **true**;

}

});

*errLabel* = **new** Label(grpInp, SWT.*LEFT*);

*errLabel*.setLocation(inp\_leftLabel, inp\_topLabel + 8 \* inp\_vertStep

+ 20);

*errLabel*.setSize(inp\_wLabel, inp\_hText);

// Разместили группу на закладке

tabInp.setControl(grpInp);

// ============= Закладка для таблицы =================

TabItem tabTab = **new** TabItem(tabFolder, SWT.*NONE*);

tabTab.setText("Таблица работ");

Group grpTbl = **new** Group(tabFolder, SWT.*SHADOW\_IN*);

Label l2 = **new** Label(grpTbl, SWT.*LEFT*);

l2.setText("Таблица созданных работ");

l2.setLocation(5, 15);

l2.setSize(145, 20);

*table* = **new** Table(grpTbl, SWT.*MULTI* | SWT.*BORDER* | SWT.*FULL\_SELECTION*

| SWT.*H\_SCROLL* | SWT.*V\_SCROLL*);

*table*.setLinesVisible(**true**);

*table*.setHeaderVisible(**true**);

*columnData* = **new** TableColumn(*table*, SWT.*NONE*);

*columnData*.setText("Дата"); // Заголовок столбца 1

*columnData*.setWidth(80);

*columnOrg* = **new** TableColumn(*table*, SWT.*NONE*);

*columnOrg*.setText("Наименование организации"); // Заголовок столбца 2

*columnOrg*.setWidth(150);

*columnINN* = **new** TableColumn(*table*, SWT.*NONE*);

*columnINN*.setText("ИНН"); // Заголовок столбца 3

*columnINN*.setWidth(80);

*columnDeyatelnost* = **new** TableColumn(*table*, SWT.*NONE*);

*columnDeyatelnost*.setText("Вид деятельности"); // Заголовок столбца 4

*columnDeyatelnost*.setWidth(150);

*columnAddress* = **new** TableColumn(*table*, SWT.*NONE*);

*columnAddress*.setText("Адрес"); // Заголовок столбца 5

*columnAddress*.setWidth(150);

*columnContact* = **new** TableColumn(*table*, SWT.*NONE*);

*columnContact*.setText("Контактное лицо"); // Заголовок столбца 6

*columnContact*.setWidth(150);

*columnTel* = **new** TableColumn(*table*, SWT.*NONE*);

*columnTel*.setText("Телефон"); // Заголовок столбца 7

*columnTel*.setWidth(150);

*table*.setSize(*table*.computeSize(SWT.*DEFAULT*, 400));

*table*.setLocation(5, 40);

Button button1 = **new** Button(grpTbl, SWT.*LEFT*);

button1.setText("Записать таблицу в CSV файл");

button1.setAlignment(SWT.*CENTER*);

button1.setLocation(0, 480);

button1.pack();

button1

.addSelectionListener(**new** org.eclipse.swt.events.SelectionAdapter() {

**public** **void** widgetSelected(

org.eclipse.swt.events.SelectionEvent e) {

FileDialog dialog = **new** FileDialog(*shell*, SWT.*SAVE*); // 85

dialog.setFilterNames(**new** String[] { "Файл CSV",

"Все файлы (\*.\*)" });

dialog.setFilterExtensions(**new** String[] { "\*.csv",

"\*.\*" }); // Windows wild cards

dialog.setFilterPath("c:\\user\\"); // Windows path

dialog.setFileName("test.csv");

*fileName* = dialog.open();

**try** {

*fOut* = **new** FileWriter(*fileName*);

**for** (**int** i = 0; i < *table*.getItemCount(); i++) {

*fOut*.write(*table*.getItem(i).getText(0) + ";");

*fOut*.write(*table*.getItem(i).getText(1) + ";");

*fOut*.write(*table*.getItem(i).getText(2) + ";");

*fOut*.write(*table*.getItem(i).getText(3) + ";");

*fOut*.write(*table*.getItem(i).getText(4) + ";");

*fOut*.write(*table*.getItem(i).getText(5) + ";");

*fOut*.write(*table*.getItem(i).getText(6) + "\n");

}

*fOut*.close();

} **catch** (IOException eio) {

*errText*.append("Ошибка при записи в файл "

+ *fileName* + "\n");

*isError* = **true**;

}

}

});

