**Министерство науки и высшего образования Российской Федерации**  
  
Федеральное государственное автономное образовательное учреждение  
высшего образования «Уральский федеральный университет  
имени первого Президента России Б.Н. Ельцина»  
  
Институт радиоэлектроники и информационных технологий – РТФ  
Центр ускоренного обучения

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 10**по дисциплине «Курсовое проектирование Java»

**Тема: «Введение в работу с XML, JSON, HTML и Excel»**

|  |  |
| --- | --- |
| Студент группы РИВ-410938у | М.О. Покуль |
| Преподаватель: | Н.А. Архипов |

**Екатеринбург 2024**

**СОДЕРЖАНИЕ**

[1 Постановка задачи 3](#_Toc190818257)

[2 Описание задачи 4](#_Toc190818258)

[3 Ход выполнения работы. 5](#_Toc190818259)

[3.1 Примеры из раздела 1 5](#_Toc190818260)

[3.1.1 Создание XML 5](#_Toc190818261)

[3.1.2 Чтение XML 8](#_Toc190818262)

[3.1.3 Создание JSON 10](#_Toc190818263)

[3.1.4 Чтение JSON 12](#_Toc190818264)

[3.1.5 Парсинг страницы https://itlearn.ru/first-steps 13](#_Toc190818265)

[3.1.6 Парсинг страницы http://fat.urfu.ru/index.html 14](#_Toc190818266)

[3.1.7 Создание файла Excel 16](#_Toc190818267)

[3.1.8 Чтение файла Excel 18](#_Toc190818268)

[3.2 Доработки XML парсера 19](#_Toc190818269)

[3.2.1 Добавление новых книг 19](#_Toc190818270)

[3.2.2 Поиск книг 21](#_Toc190818271)

[3.2.3 Удаление книг 23](#_Toc190818272)

[3.3 Доработки JSON парсера 26](#_Toc190818273)

[3.3.1 Поиск книг 26](#_Toc190818274)

[3.3.2 Добавление книги 28](#_Toc190818275)

[3.3.3 Удаление книги 29](#_Toc190818276)

[3.4 Доработки HTML парсера 31](#_Toc190818277)

[3.4.1 Запись полученных данных в файл, обработка ошибок при получении HTML-кода страницы, например вывод сообщения об ошибке и попытка переподключения к сайту. 31](#_Toc190818278)

[4 Вывод 33](#_Toc190818279)

1. Постановка задачи

Получение представление о механизме парсинга в языке программирования Java.

1. Описание задачи
2. Выполнить все примеры из раздела 1.
3. Выполнить следующие доработки проекта XML парсера:
4. Сделайте в текстовом редакторе свой файл в формате XML в зависимости от варианта (Приложение 1).
5. Добавьте возможность записывать новые книги в XML-файл. Например, пользователь может ввести данные о новой книге, а программа добавит новый элемент <book> в XML-файл.
6. Добавьте возможность поиска книг по автору или году издания. Например, пользователь может ввести автора или год издания, а программа выведет список книг, удовлетворяющих этому критерию поиска.
7. Реализуйте функцию удаления книги из XML-файла. Например, пользователь может ввести название книги, которую хочет удалить, и программа удалит соответствующий элемент <book> из XML-файла.
8. Выполнить следующие доработки проекта JSON парсера.
9. Сделайте в текстовом редакторе свой файл в формате JSON в зависимости от варианта (Приложение 1).
10. Добавьте функционал для поиска книг по автору.
11. Добавьте функционал для добавления новой книги в массив.
12. Добавьте функционал для удаления книги из массива по названию.
13. Выполнить следующие доработки проекта HTML парсера.
14. Добавьте функционал записи полученных данных в файл для сохранения информации на будущее.
15. Добавьте обработку ошибок при получении HTML-кода страницы, например вывод сообщения об ошибке и попытка переподключения к сайту.
16. Выполните следующую доработки проекта Excel парсера:
17. Улучшение работы с ошибками. Программы, которые читают Excel файлы, могут столкнуться с ошибками, такими как неправильный формат файла, отсутствие запрашиваемого листа и т.д. Улучшение работы с ошибками может включать более подробные сообщения об ошибках, рекомендации для исправления ошибок и возможность повторного запуска программы после исправления ошибок.
18. Ход выполнения работы.
    1. Примеры из раздела 1
       1. Создание XML

