

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 МОЭВМ

ОТЧЕТ
по лабораторно-практической работе № 7
«Построение отчетов
в PDF- и HTML- форматах»

Выполнил: Мохно Д. А.

Факультет КТИ

Группа № 3312

Преподаватель: Павловский М.Г.

Подпись преподавателя _____

Санкт-Петербург

2024 г

Цель работы

Знакомство с технологией обработки XML-документов и файлов.

**Распечатки XML-файлов до загрузки данных в экранную форму и
после их выгрузки.**

data.xml до внесения изменений:

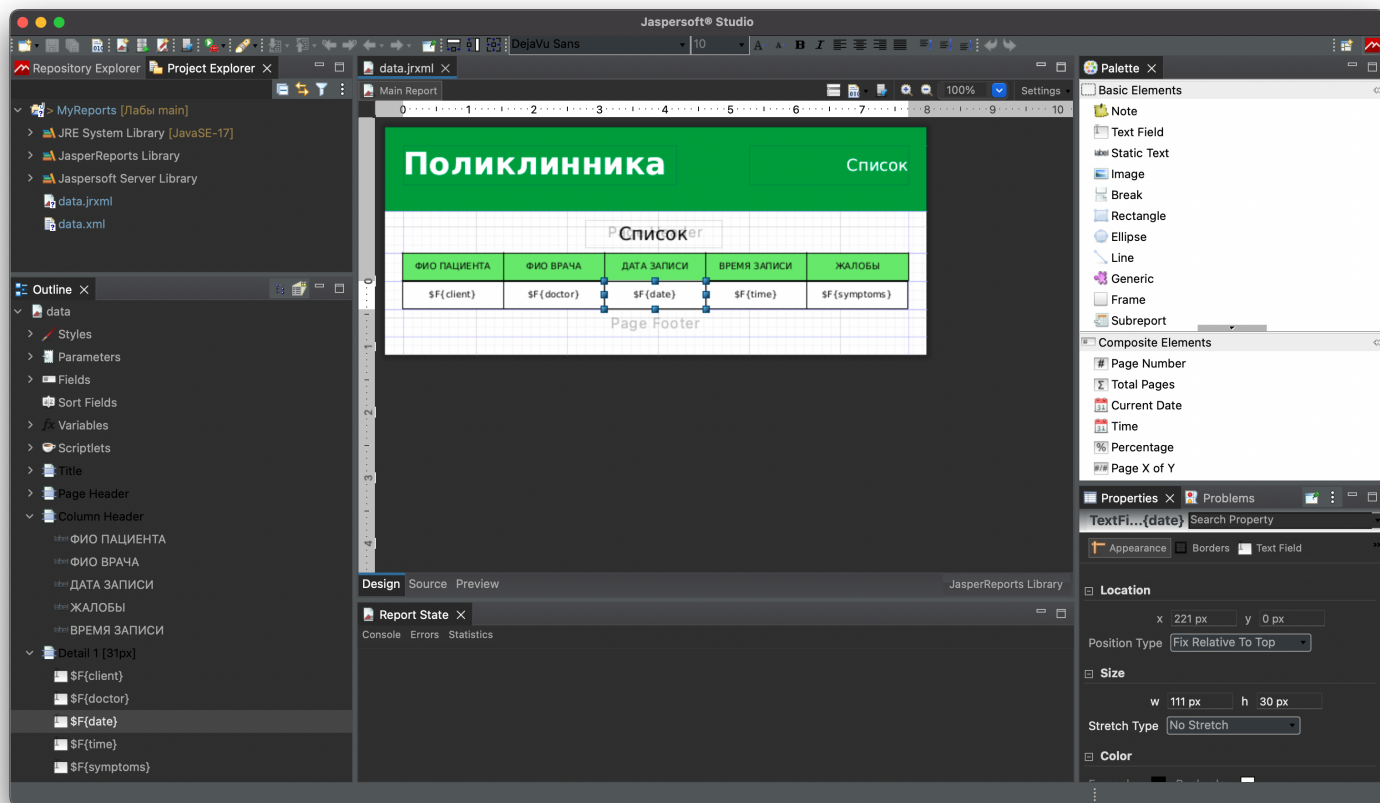
```
<?xml version="1.0" encoding="UTF-8" ?>
<base>
  <record client="Годунов Е. А." doctor="Смирнов В. П." date="24.09.2024" time="10:00"
symptoms="Боль в горле"/>
  <record client="Иванова А. В." doctor="Кузнецов М. И." date="25.09.2024"
time="11:00" symptoms="Температура"/>
  <record client="Сидоров К. А." doctor="Попов А. Н." date="26.09.2024" time="09:30"
symptoms="Кашель"/>
  <record client="Петрова Л. М." doctor="Соколов Б. Т." date="27.09.2024" time="14:00"
symptoms="Головная боль"/>
  <record client="Семенов И. О." doctor="Лебедев В. У." date="28.09.2024" time="15:15"
symptoms="Боль в спине"/>
  <record client="Николаев Д. Р." doctor="Козлов Г. А." date="29.09.2024" time="08:45"
symptoms="Аллергия"/>
  <record client="Федорова Е. С." doctor="Новиков Д. В." date="30.09.2024"
time="13:00" symptoms="Боль в животе"/>
  <record client="Алексеева М. Б." doctor="Морозов С. Е." date="01.10.2024"
time="10:30" symptoms="Усталость"/>
  <record client="Романов А. И." doctor="Петров Н. Ч." date="02.10.2024" time="12:00"
symptoms="Бронхит"/>
  <record client="Соловьев Д. П." doctor="Волков Р. Ш." date="03.10.2024" time="16:00"
symptoms="Мигрень"/>
</base>
```

