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-практической работе № 6**

**«Обработка XML-документов»**

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

Выполнил Бондаренко Р.В.

Факультет КТИ

Группа № 1307

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

Санкт-Петербург

2013 г

**Цель работы**

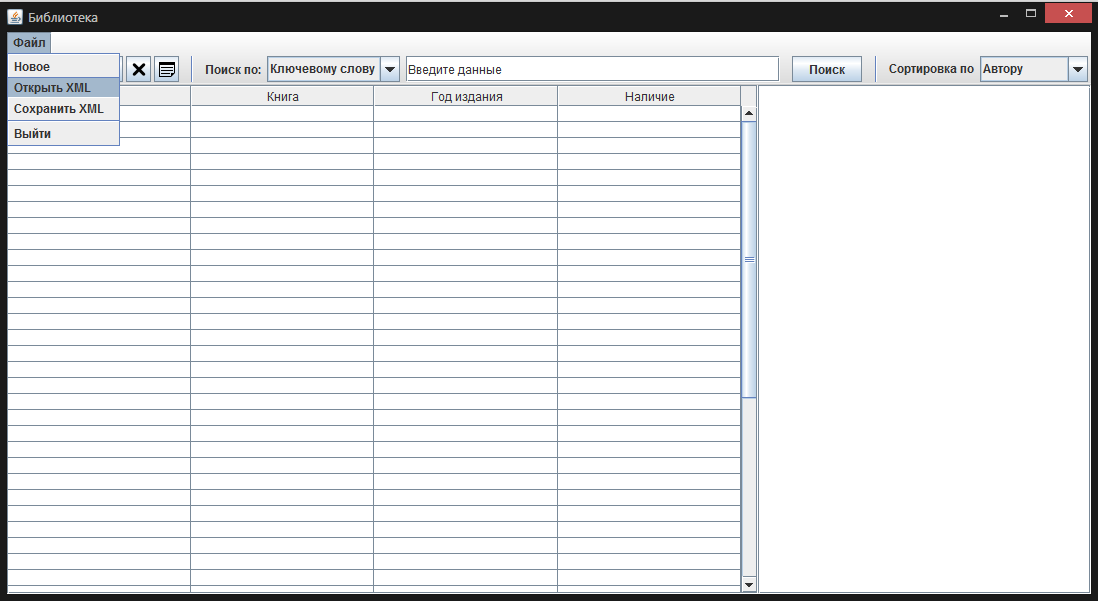
Знакомство с технологией обработки XML-документов и файлов.

**Описание задания**

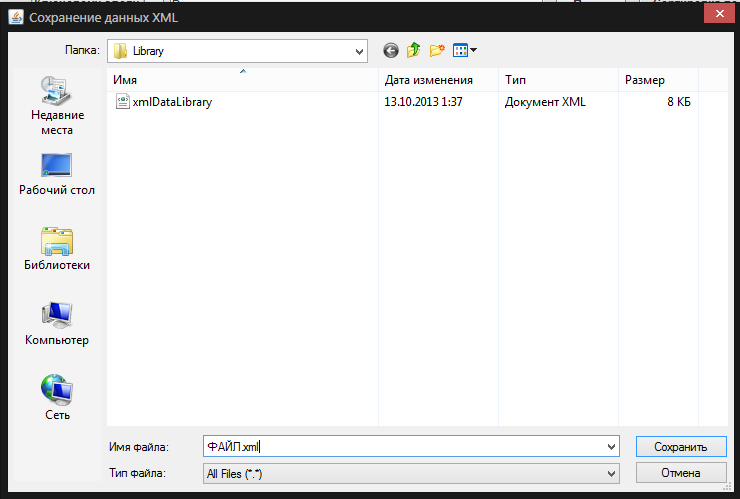
1. Распечатки ХML-файлов до загрузки данных в электронную форму и после их выгрузки.
2. Скриншоты, иллюстрирующие процесс загрузки данных в XML-файл и выгрузки из него.
3. Текст документации, сгенерированный Javadoc.
4. Фрагменты кода, отвечающие за сохранение и чтение данных из xml-файла.