package Lab10.Task1;  
  
import java.io.File;  
import javax.xml.parsers.DocumentBuilder;  
import javax.xml.parsers.DocumentBuilderFactory;  
import javax.xml.transform.\*;  
import javax.xml.transform.dom.DOMSource;  
import javax.xml.transform.stream.StreamResult;  
import org.w3c.dom.Document;  
import org.w3c.dom.Element;  
  
public class XMLWrite {  
 public static void main(String[] args) {  
 try {  
 DocumentBuilderFactory docFactory = DocumentBuilderFactory.*newInstance*();  
 DocumentBuilder docBuilder = docFactory.newDocumentBuilder();  
  
 Document doc = docBuilder.newDocument();  
 Element rootElement = doc.createElement("Library");  
 doc.appendChild(rootElement);  
  
 Element book1 = doc.createElement("book");  
 rootElement.appendChild(book1);  
  
 Element title1 = doc.createElement("title");  
 title1.appendChild(doc.createTextNode("Война и мир"));  
 book1.appendChild(title1);  
  
 Element author1 = doc.createElement("author");  
 author1.appendChild(doc.createTextNode("Лев Толстой"));  
 book1.appendChild(author1);  
  
 Element year1 = doc.createElement("year");  
 year1.appendChild(doc.createTextNode("1869"));  
 book1.appendChild(year1);  
  
 Element book2 = doc.createElement("book");  
 rootElement.appendChild(book2);  
  
 Element title2 = doc.createElement("title");  
 title2.appendChild(doc.createTextNode("Мастер и Маргарита"));  
 book2.appendChild(title2);  
  
 Element author2 = doc.createElement("author");  
 author2.appendChild(doc.createTextNode("Михаил Булгаков"));  
 book2.appendChild(author2);  
  
 Element year2 = doc.createElement("year");  
 year2.appendChild(doc.createTextNode("1967"));  
 book2.appendChild(year2);  
  
  
 doc.setXmlStandalone(true);  
 doc.normalizeDocument();  
 TransformerFactory tf = TransformerFactory.*newInstance*();  
 Transformer transformer = tf.newTransformer();  
 transformer.setOutputProperty(OutputKeys.*ENCODING*, "UTF-8");  
 transformer.setOutputProperty(OutputKeys.*STANDALONE*, "yes");  
 transformer.setOutputProperty(OutputKeys.*INDENT*, "yes");  
 DOMSource source = new DOMSource(doc);  
 StreamResult result = new StreamResult(new File("src/Lab10/Task1/example.xml"));  
 transformer.transform(source, result);  
  
 System.*out*.println("XML-файл успесшно создан");  
 } catch (Exception e){  
 e.printStackTrace();  
 }  
 }  
}

* + 1. Чтение XML

package Lab10.Task1;  
  
import org.w3c.dom.\*;  
  
import javax.xml.parsers.DocumentBuilder;  
import javax.xml.parsers.DocumentBuilderFactory;  
import java.io.File;  
  
public class XMLRead {  
 public static void main(String[] args) {  
 try {  
 File inputFile = new File("src/Lab10/Task1/example.xml");  
 DocumentBuilderFactory dbFactory = DocumentBuilderFactory.*newInstance*();  
 DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();  
 Document doc = dBuilder.parse(inputFile);  
 doc.getDocumentElement().normalize();  
 System.*out*.println("Корневой элемент = " + doc.getDocumentElement().getNodeName());  
 NodeList nodeList = doc.getElementsByTagName("book");  
 for (int i = 0; i < nodeList.getLength(); i++) {  
 Node node = nodeList.item(i);  
 System.*out*.println("\nТекущий элемент = " + node.getNodeName());  
 if (node.getNodeType() == Node.*ELEMENT\_NODE*){  
 Element element = (Element) node;  
 System.*out*.println("Название книги: " + element.getElementsByTagName("title").item(0).getTextContent());  
 System.*out*.println("Автор: " + element.getElementsByTagName("author").item(0).getTextContent());  
 System.*out*.println("Год издания: " + element.getElementsByTagName("year").item(0).getTextContent());  
 }  
 }  
 }catch (Exception e){  
 e.printStackTrace();  
 }  
 }  
}