data.xml после внесения изменений:

```
<?xml version="1.0" encoding="UTF-8" ?>
<base>
  <record client="Годунов Е. А." doctor="Смирнов В. П." date="24.09.2024" time="10:00"
symptoms="Боль в горле"/>
  <record client="Иванова А. В." doctor="Кузнецов М. И." date="25.09.2024"
time="11:00" symptoms="Температура"/>
  <record client="Сидоров К. А." doctor="Попов А. Н." date="26.09.2024" time="09:30"
symptoms="Кашель"/>
  <record client="Петрова Л. М." doctor="Соколов Б. Т." date="27.09.2024" time="14:00"
symptoms="Головная боль"/>
  <record client="Семенов И. О." doctor="Лебедев В. У." date="28.09.2024"
time="15:15" symptoms="Боль в спине"/>
```

</base>

Скриншоты, иллюстрирующие построение шаблона в дизайнере Jaspersoft studio.



Текст документации, сгенерированный Javadoc.

PACKAGE

CLASS

USE

TREE

INDEX

HELP

SUMMARY: NESTED | FIELD | CONSTR | METHODDETAIL: FIELD | CONSTR | METHOD

SEARCH

Class Processor

java.lang.Object[⚡]
Processor

public class **Processor**
extends **Object**[⚡]

Класс Processor выполняет операции по работе с файлами и таблицами для сохранения, открытия, редактирования и отображения данных в формате CSV.

Nested Class Summary

Nested Classes

Modifier and Type	Class	Description
(package private) static class	Processor.Parser	Класс Parser предоставляет методы для работы с XML-файлами.

Field Summary

Fields

Modifier and Type	Field	Description
String [⚡]	currentFilePath	Текущий путь к файлу
JTable [⚡]	table	Таблица для отображения данных
DefaultTableModel [⚡]	tableModel	Модель таблицы для управления данными
JFrame [⚡]	window	Основное окно приложения

Constructor Summary

Constructors

Constructor	Description
Processor (JFrame [⚡] frame)	Конструктор, принимающий основное окно приложения.

Method Summary

All Methods

Instance Methods

Concrete Methods

Modifier and Type	Method	Description
void	createTable (String [⚡] [] colNames)	Создает таблицу с заданными именами столбцов.
void	openFile ()	Открывает диалоговое окно для выбора CSV-файла и загружает данные в таблицу.
void	printFile ()	Выводит сообщение о печати файла.
String [⚡] [][]	readFile (String [⚡] pathToFile)	Читает содержимое CSV-файла и возвращает его как двумерный массив строк.
void	saveFile ()	Сохраняет данные таблицы в текущий CSV/XML-файл.
void	saveFileAs ()	Открывает диалоговое окно для выбора нового места и имени для сохранения CSV-файла.
private void	tableFill (String [⚡] [][] text)	Заполняет таблицу данными из двумерного массива строк.
private void	tableFill (Document [⚡] data)	Заполняет таблицу данными из двумерного массива строк.
void	writeFile (String [⚡] pathToFile)	Записывает содержимое таблицы в CSV-файл.
private void	writeFile (Document [⚡] data)	Записывает содержимое таблицы в XML-файл.