**Описание приложения**

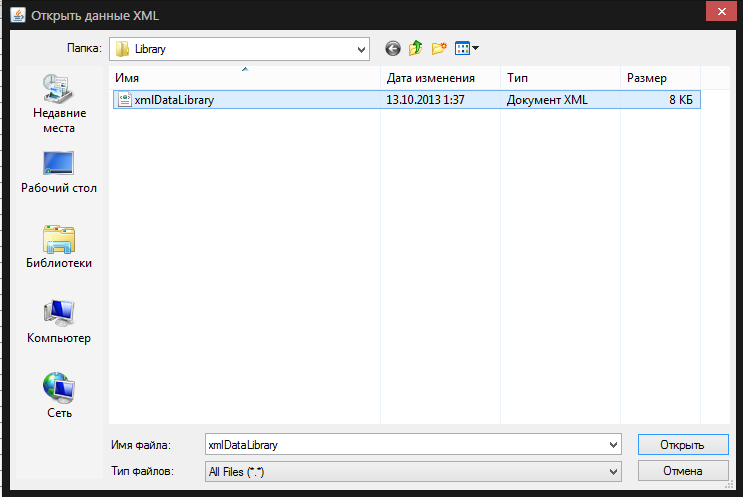
При запуске программы пользователь должен будет нажать на поле меню «Файл», после чего в открывшемся списке нажать на “ Cохранить XML ” или “Открыть XML ” для сохранения в .xml или считывания из него.



Если нажать на “ Cохранить XML ” , то появится следующие окно, где нужно будет вести имя нового файла или указать на существующий файл, после чего нажать на “Сохранить”.



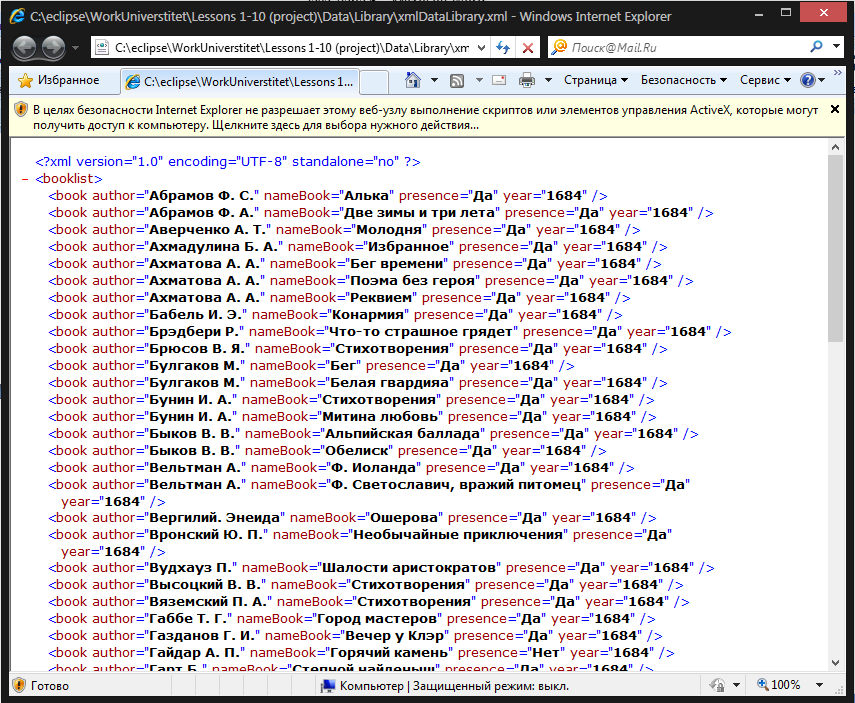
Если нажать на “ Открыть XML ” , то появится окно, где нужно будет выбрать xml файл, после чего нажать на “Открыть”.