Button button2 = **new** Button(grpTbl, SWT.*LEFT*);

button2.setText("Записать в таблицу из CSV файла");

button2.setAlignment(SWT.*CENTER*);

button2.setLocation(180, 480);

button2.pack();

button2

.addSelectionListener(**new** org.eclipse.swt.events.SelectionAdapter() {

**public** **void** widgetSelected(

org.eclipse.swt.events.SelectionEvent e) {

FileDialog dialog = **new** FileDialog(*shell*, SWT.*OPEN*); // 85

dialog.setFilterNames(**new** String[] { "Файл CSV",

"Все файлы (\*.\*)" });

dialog.setFilterExtensions(**new** String[] { "\*.csv",

"\*.\*" }); // Windows wild cards

dialog.setFilterPath("c:\\user\\"); // Windows path

dialog.setFileName("test.csv");

*fileName* = dialog.open();

**try** {

String line;

*fIn* = **new** FileReader(*fileName*);

BufferedReader br = **new** BufferedReader(*fIn*);

**while** ((line = br.readLine()) != **null**) {

TableItem item = **new** TableItem(*table*, SWT.*NONE*);

**int** i = 0;

StringTokenizer st = **new** StringTokenizer(line,

"\n;");

**while** (st.hasMoreTokens()) {

item.setText(i, st.nextToken());

i++;

}

}

*fIn*.close();

*isTable* = **true**;

*isGraph* = **true**;

} **catch** (IOException eio) {

*errText*.append("Ошибка при чтении из файла "

+ *fileName* + "\n");

*isError* = **true**;

}

}

});

tabTab.setControl(grpTbl);

// ============== Закладка для графика ================

TabItem tabGr = **new** TabItem(tabFolder, SWT.*NONE*);

tabGr.setText("График");

*grpGr* = **new** Group(tabFolder, SWT.*SHADOW\_IN*);

tabGr.setControl(*grpGr*);

*grpGr*.addListener(SWT.*Paint*, *onGraph*); // Установили обработчик для

// перерисовки графика

// ============= Закладка для отчетов =================

TabItem tabErr = **new** TabItem(tabFolder, SWT.*NONE*);

tabErr.setText("Неполадки");

Group grpErr = **new** Group(tabFolder, SWT.*SHADOW\_IN*);

Label lErr = **new** Label(grpErr, SWT.*LEFT*);

lErr.setText("Сообщения об ошибках");

lErr.setLocation(5, 15);

lErr.setSize(145, 20);

*errText* = **new** Text(grpErr, SWT.*BORDER* | SWT.*MULTI* | SWT.*V\_SCROLL*);

*errText*.setLocation(5, 35);

*errText*.setSize(550, 470);

tabErr.setControl(grpErr);

// Закончили формирование элементов формы и открываем ее

*shell*.open();

// Цикл обработки сообщений из очереди операционной системы

**while** (!*shell*.isDisposed()) {

**if** (!*display*.readAndDispatch())

*display*.sleep();

}

*display*.dispose();

}

}

Код справки

Файл VvodRaboty.htm

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN">

<HTML>

<HEAD>

<meta name="GENERATOR" content="Microsoft&reg; HTML Help Workshop 4.1">

<Title>Закладка 1 - Ввод работы</Title>

</HEAD>

<BODY>

<h1 align=center>Закладка 1 - Ввод работы</h1>

<p>Данная закладка имеет 7 полей для ввода и 1 кнопку для записи введенных данных. Эта закладка служит для записи данных об организации в таблицу на следующей закладке. Данная закладка показана ниже.</p>

<img src="1-1.png">

<p>Рассмотрим каждое поле более подробно</p>

<ol>

<li>Наименование организации - указывается название организации.</li>

<li>ИНН - указывается идентификационный номер налогоплательщика.</li>

<li>Вид деятельности - указывается каким видом деятельности занимается организация.</li>

<li>Адрес - указывается адрес организации.</li>

<li>Контактное лицо - указывается лицо, с которым можно будет связаться.</li>

<li>Телефон - указывается телефон контактного лица.</li>

<li>Дата - указывается дата записи.</li>

</ol>

<p>Кнопка "Записать" позволяет, после того, как пользователь ввел данные во все поля, записать все эти данные в таблицу на следующей закладке.</p>

</BODY>

</HTML>

Файл TablicaRabot.htm

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN">

<HTML>

<HEAD>

<meta name="GENERATOR" content="Microsoft&reg; HTML Help Workshop 4.1">

<Title>Закладка 2 - Таблица работ</Title>

</HEAD>

<BODY>

<h1 align=center>Закладка 2 - Таблица работ</h1>

<p>На данной закладке расположена таблица с семью столбцами и двумя кнопками. Данная закладка предназначена для хранения данных, а также для дальнейшей обработки этих данных на следующей закладке - графике. Закладка показана ниже.</p>