Methods inherited from class java.lang.Object[⚡]
clone[⚡], equals[⚡], finalize[⚡], getClass[⚡], hashCode[⚡], notify[⚡], notifyAll[⚡], toString[⚡], wait[⚡], wait[⚡], wait[⚡]

Field Details

window

public JFrame[⚡] window
Основное окно приложения

table


public JTable[⚡] table
Таблица для отображения данных

tableModel

public DefaultTableModel[⚡] tableModel
Модель таблицы для управления данными

currentFilePath

public String[⚡] currentFilePath
Текущий путь к файлу

Constructor Details
<div><div>Processor</div><div><div>Processor(JFrame¹² frame)</div><div>Конструктор, принимающий основное окно приложения.</div><div>Parameters: frame - основное окно приложения JFrame</div></div></div>
Method Details
<div><div>openFile</div><div><div>public void openFile()</div><div>Открывает диалоговое окно для выбора CSV-файла и загружает данные в таблицу.</div></div></div>
<div><div>saveFile </div><div><div>public void saveFile()</div><div>Сохраняет данные таблицы в текущий CSV/XML-файл.</div></div></div>
<div><div>saveFileAs</div><div><div>public void saveFileAs()</div><div>Открывает диалоговое окно для выбора нового места и имени для сохранения CSV-файла.</div></div></div>
<div><div>printFile</div><div><div>public void printFile()</div><div>Выводит сообщение о печати файла. Этот метод можно расширить для добавления функциональности печати.</div></div></div>
<div><div>readFile</div><div><div>public String¹²[][] readFile(String¹² pathToFile)</div><div>Читает содержимое CSV-файла и возвращает его как двумерный массив строк.</div><div>Parameters: pathToFile - путь к файлу для чтения</div><div>Returns: двумерный массив строк, представляющий данные таблицы, или null в случае ошибки чтения файла</div></div></div>
<div><div>writeFile</div><div><div>public void writeFile(String¹² pathToFile)</div><div>Записывает содержимое таблицы в CSV-файл.</div><div>Parameters: pathToFile - путь к файлу для записи</div></div></div>
<div><div>writeFile</div><div><div>private void writeFile(Document¹² data)</div><div>Записывает содержимое таблицы в XML-файл.</div><div>Parameters: data - объект xml документа</div></div></div>
<div><div>tableFill</div><div><div>private void tableFill(String¹²[][] text)</div><div>Заполняет таблицу данными из двумерного массива строк.</div><div>Parameters: text - двумерный массив строк, представляющий данные для таблицы</div></div></div>
<div><div>tableFill</div><div><div>private void tableFill(Document¹² data)</div><div>Заполняет таблицу данными из двумерного массива строк.</div><div>Parameters: data - объект xml документа</div></div></div>
<div><div>createTable</div><div><div>public void createTable(String¹²[] colNames)</div><div>Создает таблицу с заданными именами столбцов.</div><div>Parameters: colNames - массив строк с именами столбцов</div></div></div>

PACKAGE

CLASS

USE

TREE

INDEX

HELP

SUMMARY: NESTED | FIELD | CONSTR | METHODDETAIL: FIELD | CONSTR | METHOD

SEARCH

Class Processor.Parser

java.lang.Object[Ⓢ]
Processor.Parser

Enclosing class:
Processor

Inheritance Tree

static class Processor.Parser
extends Object[Ⓢ]

Класс Parser предоставляет методы для работы с XML-файлами. Содержит статические методы для чтения данных из XML-файла в объект `Document` и для преобразования данных таблицы в XML-формат.

Constructor Summary

Constructors

Constructor	Description
Parser()	

Method Summary

All MethodsStatic MethodsConcrete Methods

Modifier and Type	Method	Description
static Document [Ⓢ]	parse(String [Ⓢ] pathToFile)	Парсит XML-файл и возвращает объект `Document`.
static Document [Ⓢ]	to_xml(DefaultTableModel [Ⓢ] tableModel)	Преобразует данные из таблицы в XML-формат и возвращает объект `Document`.

Methods inherited from class java.lang.Object[Ⓢ]

clone[Ⓢ], equals[Ⓢ], finalize[Ⓢ], getClass[Ⓢ], hashCode[Ⓢ], notify[Ⓢ], notifyAll[Ⓢ], toString[Ⓢ], wait[Ⓢ], wait[Ⓢ], wait[Ⓢ]

Constructor Details

Parser

Parser()

Method Details

parse

public static Document[Ⓢ] parse(String[Ⓢ] pathToFile)

Парсит XML-файл и возвращает объект `Document`.

Parameters:
pathToFile - путь к XML-файлу для парсинга

Returns:
объект `Document`, представляющий содержимое XML-файла, или null в случае ошибки чтения или парсинга

to_xml

public static Document[Ⓢ] to_xml(DefaultTableModel[Ⓢ] tableModel)

Преобразует данные из таблицы в XML-формат и возвращает объект `Document`.

