Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

«Санкт-Петербургский государственный электротехнический университет

“ЛЭТИ” им.В.И.Ульянова (Ленина) »

Кафедра ВТ

**ОТЧЕТ**

**по лабораторно-практическим работам № 7 и 8**

**«Построение отчетов в PDF- и HTML- форматах»**

**«Организация многопоточных приложений»**

**по дисциплине «Объектно - ориентированное программирование на языке Java»**

Выполнил Бондаренко Р.В.

Факультет КТИ

Группа № 1307

Подпись преподавателя \_\_\_\_\_\_\_\_\_\_

Санкт-Петербург

2013 г

**Цель работы**

Знакомство со способами формирования отчетов с использованием конструктора Jaspersoft iReport Designert, библиотекой iTest и правилами построения параллельных приложений в языке Java.

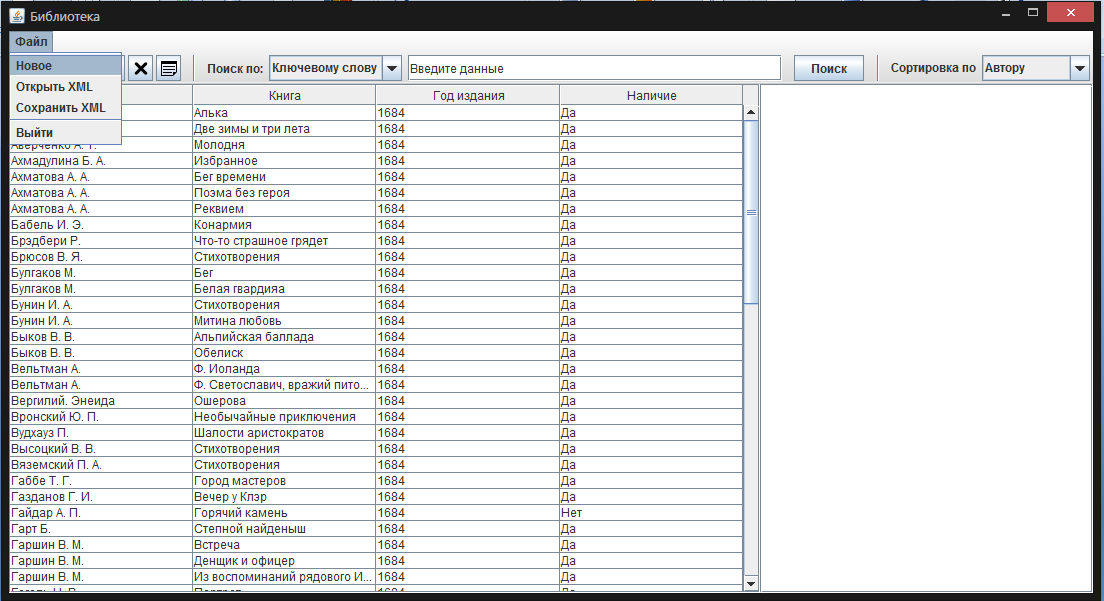
**Описание задания**

1. Распечатка исходного XML – файла.
2. Исходный и отредактированный XML-файл.
3. Скриншоты, иллюстрирующие построение шаблона в дизайнере iReport.
4. Распечатка сгенерированных файлов.
5. Текст документации, сгенерированный Javadoc.
6. Исходный текст построения отчета.
7. Фрагмент кода, отвечающие за организацию параллельной работы трех потоков.
8. Фрагмент кода, отвечающие за сохранения данных в pdf и html форматах.

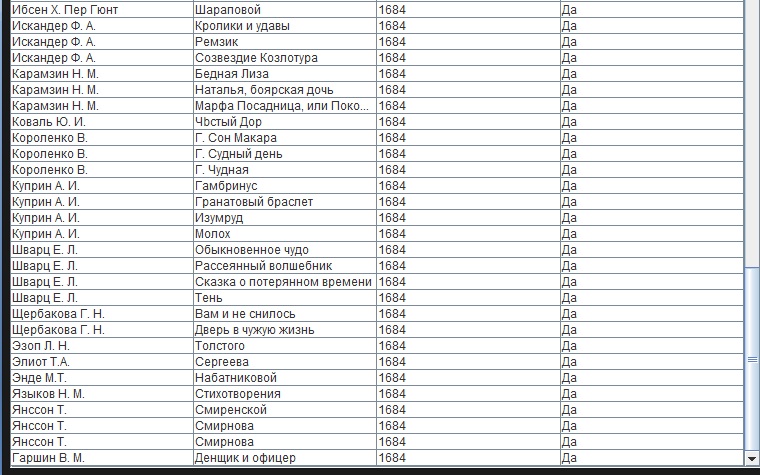
**Описание и проверка работоспособности приложения**