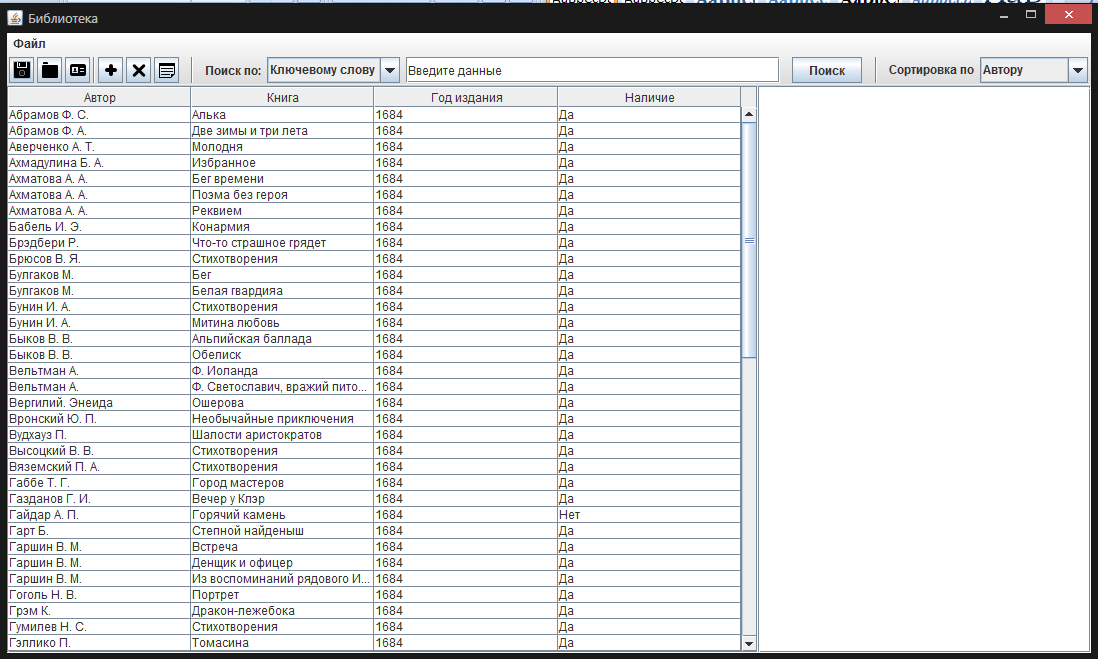
Можно открыть только файл xml формата(другие будут скрыты от пользователя).

**Описание проверки работоспособности приложения**

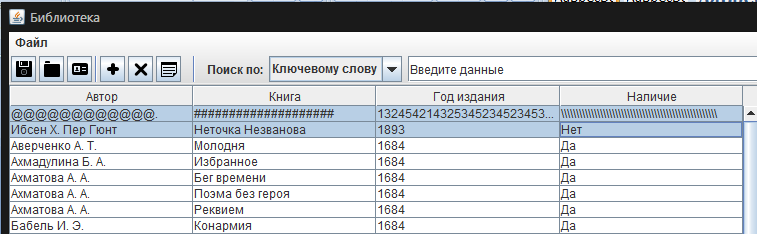
Откроем xml файл «xmlDataLibrary» .



Теперь откроем его в приложение.

