Natalia Palej

A00279259@student.ait.ie

DOM AND SAX Parsing PROJECT DOCUMENTATION

Web Development 2.2

Software Design with Artificial Intelligence for Cloud Computing

Contents

[Introduction 2](#_Toc129810338)

[DOM Parser 2](#_Toc129810339)

[SAX Parser 4](#_Toc129810340)

[Conclusion 7](#_Toc129810341)

# Introduction

The application is a Java program that uses two different parsers (DOM and SAX) to read an XML file named GolfMajors.xml. The goal of the program is to parse the XML file and extract information about golf majors. The program consists of two main files, DOM.java and SAX.java, that needed to be edited to achieve its desired functionality. The program will be able to load the XML file into memory, parse it using either the DOM or SAX parser, and output the extracted data. A convenient "browse" button is provided to select the path to the desired file, and the user is given the option to choose which parser to use for parsing the file.

# DOM Parser

The DOM class is a Java program that loads an XML file into a DOM (Document Object Model) and parses its content to create an array list of Golf objects. The XML file contains information about golf major tournaments.

Graphical user interface, application

Description automatically generated

DOM Figure 1

The loadXMLFileIntoDOM() method loads an XML file into a DOM (Document Object Model) using the DocumentBuilder and DocumentBuilderFactory classes. It parses the XML file and stores the result in the 'dom' object. If there is an error, it prints out an error message.

Chart, scatter chart

Description automatically generated

DOM Figure 2

Text

Description automatically generated

DOM Figure 3

The parseTheDOM() parses the DOM to retrieve all elements with a tag name "major". It then loops through each of the elements, creates a Golf object by calling the getGolf() method and adds it to an ArrayList called golfMajors.

Text

Description automatically generated

DOM Figure 4

getTextValue() method retrieves the text value from a given XML element based on the provided tag name. It searches for the first occurrence of the tag name in the element and returns its corresponding text value. If the tag name is not found or the text value is empty, it returns null.

A picture containing chart

Description automatically generated

DOM Figure 5

The parsed data is then stored in the ArrayList golfMajors and method outputData() displays the contents of XML file in the GUI.

A picture containing chart

Description automatically generated

DOM Figure 6

Graphical user interface, application

Description automatically generated

DOM Figure 7

The program is designed to be easily modifiable, allowing the user to change the input file name by passing a fileName parameter to the runDOM() method.

# SAX Parser

SAX (Simple API for XML) parser is a widely used Java-based API for processing XML documents. SAX parser is highly efficient and is well-suited for processing large XML documents as it reads the document sequentially without loading the entire document into memory. The SAX.java file contains the implementation of the SAX parser that is used to parse an XML file containing data on golf majors.

A picture containing table

Description automatically generated

SAX Figure 1

startElement() method is an event handler that gets called when the start of an XML element is reached during parsing. It checks which tag it is in, resets temporary values, and sets values of the Golf object based on attributes or element names. It also sets boolean value to indicate which element the parser came across.

Text

Description automatically generated

SAX Figure 2

endElement() method is called when an end tag is encountered in the XML file being parsed. The method checks which tag has ended and sets the corresponding property of the tempGolf object accordingly. If the "major" tag has ended, it means that a Golf object has been fully parsed and it is added to the list of Golf objects in the golfMajors ArrayList.

A picture containing text

Description automatically generated

SAX Figure 3

The above method adds the characters of the current XML element to a temporary variable called "tempMajor". The characters are passed in as an array of characters, and the start and length parameters indicate the starting index and the length of the characters to be added to the temporary variable.

Chart

Description automatically generated

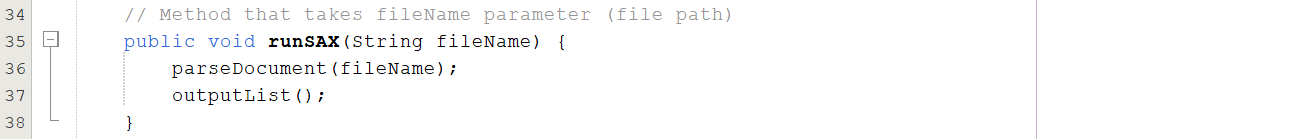
SAX Figure 4

A picture containing background pattern

Description automatically generated

SAX Figure 5

The extracted data is stored in objects of the Golf class and added to an ArrayList. The outputList() method is used to print the list of Golf objects. It simply iterates through all elements of golfMajors ArrayList and displays the result.



SAX Figure 6

Graphical user interface, application

Description automatically generated

SAX Figure 7

The runSAX() method takes the file name as a parameter and invokes the parseDocument() method to parse the XML file using the SAX parser.

# Conclusion

In conclusion, the project was quite challenging especially since the fileName could not be hardcoded and had to be passed as a parameter instead. However, I learned a lot from this project and enjoyed the process of working on it. It provided a great opportunity to explore Java programming language and its various features, such as using the Document Object Model (DOM) or SAX for parsing XML files. It was a rewarding experience that helped me develop my programming skills further.