

Exam Project



# XML Processing

*Advanced Programming Language*



Simge Haksal

## Table of Contents

Introduction .....	2
Process of The Project .....	3
SQL Conversion .....	3
XML Conversion .....	6
Conclusion .....	8
Reference List .....	8

## Figure List

- Figure 1. SQL File
- Figure 2. Postings Table
- Figure 3. Concurrent Programming Technique
- Figure 4. Getting Posts and Creating Text Node
- Figure 5. XML Conversion
- Figure 6. Database Connection
- Figure 7. Database Selection
- Figure 8. DOM Parser in Action
- Figure 9. Object of Generic Class
- Figure 10. Insert the Posts into Database
- Figure 11. Last update of DB

# XML Processing

## *Advanced Programming Language*

### Introduction

---

The aim of this project is writing a program that provides a conversation between a database and an XML file. By doing this, I used Java Language. Java is one of the object-oriented languages. In other words, everything is an object. Since I know the object-oriented concept, it was easy to master the project. I have already installed Java in my Windows, and the software tool is Eclipse.

The database part of the project was constituted on WampServer since refers to a software stack for the Microsoft Windows. WampServer is the best choice for MySQL database and PHP programming language. Also, I need phpMyAdmin to handle the administration of MySQL over the web because it supports a wide range of operations on MySQL. In this way, I could manage the database, tables, columns, relation, indexes, users, permissions and many other things.

To insert the XML file into the database by using a tree structure, I need to access the contents of the XML document. I could convert the value of one data type to a value of another data type by parsing, and the XML DOM defines a standard way for accessing and manipulating the XML document that I prepared. All XML elements can be accessed through the XML DOM. Thanks to the XML DOM, I could get, change, add, or delete XML elements. However, before an XML document can be accessed, it must be loaded into an XML DOM object. Shortly, I parsed a text string into an XML DOM object, and extract the information from it with JAVA.

I think now everything is clear. However, I had to cover three techniques by doing this project. I chose these; ADT, Parametric Types, Object Orientation, Generic Programming, and Concurrent Programming. Array, list, map, queue, set, stack, table, tree and vector is the example of ADT. I used array list in my project. For the parametric type or parameterized type, we can say that it is a type that is declared with a variable that is itself a type. I used this type with a type argument in one of my classes. Also, I want to point out that I needed to typecast to use the Array List on every time. As I said before, since I used Java programming language, I already used object orientation concept. In this way I completed one of the techniques in the restriction list.

Let's continue with Generic Programming. Java includes support for writing generic classes and methods that can operate on a variety of data types while often avoiding the need for explicit casts. The generics framework allows us to define a class in terms of a set of formal type parameters, which can then be used as the declared type for variables, parameters, and return values within the class definition. The classes like Array List use generics very well. I could write a method/class/interface once and use for any type we want.

The last technique is Concurrent Programming. I used multi-threads to use this technique. When a Java program starts up, a single thread is always created for the program. However, the new threads can be created. I created two threads. In this way, we can read more than XML file at the same time. To sum up, I realized all the techniques that I chose in this project.

## Process of The Project

### SQL Conversion

I prefer to divide the project into two different parts: Convert SQL and Convert XML. Let's start with converting SQL to XML. I built a ConvertSQL class and the helping classes are DatabaseSelection.

Firstly, I created **“database.sql”** file on MySQL Workbench.

```

1 insert into postings values (null,'2003-01-11 10:15:00','a@abc.com','Welcome',
2 'Welcome to the bulletin board',null);
3 insert into postings values (null,'2003-01-12 08:00:00','b@abc.com','Welcome',
4 'This is posting 2','1');
5 insert into postings values (null,'2003-01-13 09:00:00','c@abc.com','Welcome',
6 'This is posting 3','1');
7 insert into postings values (null,'2003-01-14 09:00:00','a@abc.com','Welcome',
8 'This is posting 4',null);
9 insert into postings values (null,'2003-01-15 09:00:00','a@abc.com','Welcome',
10 'This is posting 5',null);
11 insert into postings values (null,'2003-01-16 09:00:00','a@abc.com','Welcome',
12 'This is posting 6',null);
13 insert into postings values (null,'2003-01-17 09:00:00','c@abc.com','Welcome',
14 'This is posting 7','1:2');
15 insert into postings values (null,'2003-01-18 09:00:00','d@abc.com','Welcome',
16 'This is posting 8','1:2');
17 insert into postings values (null,'2003-01-19 09:00:00','a@abc.com','Welcome',
18 'This is posting 9','1:2:8');
19 insert into postings values (null,'2003-01-20 09:00:00','b@abc.com','Welcome',
20 'This is posting 10','6');