* + 1. Создание JSON

package Lab10.Task1;  
  
import org.json.simple.JSONArray;  
import org.json.simple.JSONObject;  
  
import java.io.FileWriter;  
import java.io.IOException;  
  
public class JSONWrite {  
 public static void main(String[] args) {  
 JSONObject library = new JSONObject();  
 JSONArray books = new JSONArray();  
  
 JSONObject book1 = new JSONObject();  
 book1.put("title","Война и мир");  
 book1.put("author","Лев Толстой");  
 book1.put("year",1869);  
  
 JSONObject book2 = new JSONObject();  
 book2.put("title","Мастер и Маргарита");  
 book2.put("author","Михаил Булгаков");  
 book2.put("year",1967);  
  
 books.add(book1);  
 books.add(book2);  
  
 library.put("books", books);  
  
 try (FileWriter file = new FileWriter("src/Lab10/Task1/example.json")){  
 file.write(library.toJSONString());  
 System.*out*.println("Json файл успешно создан!");  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
}

* + 1. Чтение JSON

package Lab10.Task1;  
  
import org.json.simple.JSONArray;  
import org.json.simple.JSONObject;  
import org.json.simple.parser.JSONParser;  
  
import java.io.FileReader;  
  
public class JSONRead {  
 public static void main(String[] args) {  
 try {  
 JSONParser parser = new JSONParser();  
 Object obj = parser.parse(new FileReader("src/Lab10/Task1/example.json"));  
 JSONObject jsonObject = (JSONObject) obj;  
 System.*out*.println("Корневой элемент: " + jsonObject.keySet().iterator().next());  
 JSONArray jsonArray = (JSONArray) jsonObject.get("books");  
  
 for (Object o : jsonArray){  
 JSONObject book = (JSONObject) o;  
 System.*out*.println("\nТекущий элемент: book");  
 System.*out*.println("Название книги = " + book.get("title"));  
 System.*out*.println("Автор = " + book.get("author"));  
 System.*out*.println("Год издания = " + book.get("year"));  
 }  
 }catch (Exception e){  
 e.printStackTrace();  
 }  
 }  
}

* + 1. Парсинг страницы <https://itlearn.ru/first-steps>

package Lab10.Task1;  
  
import org.jsoup.Jsoup;  
import org.jsoup.nodes.Document;  
import org.jsoup.nodes.Element;  
import org.jsoup.select.Elements;  
  
public class HTMLParse1 {  
 public static void main(String[] args) {  
 String url = "https://itlearn.ru/first-steps";  
 try {  
 Document doc = Jsoup.*connect*(url).get();  
 Elements links = doc.select("a[href]");  
 for (Element link : links){  
 System.*out*.println(link.attr("abs:href"));  
 }  
 }catch (Exception e){  
 e.printStackTrace();  
 }  
 }  
}

* + 1. Парсинг страницы <http://fat.urfu.ru/index.html>

package Lab10.Task1;  
  
import org.jsoup.Jsoup;  
import org.jsoup.nodes.Document;  
import org.jsoup.nodes.Element;  
import org.jsoup.nodes.Node;  
import org.jsoup.select.Elements;  
  
import java.io.IOException;  
import java.util.List;  
  
public class HTMLParse2 {  
 public static void main(String[] args) {  
 try {  
 Document doc = Jsoup.*connect*("http://fat.urfu.ru/index.html").get();  
  