При запуске программы пользователь должен будет нажать на поле меню «Файл», после чего в открывшемся списке нажать на “ Новое ” , в результате произойдёт следующие:

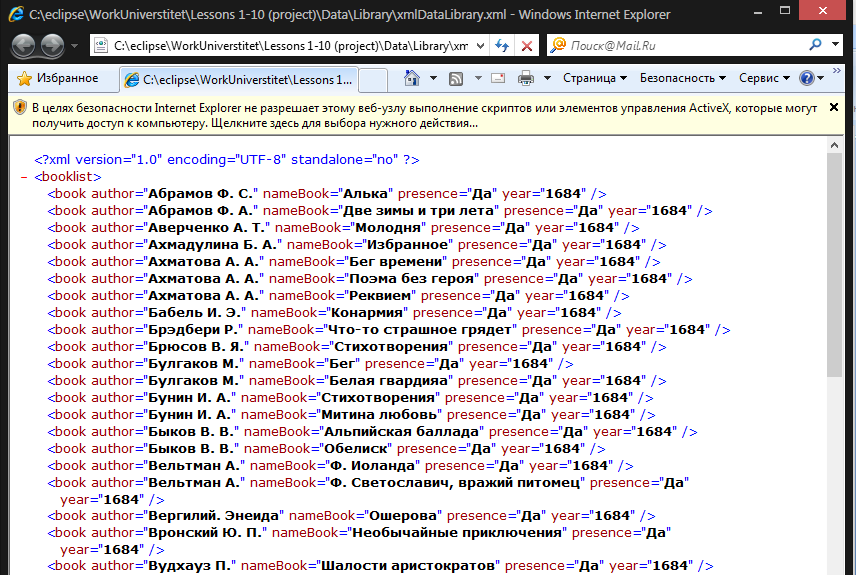
1. Начнётся загрузка данных из xml файла “xmlDataLibrary” в таблицу.

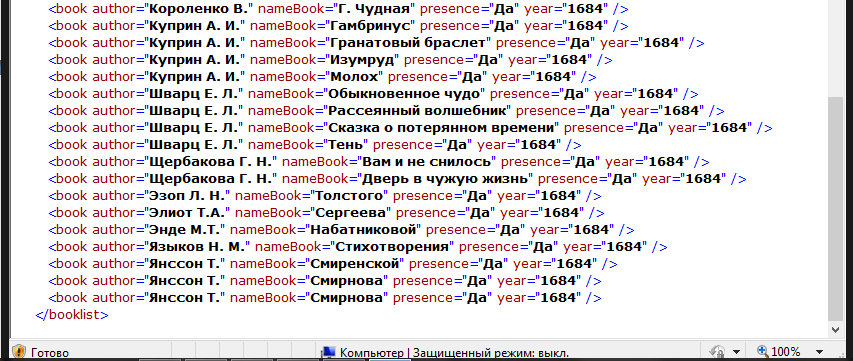


1. К данным, которые загрузились из “xmlDataLibrary” файла, будет добавлена ещё одна книга Гашина В.М. (в конце списка).

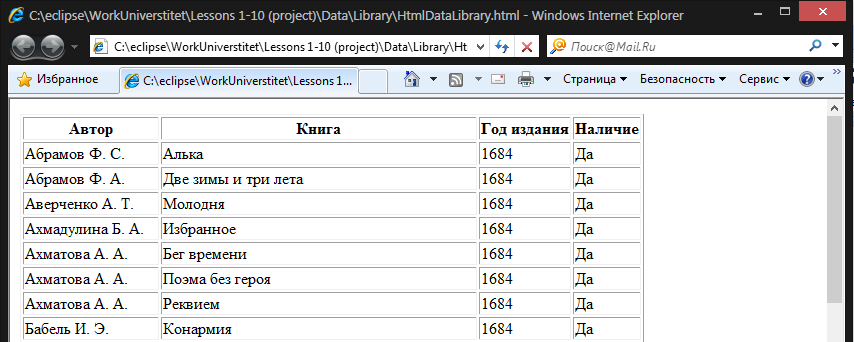


Откроем файла “xmlDataLibrary”, чтобы убедиться, что он не содержит последнею строку(книгу Гашина В.М).