```

**Figure 1.** SQL File

In the proposal, I imagined building an examination file related to Bulletin Board. While I am doing this, I changed the concept a little bit. It is again a bulletin board; posting board. Now, it contains email addresses, date and time, post description, etc. I uploaded the database.sql file on localhost/phpMyAdmin under the **“xmlprocessing”** database on **“postings”** table. You can see the details about columns on the picture below.

<div>←→</div>		<div>▼</div>	postId	postDate	postedBy	postSubject	content	ancestorPath	
<input type="checkbox"/>	<div><div><div></div><div></div><div></div></div><div>Düzenle</div></div>	<div><div><div></div><div></div><div></div></div><div>Kopyala</div></div>	<div><div><div></div><div></div><div></div></div><div>Sil</div></div>	80	2003-01-20 09:00:00	b@abc.com	Welcome	This is posting 10	6
<input type="checkbox"/>	<div><div><div></div><div></div><div></div></div><div>Düzenle</div></div>	<div><div><div></div><div></div><div></div></div><div>Kopyala</div></div>	<div><div><div></div><div></div><div></div></div><div>Sil</div></div>	79	2003-01-19 09:00:00	a@abc.com	Welcome	This is posting 9	1:2:8
<input type="checkbox"/>	<div><div><div></div><div></div><div></div></div><div>Düzenle</div></div>	<div><div><div></div><div></div><div></div></div><div>Kopyala</div></div>	<div><div><div></div><div></div><div></div></div><div>Sil</div></div>	78	2003-01-18 09:00:00	d@abc.com	Welcome	This is posting 8	1:2
<input type="checkbox"/>	<div><div><div></div><div></div><div></div></div><div>Düzenle</div></div>	<div><div><div></div><div></div><div></div></div><div>Kopyala</div></div>	<div><div><div></div><div></div><div></div></div><div>Sil</div></div>	77	2003-01-17 09:00:00	c@abc.com	Welcome	This is posting 7	1:2
<input type="checkbox"/>	<div><div><div></div><div></div><div></div></div><div>Düzenle</div></div>	<div><div><div></div><div></div><div></div></div><div>Kopyala</div></div>	<div><div><div></div><div></div><div></div></div><div>Sil</div></div>	76	2003-01-16 09:00:00	a@abc.com	Welcome	This is posting 6	NULL
<input type="checkbox"/>	<div><div><div></div><div></div><div></div></div><div>Düzenle</div></div>	<div><div><div></div><div></div><div></div></div><div>Kopyala</div></div>	<div><div><div></div><div></div><div></div></div><div>Sil</div></div>	75	2003-01-15 09:00:00	a@abc.com	Welcome	This is posting 5	NULL
<input type="checkbox"/>	<div><div><div></div><div></div><div></div></div><div>Düzenle</div></div>	<div><div><div></div><div></div><div></div></div><div>Kopyala</div></div>	<div><div><div></div><div></div><div></div></div><div>Sil</div></div>	74	2003-01-14 09:00:00	a@abc.com	Welcome	This is posting 4	NULL
<input type="checkbox"/>	<div><div><div></div><div></div><div></div></div><div>Düzenle</div></div>	<div><div><div></div><div></div><div></div></div><div>Kopyala</div></div>	<div><div><div></div><div></div><div></div></div><div>Sil</div></div>	73	2003-01-13 09:00:00	c@abc.com	Welcome	This is posting 3	1
<input type="checkbox"/>	<div><div><div></div><div></div><div></div></div><div>Düzenle</div></div>	<div><div><div></div><div></div><div></div></div><div>Kopyala</div></div>	<div><div><div></div><div></div><div></div></div><div>Sil</div></div>	72	2003-01-12 08:00:00	b@abc.com	Welcome	This is posting 2	1
<input type="checkbox"/>	<div><div><div></div><div></div><div></div></div><div>Düzenle</div></div>	<div><div><div></div><div></div><div></div></div><div>Kopyala</div></div>	<div><div><div></div><div></div><div></div></div><div>Sil</div></div>	71	2003-01-11 10:15:00	a@abc.com	Welcome	Welcome to the bulletin board	NULL