 Elements newsParent = doc.select("body > table > tbody > tr > td > div > table > " + "tbody > tr:nth-child(5) > td:nth-child(3) > table > tbody > " + "tr > td:nth-child(1)");  
  
 for (int i = 0; i < 20; i++) {  
 if (!(i % 2 == 0)){  
 List<Node> nodes = newsParent.get(0).childNodes();  
 System.*out*.println("Тема: " + ((Element) nodes.get(i)).getElementsByClass("blocktitle").get(0).childNodes().get(0));  
 System.*out*.println("Дата: " + ((Element) nodes.get(i)).getElementsByClass("blockdate").get(0).childNodes().get(0) + "\n");  
 }  
 }  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
  
 }  
}

* + 1. Создание файла Excel

package Lab10.Task1;  
  
import org.apache.poi.ss.usermodel.Row;  
import org.apache.poi.xssf.usermodel.XSSFSheet;  
import org.apache.poi.xssf.usermodel.XSSFWorkbook;  
  
import java.io.FileOutputStream;  
import java.io.IOException;  
  
public class EXCELWrite {  
 public static void main(String[] args) throws IOException {  
 XSSFWorkbook workbook = new XSSFWorkbook();  
 XSSFSheet sheet = workbook.createSheet("Товары");  
  
 Row headerRow = sheet.createRow(0);  
 headerRow.createCell(0).setCellValue("Товар");  
 headerRow.createCell(1).setCellValue("Характеристики");  
 headerRow.createCell(2).setCellValue("Стоимость");  
  
 Row dataRow1 = sheet.createRow(1);  
 dataRow1.createCell(0).setCellValue("Книга");  
 dataRow1.createCell(1).setCellValue("Жанр: Фантастика, Автор: Иванов И.И.");  
 dataRow1.createCell(2).setCellValue(500.0);  
  
 Row dataRow2 = sheet.createRow(2);  
 dataRow2.createCell(0).setCellValue("Компьютер");  
 dataRow2.createCell(1).setCellValue("Процессор: Intel Core i5, Оперативная память:");  
 dataRow2.createCell(2).setCellValue(25000.0);  
  
 String filePath = "src/Lab10/Task1/example.xlsx";  
 FileOutputStream outputStream = new FileOutputStream(filePath);  
 workbook.write(outputStream);  
 workbook.close();  
 outputStream.close();  
  
 System.*out*.println("Данные записаны в файл: " + filePath);  
 }  
}

* + 1. Чтение файла Excel

package Lab10.Task1;  
  
import org.apache.poi.ss.usermodel.Cell;  
import org.apache.poi.ss.usermodel.Row;  
import org.apache.poi.xssf.usermodel.XSSFSheet;  
import org.apache.poi.xssf.usermodel.XSSFWorkbook;  
  
import java.io.FileInputStream;  
import java.io.IOException;  
  
public class EXCELRead {  
 public static void main(String[] args) throws IOException {  
 String filePath = "src/Lab10/Task1/example.xlsx";  
 FileInputStream inputStream = new FileInputStream(filePath);  
  
 XSSFWorkbook workbook = new XSSFWorkbook(inputStream);  
 XSSFSheet sheet = workbook.getSheet("Товары");  
  
 for (Row row : sheet) {  
 for (Cell cell : row) {  
 System.*out*.print(cell.toString() + "\t");  
 }  
 System.*out*.println();  
 }  
  
 workbook.close();  
 inputStream.close();  
 }  
}

* 1. Доработки XML парсера
     1. Добавление новых книг

package Lab10.Task2;  
  
import org.w3c.dom.Document;  
import org.w3c.dom.Element;  
import org.w3c.dom.Node;  
import org.w3c.dom.NodeList;  
  
import javax.xml.parsers.DocumentBuilder;  
import javax.xml.parsers.DocumentBuilderFactory;  
import javax.xml.transform.OutputKeys;  
import javax.xml.transform.Transformer;  
import javax.xml.transform.TransformerFactory;  
import javax.xml.transform.dom.DOMSource;  
import javax.xml.transform.stream.StreamResult;  
import java.io.File;  
  
public class XMLAdd {  
 public static void main(String[] args) {  
 try {  
 File inputFile = new File("src/Lab10/Task1/example.xml");  
 DocumentBuilderFactory dbFactory = DocumentBuilderFactory.*newInstance*();  
 DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();  
 Document doc = dBuilder.parse(inputFile);  
  