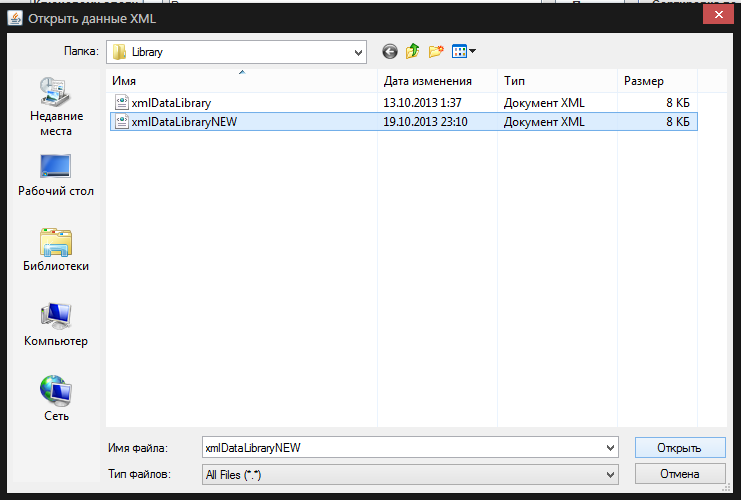


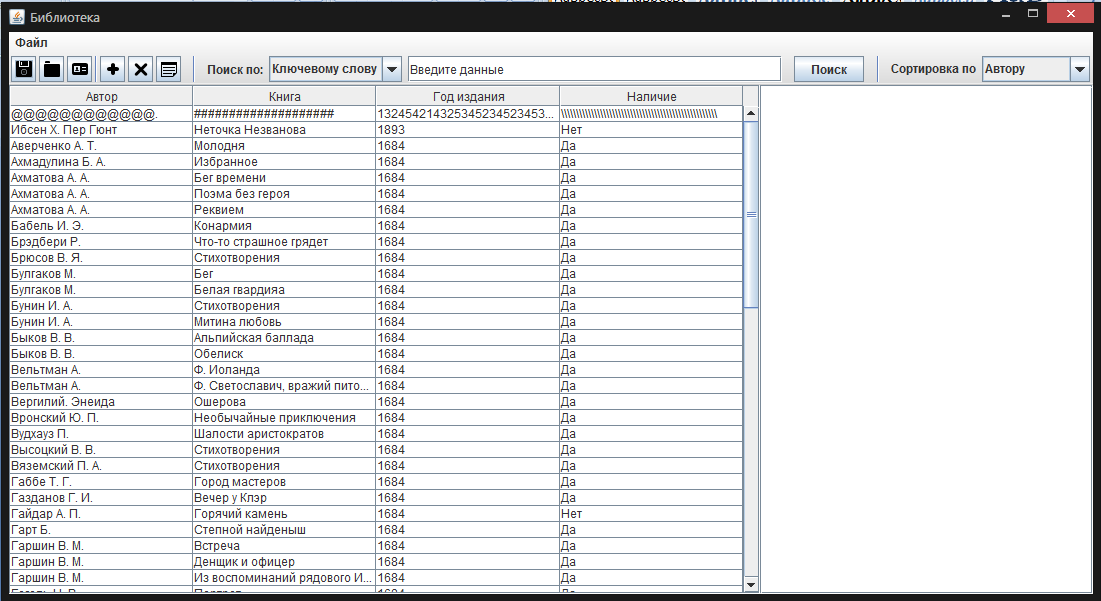
Далее изменим данные в первой и второй строке таблицы.



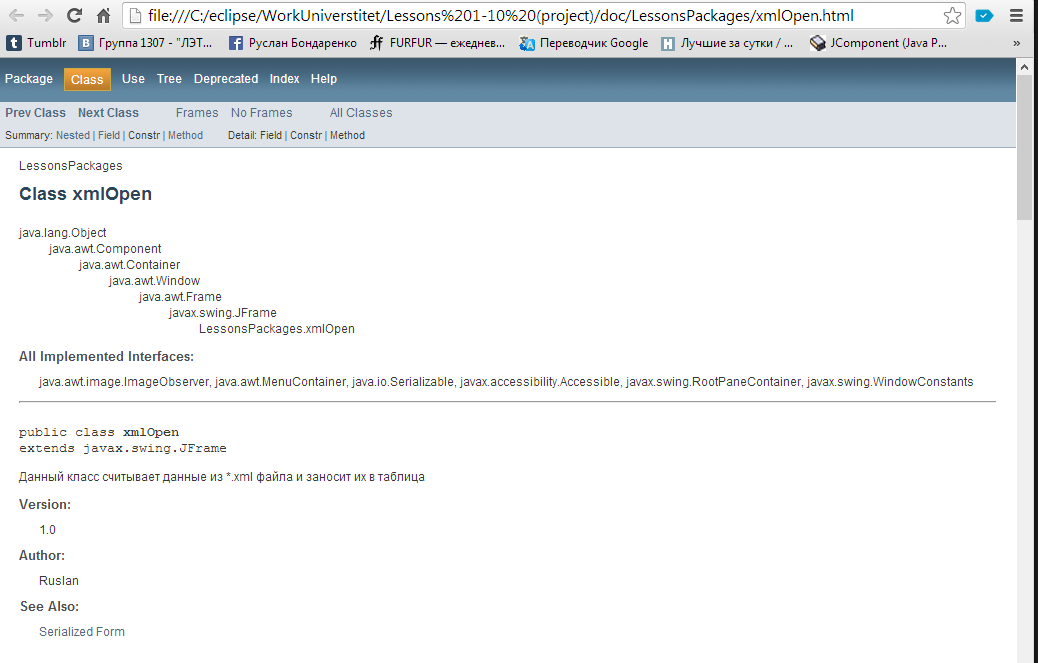
Теперь сохраним их в xml файле «xmlDataLibraryNEW» и откроем его вручную и через приложение.