**Figure 2.** Postings Table

Secondly, I created thread to write more than XML file at the same time.

```
class ThreadDemo extends Thread {  
    private Thread t;  
    private String threadName;  
  
    ThreadDemo( String name) {  
        threadName = name;  
        System.out.println("Creating " + threadName );  
    }  
}
```

**Figure 3.** *Concurrent Programming Technique*

After I created a thread, I wrote a function to run the thread. I created a document to write the XML conversion. Then, by using the array list, I appended child elements to the root element. It continues until all the elements in the database finishes.

```
private static Node getPosting(Document doc, String obj, String obj2, String obj3, String obj4) {  
    Element posting = doc.createElement("posting");  
    posting.appendChild(getElements(doc, posting, "postDate", obj));  
    posting.appendChild(getElements(doc, posting, "postedBy", obj2));  
    posting.appendChild(getElements(doc, posting, "postSubject", obj3));  
    posting.appendChild(getElements(doc, posting, "content", obj4));  
    return posting;  
}  
  
// Method creates text node  
private static Node getElements(Document doc, Element element, String name, String obj) {  
    Element node = doc.createElement(name);  
    node.appendChild(doc.createTextNode(obj));  
    return node;  
}
```

**Figure 4.** *Getting Posts and Creating Text Node*

Let's turn the run function with try and catch statements. It runs the threads. Until the end of the creating XML file, we can find it on the folder that I chose.



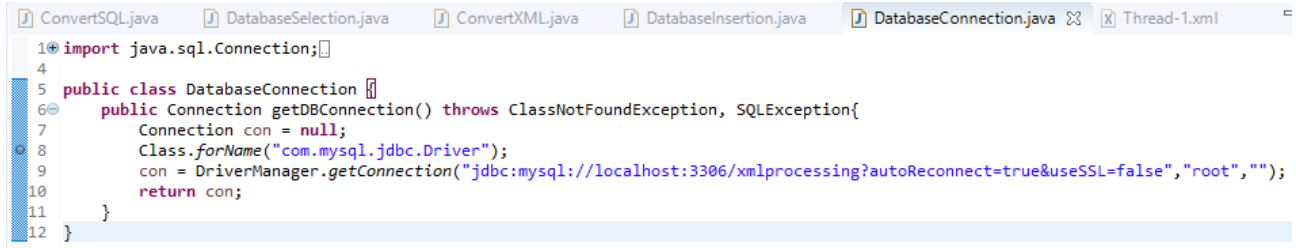
The screenshot shows a Java IDE console window with the following output:

```
<terminated> ConvertSQL (1) [Java Application] C:\Program Files\Java\jre1.8.0_211\bin\javaw.exe (16 May 2019 15:10:59)  
Creating Thread-1  
Starting Thread-1  
Creating Thread-2  
Starting Thread-2  
Running Thread-1  
Running Thread-2  
<?xml version="1.0" encoding="UTF-8" standalone="no"?>  
<Posts xmlns="http://localhost/phpmyadmin">  
  <posting>  
    <postDate>80</postDate>  
    <postedBy>2003-01-20 09:00:00.0</postedBy>  
    <postSubject>b@abc.com</postSubject>  
    <content>Welcome</content>  
  </posting>  
  <posting>  
    <postDate>79</postDate>  
    <postedBy>2003-01-19 09:00:00.0</postedBy>  
    <postSubject>a@abc.com</postSubject>  
    <content>Welcome</content>  
  </posting>  
</Posts>
```

**Figure 5.** *XML Conversion*

In the related folder, you can find **Thread-1** and **Thread-2** XML files. Both contain the same information, but I proved that I could write the data with different files simultaneously. In this way, one of the Concurrent Programming techniques is used successfully.