 Element rootElement = doc.getDocumentElement();  
  
  
 Element book1 = doc.createElement("book");  
 rootElement.appendChild(book1);  
  
 Element title1 = doc.createElement("title");  
 title1.appendChild(doc.createTextNode("Война"));  
 book1.appendChild(title1);  
  
 Element author1 = doc.createElement("author");  
 author1.appendChild(doc.createTextNode("Лев"));  
 book1.appendChild(author1);  
  
 Element year1 = doc.createElement("year");  
 year1.appendChild(doc.createTextNode("1111"));  
 book1.appendChild(year1);  
  
 */\*-------------------------------------------------------------------------\*/* doc.getDocumentElement().normalize();  
  
 Transformer transformer = TransformerFactory.*newInstance*().newTransformer();  
 transformer.setOutputProperty(OutputKeys.*INDENT*, "yes");  
 DOMSource source = new DOMSource(doc);  
 StreamResult result = new StreamResult(new File("src/Lab10/Task1/example.xml"));  
 transformer.transform(source, result);  
 }catch (Exception e){  
 e.printStackTrace();  
 }  
 }  
}

* + 1. Поиск книг

package Lab10.Task2;  
  
import org.w3c.dom.Document;  
import org.w3c.dom.Element;  
import org.w3c.dom.Node;  
import org.w3c.dom.NodeList;  
  
import javax.xml.parsers.DocumentBuilder;  
import javax.xml.parsers.DocumentBuilderFactory;  
import java.io.File;  
import java.util.stream.\*;  
import java.util.List;  
  
public class XMLSearch {  
 public static void main(String[] args) {  
 try {  
 File inputFile = new File("src/Lab10/Task1/example.xml");  
 DocumentBuilderFactory dbFactory = DocumentBuilderFactory.*newInstance*();  
 DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();  
 Document doc = dBuilder.parse(inputFile);  
 System.*out*.println("Корневой элемент = " + doc.getDocumentElement().getNodeName());  
 NodeList nodeList = doc.getElementsByTagName("book");  
  
 List<Element> books = IntStream.*range*(0, nodeList.getLength())  
 .mapToObj(nodeList::item)  
 .filter(node -> node.getNodeType() == Node.*ELEMENT\_NODE*)  
 .map(node -> (Element) node)  
 .filter(element -> {  
 String author = element.getElementsByTagName("author").item(0).getTextContent();  
 String year = element.getElementsByTagName("year").item(0).getTextContent();  
 return author.equalsIgnoreCase("Лев Толстой") && year.equals("1869");  
 })  
 .collect(Collectors.*toList*());  
  
 for (Element book : books) {  
 System.*out*.println("\nТекущий элемент = " + book.getNodeName());  
 if (book.getNodeType() == Node.*ELEMENT\_NODE*){  
 System.*out*.println("Название книги: " + book.getElementsByTagName("title").item(0).getTextContent());  
 System.*out*.println("Автор: " + book.getElementsByTagName("author").item(0).getTextContent());  
 System.*out*.println("Год издания: " + book.getElementsByTagName("year").item(0).getTextContent());  
 }  
 }  
 }catch (Exception e){  
 e.printStackTrace();  
 }  
 }  
}