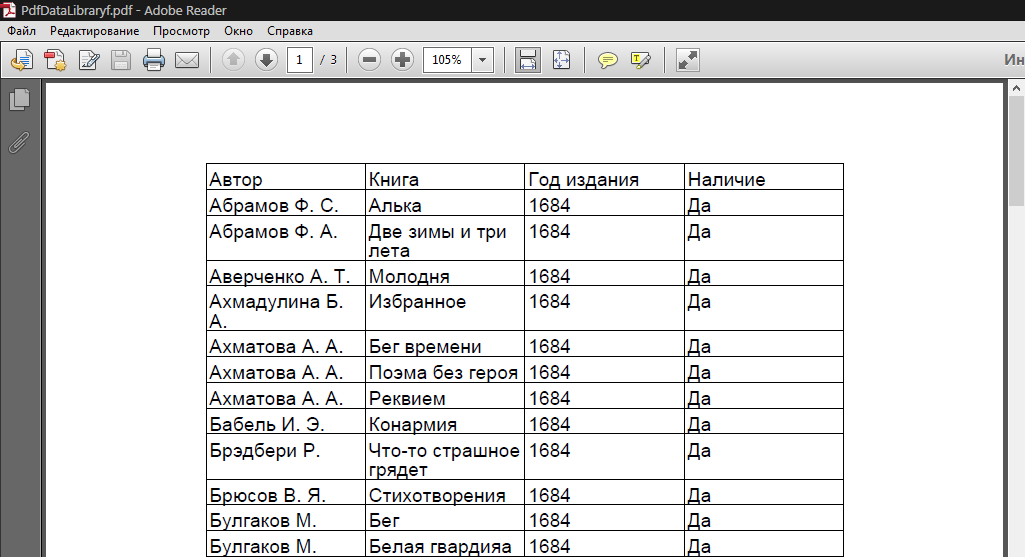


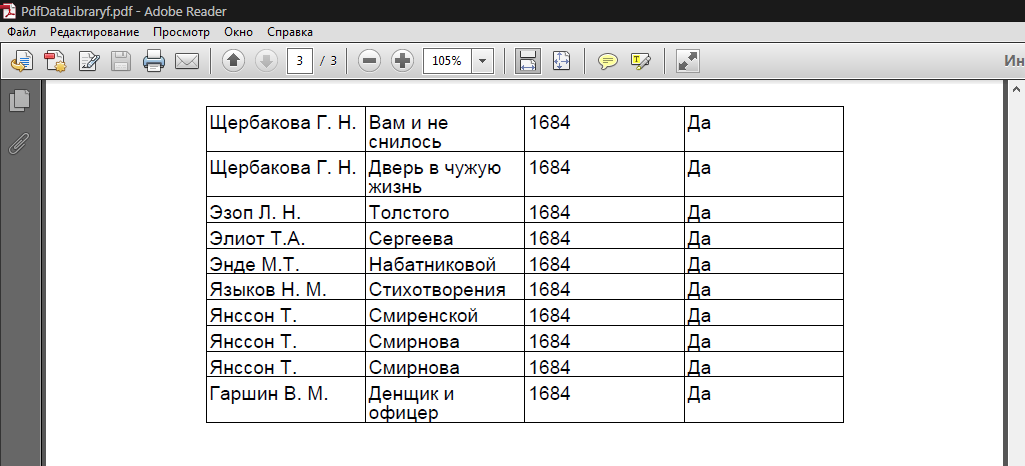


1. Вся данные из таблицы сохраняются в pdf файле “PdfDataLibraryf” и html файле “HtmlDataLibrary”.