Of course, there are intermediate steps like selecting and taking the data from the DatabaseSelection class, and necessary database connections in the DatabaseConnection class.



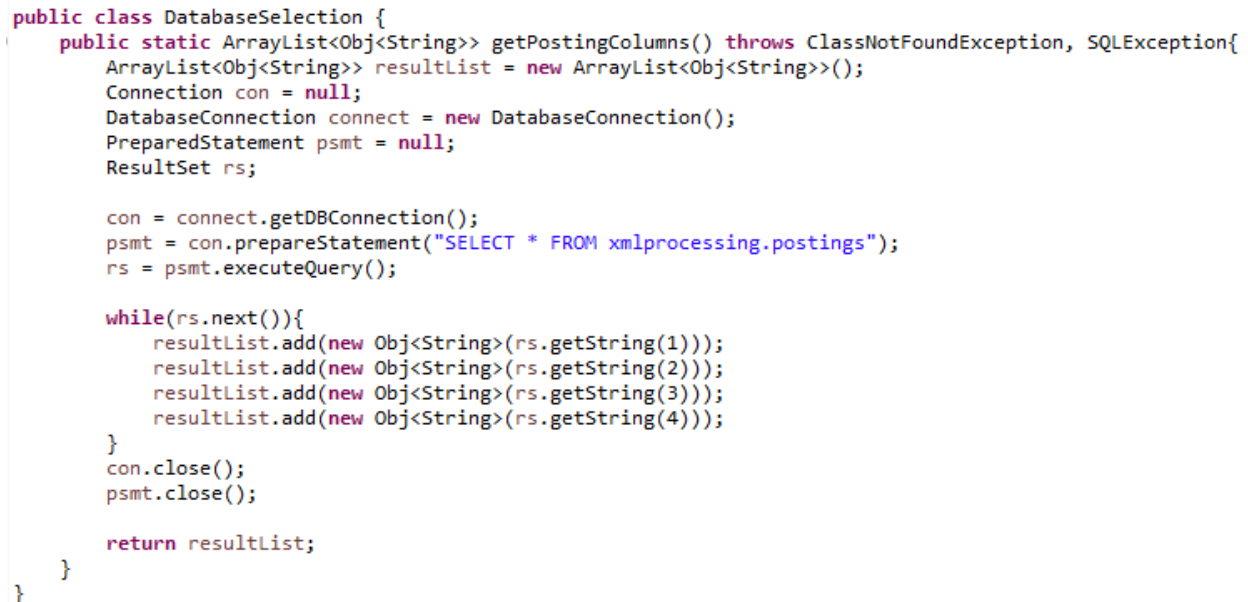
```

1 import java.sql.Connection;
4
5 public class DatabaseConnection {
6     public Connection getDBConnection() throws ClassNotFoundException, SQLException{
7         Connection con = null;
8         Class.forName("com.mysql.jdbc.Driver");
9         con = DriverManager.getConnection("jdbc:mysql://localhost:3306/xmlprocessing?autoReconnect=true&useSSL=false","root","");
10        return con;
11    }
12 }

```

**Figure 6.** Database Connection

I need this connection to run “**mysql-connector-java-5.1.39-bin.jar**”. Moreover, I used basic SQL statements like select -> from -> where. By using these necessary statements, I took each information from the *xmlprocessing* database.



```

public class DatabaseSelection {
    public static ArrayList<Obj<String>> getPostingColumns() throws ClassNotFoundException, SQLException{
        ArrayList<Obj<String>> resultList = new ArrayList<Obj<String>>();
        Connection con = null;
        DatabaseConnection connect = new DatabaseConnection();
        PreparedStatement psmt = null;
        ResultSet rs;

        con = connect.getDBConnection();
        psmt = con.prepareStatement("SELECT * FROM xmlprocessing.postings");
        rs = psmt.executeQuery();

        while(rs.next()){
            resultList.add(new Obj<String>(rs.getString(1)));
            resultList.add(new Obj<String>(rs.getString(2)));
            resultList.add(new Obj<String>(rs.getString(3)));
            resultList.add(new Obj<String>(rs.getString(4)));
        }
        con.close();
        psmt.close();

        return resultList;
    }
}