* + 1. Удаление книг

package Lab10.Task2;  
  
import org.w3c.dom.Document;  
import org.w3c.dom.Element;  
import org.w3c.dom.Node;  
import org.w3c.dom.NodeList;  
  
import javax.xml.parsers.DocumentBuilder;  
import javax.xml.parsers.DocumentBuilderFactory;  
import javax.xml.transform.OutputKeys;  
import javax.xml.transform.Transformer;  
import javax.xml.transform.TransformerFactory;  
import javax.xml.transform.dom.DOMSource;  
import javax.xml.transform.stream.StreamResult;  
import java.io.File;  
import java.util.stream.\*;  
import java.util.List;  
  
public class XMLRemove {  
 public static void main(String[] args) {  
 try {  
 File inputFile = new File("src/Lab10/Task1/example.xml");  
 DocumentBuilderFactory dbFactory = DocumentBuilderFactory.*newInstance*();  
 DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();  
 Document doc = dBuilder.parse(inputFile);  
 System.*out*.println("Корневой элемент = " + doc.getDocumentElement().getNodeName());  
 NodeList nodeList = doc.getElementsByTagName("book");  
  
 List<Element> books = IntStream.*range*(0, nodeList.getLength())  
 .mapToObj(nodeList::item)  
 .filter(node -> node.getNodeType() == Node.*ELEMENT\_NODE*)  
 .map(node -> (Element) node)  
 .filter(element -> {  
 String author = element.getElementsByTagName("author").item(0).getTextContent();  
 String year = element.getElementsByTagName("year").item(0).getTextContent();  
 return author.equalsIgnoreCase("Лев Толстой") && year.equals("1869");  
 })  
 .collect(Collectors.*toList*());  
  
 for (Element book : books) {  
 System.*out*.println("\nТекущий элемент = " + book.getNodeName());  
 if (book.getNodeType() == Node.*ELEMENT\_NODE*){  
 Element element = (Element) book;  
 Node parentNode = element.getParentNode();  
 parentNode.removeChild(element);  
 System.*out*.println("Название книги: " + element.getElementsByTagName("title").item(0).getTextContent());  
 System.*out*.println("Автор: " + element.getElementsByTagName("author").item(0).getTextContent());  
 System.*out*.println("Год издания: " + element.getElementsByTagName("year").item(0).getTextContent());  
 }  
 }  
  
 Transformer transformer = TransformerFactory.*newInstance*().newTransformer();  
 DOMSource source = new DOMSource(doc);  
 StreamResult result = new StreamResult(new File("src/Lab10/Task1/example.xml"));  
 transformer.transform(source, result);  
 }catch (Exception e){  
 e.printStackTrace();  
 }  
 }  
}

* 1. Доработки JSON парсера
     1. Поиск книг

package Lab10.Task3;  
  
import org.json.simple.JSONArray;  
import org.json.simple.JSONObject;  
import org.json.simple.parser.JSONParser;  
  
import java.io.FileReader;  
import java.util.stream.Stream;  
  
public class JSONSearch {  
 public static void main(String[] args) {  
 try {  
 JSONParser parser = new JSONParser();  
 Object obj = parser.parse(new FileReader("src/Lab10/Task1/example.json"));  
 JSONObject jsonObject = (JSONObject) obj;  
 System.*out*.println("Корневой элемент: " + jsonObject.keySet().iterator().next());  
 JSONArray jsonArray = (JSONArray) jsonObject.get("books");  
  
 String author = "Лев Толстой";  
 jsonArray.stream().map(book -> (JSONObject) book)  
 .filter(book -> author.equals(((JSONObject) book).get("author")))  
 .forEach(book -> {  
 System.*out*.println("\nТекущий элемент: book");  
 System.*out*.println("Название книги: " + ((JSONObject) book).get("title"));  
 System.*out*.println("Автор: " + ((JSONObject) book).get("author"));  
 System.*out*.println("Год издания: " + ((JSONObject) book).get("year"));  
 });  
  