Parameters:
tableModel - модель таблицы `DefaultTableModel`, содержащая данные для преобразования

Returns:
объект `Document`, представляющий данные в XML-формате

Throws:
RuntimeException[Ⓢ] - если возникает ошибка конфигурации парсера

Фрагменты кода, отвечающие за сохранение и чтение данных из XML-файла.

```
/**
 * Класс Parser предоставляет методы для работы с XML-файлами.
 * Содержит статические методы для чтения данных из XML-файла в объект
 * `Document`
 * и для преобразования данных таблицы в XML-формат.
 */
static class Parser {
    /**
     * Парсит XML-файл и возвращает объект `Document`.
     *
     * @param pathToFile путь к XML-файлу для парсинга
     * @return объект `Document`, представляющий содержимое XML-файла,
     * или null в случае ошибки чтения или парсинга
     */
    public static Document parse(String pathToFile) {
        try {
```

```

        // Создание парсера документа
        DocumentBuilder dBuilder =
DocumentBuilderFactory.newInstance().newDocumentBuilder();
        // Чтение документа из файла
        Document doc = dBuilder.parse(new File(pathToFile));
        // Нормализация документа
        doc.getDocumentElement().normalize();
        return doc;
    } catch (ParserConfigurationException | IOException | SAXException e)
{
        JOptionPane.showMessageDialog(null, "Проблема с чтением файла " +
pathToFile,
            "ERROR", JOptionPane.INFORMATION_MESSAGE);
    }
    // Обработка ошибки парсера при чтении данных из XML-файла
    return null;
}

/**
 * Преобразует данные из таблицы в XML-формат и возвращает объект
 * `Document`.
 *
 * @param tableModel модель таблицы `DefaultTableModel`, содержащая
данные для преобразования
 * @return объект `Document`, представляющий данные в XML-формате
 * @throws RuntimeException если возникает ошибка конфигурации парсера
 */
public static Document to_xml(DefaultTableModel tableModel) {
    try {
        // Создание парсера документа
        DocumentBuilder builder =
DocumentBuilderFactory.newInstance().newDocumentBuilder();
        // Создание пустого документа
        Document doc = builder.newDocument();

        // Создание корневого элемента booklist и добавление его в
документ
        Node data = doc.createElement("base");
        doc.appendChild(data);

        // Создание дочерних элементов book и присвоение значений
атрибутам
        for (int i = 0; i < tableModel.getRowCount(); i++) {
            Element book = doc.createElement("record");
            data.appendChild(book);
            book.setAttribute("client", (String) tableModel.getValueAt(i,
0));
            book.setAttribute("doctor", (String) tableModel.getValueAt(i,
1));
            book.setAttribute("date", (String) tableModel.getValueAt(i,
2));
            book.setAttribute("time", (String) tableModel.getValueAt(i,
2));
            book.setAttribute("symptoms", (String)
tableModel.getValueAt(i, 2));
        }
        return doc;
    } catch (ParserConfigurationException e) {
        throw new RuntimeException(e);
    }
}
}

```

```

/**
 * Заполняет таблицу данными из двумерного массива строк.
 *
 * @param data объект xml документа
 */
private void tableFill(Document data) {
    if (data != null) {
        NodeList list = data.getElementsByTagName("record");
        // Цикл просмотра списка элементов и запись данных в таблицу
        for (int i = 0; i < list.getLength(); i++) {
            // Выбор очередного элемента списка
            Node elem = list.item(i);
            // Получение списка атрибутов элемента
            NamedNodeMap attrs = elem.getAttributes();
            // Чтение атрибутов элемента
            String client = attrs.getNamedItem("client").getNodeValue();
            String doctor = attrs.getNamedItem("doctor").getNodeValue();
            String date = attrs.getNamedItem("date").getNodeValue();
            String time = attrs.getNamedItem("time").getNodeValue();
            String symptoms = attrs.getNamedItem("symptoms").getNodeValue();
            // Запись данных в таблицу
            tableModel.addRow(new String[]{client, doctor, date, time,
symptoms});
        }
    }
}

```

```

/**
 * Записывает содержимое таблицы в XML-файл.
 *
 * @param data объект xml документа
 */
private void writeFile(Document data) {
    try {
        // Создание преобразователя документа
        Transformer trans =
TransformerFactory.newInstance().newTransformer();
        // Создание файла с именем books.xml для записи документа
        java.io.FileWriter fw = new FileWriter(currentFilePath);
        // Запись документа в файл
        trans.transform(new DOMSource(data), new StreamResult(fw));
    } catch (TransformerException | IOException e) {
        JOptionPane.showMessageDialog(window, "Проблема с сохранением в файл
" + currentFilePath,
            "ERROR", JOptionPane.INFORMATION_MESSAGE);
    }
}

```

Вывод

В результате выполнения работы были изучены правила работы с построением отчётов формате pdf и html в java и получены практические навыки в программировании на этом языке.

Ссылки

https://github.com/DanyaMokhno/OOP_Labs/tree/main/com.study_oop.Laba_7