```

**Figure 7.** Database Selection

This is the end of the conversion process of SQL. To sum up, unlike CSV files and database tables, XML files are not naturally organized into rows and columns: XML is hierarchical. In this way, XML is like JSON. To convert XML to SQL then, MySQL helped me to work out how to flatten XML data into a tabular form. In the second section, I will continue with the conversion of XML file into SQL. I will prove everything by checking the tables on phpMyAdmin.

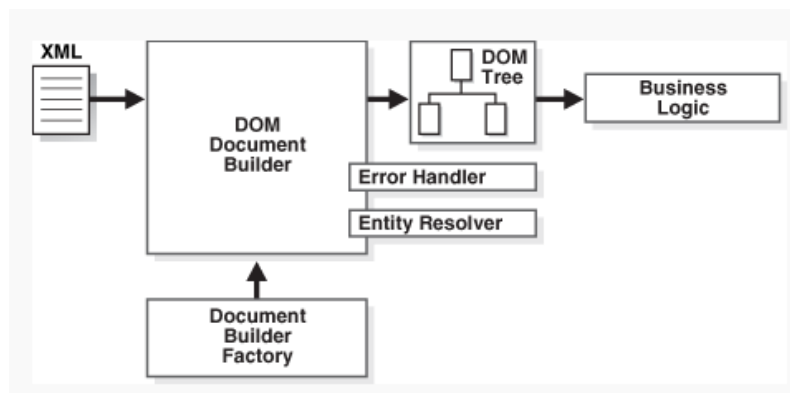
## XML Conversion

Let's start with the logic of DOM Parser. DOM parser is intended for working with XML as an object graph, tree like structure, in memory. So, it is so called **Document Object Model (DOM)**.

In first, the parser traverses the input XML file and creates **DOM** objects corresponding to the nodes in XML file. These DOM objects are linked together in a tree like structure. Once the parser is done with parsing process, we get this tree-like DOM object structure back from it. Then, we can traverse the DOM structure back and forth as we want to get, update, delete data from it.

For the coding side of DOM, the process continues in this order [1]:

- Importing XML related packages
- Creating a DocumentBuilder
- Creating a Document from a file or stream
- Validating the document structure
- Extracting the root element
- Examining attributes
- Examining sub-elements
- Reading XML with DOM parser
- Parsing unknown XML with DOM parser



**Figure 8.** DOM Parser in Action

Let's continue with the conversion process. In this part, I want to point out Generic Programming's techniques.

```

public class DatabaseInsertion {
    public static void setPostingColumns(NodeList List) throws ClassNotFoundException, SQLException{
        ArrayList<Obj<String>> postingList = new ArrayList<Obj<String>>();
        //take each nodes, convert and add into parts array
        for(int i = 0; i < List.getLength(); i++){
            String string = List.item(i).getTextContent();
            String[] parts = string.split("\n");
            for(int j = 1; j < parts.length; j++){
                postingList.add(new Obj<String>(parts[j].toString()));
            }
        }
    }
}
  
```

**Figure 9.** Object of Generic Class

We can also write generic functions that can be called with different types of arguments based on the type of arguments passed to generic method, the compiler handles each method. In this way, there will be no problem for the type of my items. It can be integer, float, character, string, there will be no problem. I kept all of them as string type [2].