 }catch (Exception e){  
 e.printStackTrace();  
 }  
 }  
}

* + 1. Добавление книги

package Lab10.Task3;  
  
import org.json.simple.JSONArray;  
import org.json.simple.JSONObject;  
import org.json.simple.parser.JSONParser;  
  
import java.io.FileReader;  
import java.io.FileWriter;  
  
public class JSONAdd {  
 public static void main(String[] args) {  
 var path = "src/Lab10/Task1/example.json";  
 try {  
 JSONParser parser = new JSONParser();  
 Object obj = parser.parse(new FileReader(path));  
 JSONObject jsonObject = (JSONObject) obj;  
  
 JSONArray jsonArray = (JSONArray) jsonObject.get("books");  
 JSONObject newBook = new JSONObject();  
 newBook.put("title", "Новая книга");  
 newBook.put("author", "Новый автор");  
 newBook.put("year", 2023);  
 jsonArray.add(newBook);  
  
 try (FileWriter file = new FileWriter(path)){  
 file.write(jsonObject.toJSONString());  
 }  
  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}

* + 1. Удаление книги

package Lab10.Task3;  
  
import org.json.simple.JSONArray;  
import org.json.simple.JSONObject;  
import org.json.simple.parser.JSONParser;  
  
import java.io.FileReader;  
import java.io.FileWriter;  
import java.util.Iterator;  
  
public class JSONRemove {  
 public static void main(String[] args) {  
 var path = "src/Lab10/Task1/example.json";  
 try {  
 JSONParser parser = new JSONParser();  
 Object obj = parser.parse(new FileReader(path));  
 JSONObject jsonObject = (JSONObject) obj;  
  
 JSONArray jsonArray = (JSONArray) jsonObject.get("books");  
 String title = "Новая книга";  
 Iterator iterator = jsonArray.iterator();  
 while (iterator.hasNext()) {  
 JSONObject book = (JSONObject) iterator.next();  
 if (title.equals(book.get("title"))) {  
 iterator.remove();  
 }  
 }  
  
 try (FileWriter file = new FileWriter(path)){  
 file.write(jsonObject.toJSONString());  
 }  
  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}

* 1. Доработки HTML парсера
     1. Запись полученных данных в файл, обработка ошибок при получении HTML-кода страницы, например вывод сообщения об ошибке и попытка переподключения к сайту.

package Lab10.Task4;  
  
import org.apache.poi.ss.usermodel.Row;  
import org.apache.poi.xssf.usermodel.XSSFSheet;  
import org.apache.poi.xssf.usermodel.XSSFWorkbook;  
import org.jsoup.HttpStatusException;  
import org.jsoup.Jsoup;  
import org.jsoup.nodes.Document;  
import org.jsoup.nodes.Element;  
import org.jsoup.select.Elements;  
  
import java.io.FileOutputStream;  
  
public class HTMLParse3 {  
 public static void main(String[] args) {  
  
 String url = "https://itlearn.ru/first-steps";  
 String filePath = "src/Lab10/Task4/links.xlsx";  
  
 int maxAttempts = 3;  
 int attempt = 0;  
  
 XSSFWorkbook workbook = new XSSFWorkbook();  
 XSSFSheet sheet = workbook.createSheet("Товары");  
  
 while (attempt < maxAttempts) {  
 try {  
 Document doc = Jsoup.*connect*(url).get();  
 Elements links = doc.select("a[href]");  
 for (int i = 0; i < links.size(); i++) {  
 Element link = links.get(i);  
 sheet.createRow(i).createCell(0).setCellValue(link.attr("abs:href"));  
 }  
  
 FileOutputStream outputStream = new FileOutputStream(filePath);  
 workbook.write(outputStream);  
 workbook.close();  
 outputStream.close();  
  
 break;  
 }catch (HttpStatusException e){  
 attempt++;  
 System.*out*.println("Попытка " + attempt + " не удалась. Ошибка: " + e.getMessage());  
 if (attempt == maxAttempts) {  
 System.*out*.println("Достигнуто максимальное количество попыток. Выход.");  
 }  
 }catch (Exception e){  
 e.printStackTrace();  
 break;  
 }  
 }  
 }  
}

1. Вывод

Получено представление о механизме парсинга в языке программирования Java.