<img src="2-1.png">

<p>Рассмотрим каждый элемент закладки</p>

<ol>

<li>Таблица созданных работ - таблица, куда записываются данные, введенные пользователем на предыдущей закладке. Таблица имеет 7 столбцов, которые соответсвуют полям из предыдущей закладки. Редактировать данные самостоятельно в таблице нельзя.</li>

<li>Кнопка "Записать таблицу в CSV файл" - кнопка, предназначенная для записи данных из таблицы в CSV файл.</li>

<li>Кнопка "Записать в таблицу из CSV файла" - кнопка, предназначенная для записи данных из CSV файла в таблицу.</li>

</ol>

<p>Вводить данные в таблицу можно двумя способами:</p>

<ol>

<li>Вводить данные на первой закладке и с помощью кнопки "Записать" добавлять эти данные в таблицу.</li>

<li>Использовать кнопку "Записать в таблицу из CSV файла" для заполнения данными из CSV файла.</li>

</ol>

<p>Заполненная таблица имеет следующий вид</p>

<img src="2-2.png">

</BODY>

</HTML>

Файл Grafik.htm

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN">

<HTML>

<HEAD>

<meta name="GENERATOR" content="Microsoft&reg; HTML Help Workshop 4.1">

<Title>Закладка 3 - График</Title>

</HEAD>

<BODY>

<h1 align=center>Закладка 3 - График</h1>

<p>На данной закладке распологается график. В начале работы программы его нет, для того чтобы он появился, необходимо заполнить таблицу созданных работ. Этот график позволяет увидеть сколько записей было сделано за конкретную дату. Начальное изображение этой закладки показано ниже.</p>

<img src="3-1.png">

<p>Так выглядит график если таблица созданных работ заполнена.</p>

<img src="3-2.png">

<p>График перерисовывается всякий раз, как меняются данные в таблице, к примеру если в таблицу была добавлена новая запись.</p>

</BODY>

</HTML>

Файл Nepoladki.htm

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN">

<HTML>

<HEAD>

<meta name="GENERATOR" content="Microsoft&reg; HTML Help Workshop 4.1">

<Title>Закладка 4 - Неполадки</Title>

</HEAD>

<BODY>

<h1 align=center>Закладка 4 - Неполадки</h1>

<p>На данной закладке распологается одно многострочное поле ввода. Оно предназначено для отображения ошибок, которые может увидеть пользователь, если он что-то неправильно ввёл. Если ошибок нет, то это поле будет пустым. Данный вариант как раз показан ниже.</p>

<img src="4-1.png">

<p>Если пользователь где-то неправильно заполнил поле, то на данной вкладке появится сообщение о том, какое поле неправильно заполнено. Неправильно заполнены могут быть все поля. Данный пример показан ниже.</p>

<img src="4-2.png">

</BODY>

</HTML>

Файл MainMenu.htm

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML//EN">

<HTML>

<HEAD>

<meta name="GENERATOR" content="Microsoft&reg; HTML Help Workshop 4.1">

<Title>Главное меню приложения</Title>

</HEAD>

<BODY>

<h1 align=center>Главное меню приложения</h1>

<p>Главное меню приложения делится на 3 раздела. Первый раздел называется "Файлы". Данный раздел предназначен для работы с файлами и функцией выхода из программы, он содержит 3 пункта меню. Рассмотрим каждый из них.</p>

<ol>

<li>Открыть - позволяет открыть ранее созданный файл fun и записать из него данные в поля на первой вкладке.</li>

<li>Сохранить - позволяет сохранить записанные в полях данные в файл fun.</li>

<li>Выход - осуществляет выход из программы.</li>

</ol>

<p>Данный раздел меню показан ниже.</p>

<img src="5-1.png">

<p></p>Второй раздел называется "Функции". Данный раздел содержит дополнительные функции программы. Здесь находится 2 пункта меню.

<ol>

<li>Проверить введенные данные - данная функция позволяет проверить корректность ввода данных на первой закладке. Если все данные верны то программа выдаст следующее.

<img src="5-2.png">

<p>Если где-то ошибка ввода данных, программа предупредит об этом пользователя.</p>

<img src="5-3.png">

</li>

<li>Напечать график - позволяет напечатать график с 3 закладки, при условии что есть принтер, или другое устройство для печати.</li>

</ol>

<p>Данный раздел показан ниже.</p>

<img src="5-4.png">

<p>Третий раздел называется "Справка". Он не содержит дополнительных пунктов меню и предназначен для отображенния данного справочного руководства. Его можно увидеть на всех предыдущих изображениях.</p>

</BODY>

</HTML>