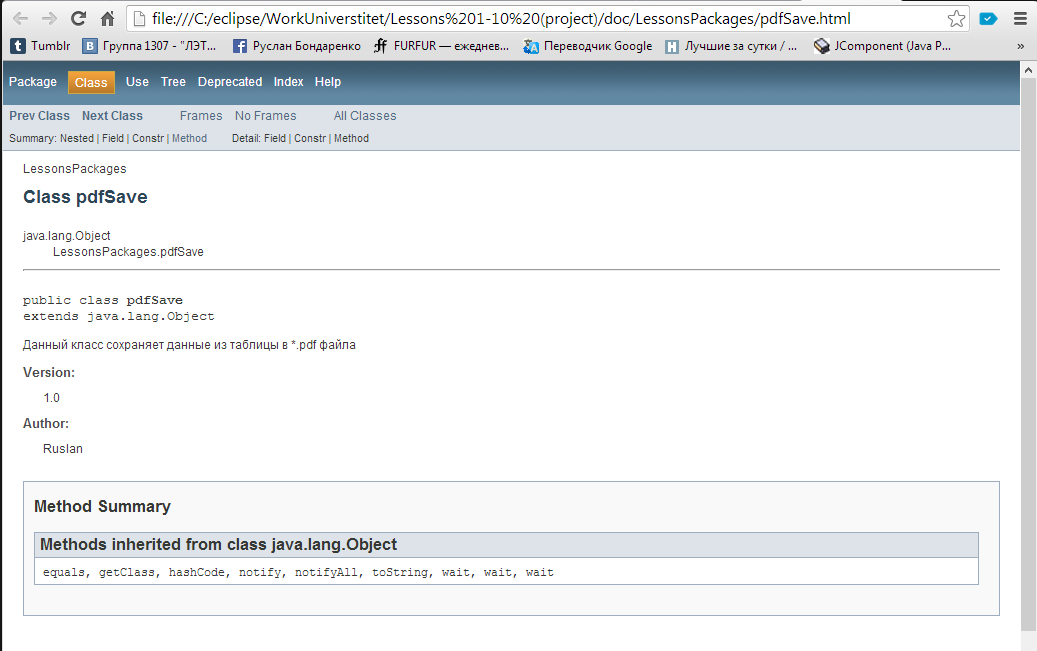


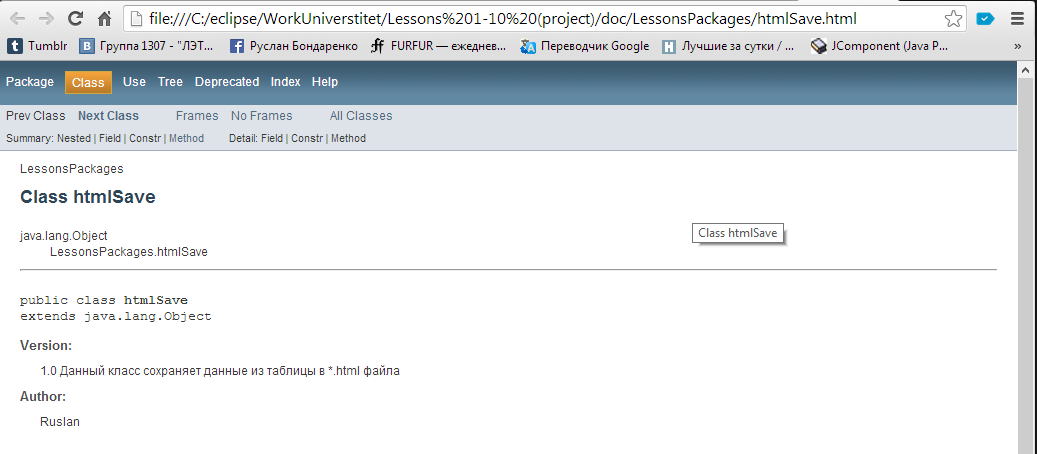






**Текст документации Javadoc**

****

****

**Фрагмент кода, отвечающие за работу трех потоков**

**class** tthread **extends** Thread{

**private** **int** type;

**public** tthread(**int** i) {

type=i;

}

**public** **void** run() {

**if** (type==1) {

**synchronized** (*shared*) {

**try** {

*shared*.wait();

} **catch** (InterruptedException e) {

e.printStackTrace();

}

**try** {

xmlOpen q = **new** xmlOpen("Открыть данные XML",tableModel,table);

} **catch** (Exception e1) {

e1.printStackTrace();

}

}

}

**if** (type==2) {

**synchronized** (*shared*) {

*shared*.notifyAll();

**try** {

*shared*.wait();

} **catch** (InterruptedException e) {

e.printStackTrace();

}

tableModel.addRow(**new** String[] {"Гаршин В. М.","Денщик и офицер","1684","Да"} );

*shared*.notifyAll();

}

}

**if** (type==3) {

**synchronized** (*shared*) {

*shared*.notifyAll();

**try** {

*shared*.wait();