I created ConvertXML and DatabaseInsertion classes to complete the conversion process. I split data from the XML files that I created for this process. I took each node, converted it, and added into parts array. Since I wrote database connection in the previous process, I used again the same class to provide the connection of server. I inserted each column into database. You can understand the process from the image below.

```
Connection con = null;
DatabaseConnection connect = new DatabaseConnection();
PreparedStatement psmt = null; //

con = connect.getDBConnection();
psmt = con.prepareStatement("INSERT INTO xmlprocessing.postings(postDate , postedBy, postSubject, content) VALUES (?, ?, ?, ?)");
int i = 0;
while(i < postingList.size()){
    psmt.setString(1, postingList.get(i++).toString());
    psmt.setString(2, postingList.get(i++).toString());
    psmt.setString(3, postingList.get(i++).toString());
    psmt.setString(4, postingList.get(i++).toString());
    psmt.executeUpdate();
}
con.close();
psmt.close();
```

**Figure 10.** *Insert the Posts into Database*

After that, I checked the phpMyAdmin and I saw that all the information in the result-1 and result-2 XML file now in the xmlprocessing database with previous data.

					postid	postDate	postedBy	postSubject	content	ancestorPath
<input type="checkbox"/>	Düzenle	Kopyala	Sil	100	2003-01-20 09:00:00	b@abc.com	Welcome	This is posting 10	NULL	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	99	2003-01-19 09:00:00	a@abc.com	Welcome	This is posting 9	NULL	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	98	2003-01-18 09:00:00	d@abc.com	Welcome	This is posting 8	NULL	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	97	2003-01-17 09:00:00	c@abc.com	Welcome	This is posting 7	NULL	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	96	2003-01-16 09:00:00	a@abc.com	Welcome	This is posting 6	NULL	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	95	2003-01-15 09:00:00	a@abc.com	Welcome	This is posting 5	NULL	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	94	2003-01-14 09:00:00	a@abc.com	Welcome	This is posting 4	NULL	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	93	2003-01-20 09:00:00	b@abc.com	Welcome	This is posting 10	NULL	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	92	2003-01-13 09:00:00	c@abc.com	Welcome	This is posting 3	NULL	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	91	2003-01-19 09:00:00	a@abc.com	Welcome	This is posting 9	NULL	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	90	2003-01-12 08:00:00	b@abc.com	Welcome	This is posting 2	NULL	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	89	2003-01-11 10:15:00	a@abc.com	Welcome	XXX	NULL	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	88	2003-01-18 09:00:00	d@abc.com	Welcome	This is posting 8	NULL	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	87	2003-01-17 09:00:00	c@abc.com	Welcome	This is posting 7	NULL	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	86	2003-01-16 09:00:00	a@abc.com	Welcome	This is posting 6	NULL	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	85	2003-01-15 09:00:00	a@abc.com	Welcome	This is posting 5	NULL	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	84	2003-01-14 09:00:00	a@abc.com	Welcome	This is posting 4	NULL	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	83	2003-01-13 09:00:00	c@abc.com	Welcome	This is posting 3	NULL	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	82	2003-01-12 08:00:00	b@abc.com	Welcome	This is posting 2	NULL	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	81	2003-01-11 10:15:00	a@abc.com	Welcome	XXX	NULL	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	80	2003-01-20 09:00:00	b@abc.com	Welcome	This is posting 10	8	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	79	2003-01-19 09:00:00	a@abc.com	Welcome	This is posting 9	1:2:8	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	78	2003-01-18 09:00:00	d@abc.com	Welcome	This is posting 8	1:2	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	77	2003-01-17 09:00:00	c@abc.com	Welcome	This is posting 7	1:2	
<input type="checkbox"/>	Düzenle	Kopyala	Sil	76	2003-01-16 09:00:00	a@abc.com	Welcome	This is posting 6	NULL	

**Figure 11.** *Last update of DB*



## Conclusion

---

I completed the project that I chose successfully. While I was doing this, I two different techniques. I already have known the object-oriented programming and its logic. Now, I also know the Concurrent Programming and the Generic Programming, at least their meanings and the way of use. I will show and explain all these processes in details while I will do the final exam on 17<sup>th</sup> of May. I hope you liked my project. You can check everything from my GitHub profile:

<https://github.com/simgehaksal>

## Reference List

---

1. Gupta, L. (n.d.), “HowToDoInJava”. Retrieved from <https://howtodoinjava.com/xml/read-xml-dom-parser-example/>
2. Singh, D. (n.d.), “Generics in Java”. Retrieved from <https://www.geeksforgeeks.org/generics-in-java/>