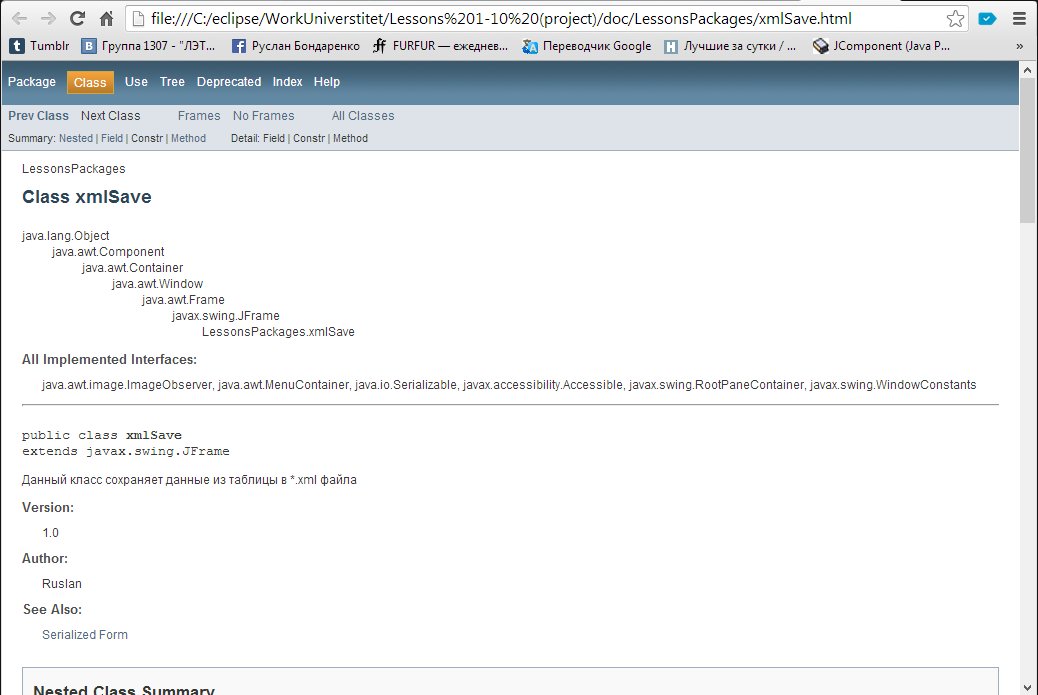






**Текст документации Javadoc**

****

****

**Фрагменты кода, отвечающие за сохранение и чтение данных из xml-файла**

**1.Чтение данных из xml-файла**

**package** LessonsPackages;

**import** java.awt.FileDialog;

**import** java.io.File;

**import** java.io.IOException;

**import** javax.swing.JFrame;

**import** javax.swing.JOptionPane;

**import** javax.swing.JTable;

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

**import** javax.xml.parsers.DocumentBuilder;

**import** javax.xml.parsers.DocumentBuilderFactory;

**import** javax.xml.transform.TransformerConfigurationException;

**import** javax.xml.transform.TransformerException;

**import** org.w3c.dom.Document;

**import** org.w3c.dom.NamedNodeMap;

**import** org.w3c.dom.Node;

**import** org.w3c.dom.NodeList;

**import** org.xml.sax.SAXException;

**public** **class** xmlOpen **extends** JFrame{

**static** String *fileNameOpen*;

xmlOpen(String str, DefaultTableModel tableModel,JTable table) **throws** Exception {

//if(tableModel.getRowCount()==0) return;

FileDialog openXML = **new** FileDialog(**this**,str,FileDialog.*LOAD*);

openXML.setFile("\*.xml");// Установка начального каталога

openXML.setVisible(**true**);

//Определяем имя каталога или файла

*fileNameOpen* = openXML.getDirectory() + openXML.getFile();

**if**(*fileNameOpen* == **null**) **return**; // Пользователь нажал отмена

tableModel = (DefaultTableModel) table.getModel();

tableModel.setRowCount(0);

**try** {

Document doc = *getDocument*();

//showDocument(doc, textArea);

doc.getDocumentElement().normalize();

// Получение списка элементов book

NodeList nlBooks = doc.getElementsByTagName("book");

**for** ( **int** temp = 0; temp < nlBooks.getLength(); temp ++){

Node elem = nlBooks.item(temp);

NamedNodeMap attrs = elem.getAttributes();

String author = attrs.getNamedItem("author").getNodeValue();

String titlse = attrs.getNamedItem("nameBook").getNodeValue();

String have = attrs.getNamedItem("year").getNodeValue();