} **catch** (InterruptedException e) {

e.printStackTrace();

}

pdfSave qwe = **new** pdfSave(tableModel);

htmlSave qwe2 = **new** htmlSave(tableModel);

}

}

}

}

**new** tthread(1).start();

**new** tthread(2).start();

**new** tthread(3).start();

**Фрагмент кода, отвечающие за сохранения данных в pdf и html форматах**

**Сохранения данных в pdf формате**

**package** LessonsPackages;

**import** java.io.FileNotFoundException;

**import** java.io.FileOutputStream;

**import** java.io.IOException;

**import** javax.swing.table.DefaultTableModel;

**import** com.itextpdf.text.Document;

**import** com.itextpdf.text.DocumentException;

**import** com.itextpdf.text.Font;

**import** com.itextpdf.text.PageSize;

**import** com.itextpdf.text.Phrase;

**import** com.itextpdf.text.pdf.BaseFont;

**import** com.itextpdf.text.pdf.PdfPCell;

**import** com.itextpdf.text.pdf.PdfPTable;

**import** com.itextpdf.text.pdf.PdfWriter;

**public** **class** pdfSave {

pdfSave(DefaultTableModel tableModel) {

Document document = **new** Document(PageSize.*A4*, 50, 50, 50, 50);

PdfPTable t = **new** PdfPTable(4);

**try** {

PdfWriter writer = PdfWriter.*getInstance*(document, **new** FileOutputStream("./Data/Library/PdfDataLibraryf.pdf"));

} **catch** (FileNotFoundException e) {

e.printStackTrace();

} **catch** (DocumentException e) {

e.printStackTrace();

}

BaseFont bfComic = **null**;

**try** {

bfComic = BaseFont.*createFont*("/Windows/Fonts/Arial.ttf",BaseFont.*IDENTITY\_H*, BaseFont.*EMBEDDED*);

} **catch** (DocumentException e1) {

e1.printStackTrace();

} **catch** (IOException e1) {

e1.printStackTrace();

}

Font font1 = **new** Font(bfComic, 12);

t.addCell(**new** PdfPCell(**new** Phrase("Автор",font1)));

t.addCell(**new** PdfPCell(**new** Phrase("Книга",font1)));

t.addCell(**new** PdfPCell(**new** Phrase("Год издания",font1)));

t.addCell(**new** PdfPCell(**new** Phrase("Наличие",font1)));

**for**(**int** i = 0; i < tableModel.getRowCount(); i++){

t.addCell(**new** Phrase((String) tableModel.getValueAt(i,0),font1));

t.addCell(**new** Phrase((String) tableModel.getValueAt(i,1),font1));

t.addCell(**new** Phrase((String) tableModel.getValueAt(i,2),font1));

t.addCell(**new** Phrase((String) tableModel.getValueAt(i,3),font1));

}

document.open();

**try** {

document.add(t);

} **catch** (DocumentException e) {

e.printStackTrace();

}

document.close();

}

}

**Сохранения данных в html формате**

**package** LessonsPackages;

**import** java.io.FileWriter;

**import** java.io.IOException;

**import** java.io.PrintWriter;

**import** javax.swing.table.DefaultTableModel;

**public** **class** htmlSave {

htmlSave(DefaultTableModel tableModel){

PrintWriter pw = **null**;

**try** {

pw = **new** PrintWriter(**new** FileWriter("./Data/Library/HtmlDataLibrary.html"));

} **catch** (IOException e) {

e.printStackTrace();

}

pw.println("<TABLE BORDER><TR><TH>Автор<TH>Книга<TH>Год издания<TH>Наличие</TR>");

**for**(**int** i = 0; i < tableModel.getRowCount(); i++) {

**int** square = i \* i;

pw.println("<TR><TD>" + (String) tableModel.getValueAt(i,0)

+ "<TD>" + (String) tableModel.getValueAt(i,1)

+ "<TD>" + (String) tableModel.getValueAt(i,2)

+ "<TD>" + (String) tableModel.getValueAt(i,3));

}

pw.println("</TABLE>");

pw.close();

}

}

**Вывод**

Во время выполнения данной лабораторной работы, были получены навыки работы с технологией формирования отчетов с использованием конструктора Jaspersoft iReport Designert, библиотекой iTest и правилами построения параллельных приложений в языке Java. В пункте «Описание и проверки работоспособности приложения» ошибок выявлено не было.