String xz = attrs.getNamedItem("presence").getNodeValue();

tableModel.addRow(**new** String[] {author, titlse, have, xz });

}

// Ошибка при чтение XML файла

}**catch**(SAXException e){

e.printStackTrace();

// Ошибка ввода - вывода

}**catch**(IOException e){

e.printStackTrace();

}

}

**private** **static** Document getDocument() **throws** Exception {

**try** {

// Получаем парсер, порождающий дерево объектов XML - документов

DocumentBuilderFactory f = DocumentBuilderFactory.*newInstance*();// Новый экземпляр

// Создает пустой документ

DocumentBuilder builder = f.newDocumentBuilder();

**return** builder.parse(**new** File(*fileNameOpen*));

} **catch** (Exception exception) {

**throw** **new** Exception("XML parsing error!");

}

}

}

**2.Сохранение данных из xml-файла**

**package** LessonsPackages;

**import** java.awt.FileDialog;

**import** java.io.FileOutputStream;

**import** java.io.FileWriter;

**import** java.io.IOException;

**import** javax.swing.JFrame;

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

**import** javax.xml.parsers.DocumentBuilder;

**import** javax.xml.parsers.DocumentBuilderFactory;

**import** javax.xml.transform.OutputKeys;

**import** javax.xml.transform.Transformer;

**import** javax.xml.transform.TransformerConfigurationException;

**import** javax.xml.transform.TransformerException;

**import** javax.xml.transform.TransformerFactory;

**import** javax.xml.transform.dom.DOMSource;

**import** javax.xml.transform.stream.StreamResult;

**import** org.w3c.dom.Document;

**import** org.w3c.dom.Node;

**import** org.w3c.dom.Attr;

**import** org.w3c.dom.Element;

**public** **class** xmlSave **extends** JFrame{

xmlSave(String str, DefaultTableModel tableModel) **throws** Exception{

//if(tableModel.getRowCount().==0) return;

FileDialog savaXML = **new** FileDialog(**this**,str,FileDialog.*SAVE*);

savaXML.setFile("\*.xml");// Установка начального каталога

savaXML.setVisible(**true**);

//Определяем имя каталога или файла

String fileNameSave = savaXML.getDirectory() + savaXML.getFile();

**if**(fileNameSave == **null**) **return**; // Пользователь нажал отмена

Document doc = *getDocument*();

// Создаём корневой элемент booklist и добавляем его в документ

Node booklist = doc.createElement("booklist");// создать элемент

doc.appendChild(booklist);// добавляем ребёнка

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

Element book = doc.createElement("book");

booklist.appendChild(book);

book.setAttribute("author", (String) tableModel.getValueAt(i, 0));

book.setAttribute("nameBook", (String) tableModel.getValueAt(i, 1));

book.setAttribute("year", (String) tableModel.getValueAt(i, 2));

book.setAttribute("presence", (String) tableModel.getValueAt(i, 3));

}

**try**{

// Создание преобразование документа

Transformer trans=TransformerFactory.*newInstance*().newTransformer();

trans.setOutputProperty(OutputKeys.*METHOD*, "xml");

trans.setOutputProperty(OutputKeys.*INDENT*, "yes");// хз

trans.transform(**new** DOMSource(doc), **new** StreamResult(**new** FileOutputStream(fileNameSave)));

// Ошибка создания XML преобразователя

}**catch**(TransformerConfigurationException e){

e.printStackTrace();

// Ошибка работы XML преобразователя

}**catch**(TransformerException e){

e.printStackTrace();

// Ошибка ввода - вывода

}**catch**(IOException e){

e.printStackTrace();

}

}

**private** **static** Document getDocument() **throws** Exception {

**try** {

// Получаем парсер, порождающий дерево объектов XML - документов

DocumentBuilderFactory f = DocumentBuilderFactory.*newInstance*();// Новый экземпляр

// Создает пустой документ

DocumentBuilder builder = f.newDocumentBuilder();

// Разбирает ( получает ) данные по пути

//return builder.parse(new File("test.xml"));

**return** builder.newDocument();

} **catch** (Exception exception) {

**throw** **new** Exception("XML parsing error!");

}

}

}

**Вывод**

Во время выполнения данной лабораторной работы, были получены навыки работы с технологией обработки XML-документов и файлов. В пункте «Описание проверки работоспособности приложения» ошибок выявлено не было.