

Lab Sheet 03 – XML

Part 2: Creating Your First XML Document

books.xml

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<library>
```

```
  <book>
```

```
    <title>The Great Gatsby</title>
```

```
    <author>F. Scott Fitzgerald</author>
```

```
    <year>1925</year>
```

```
    <genre>Fiction</genre>
```

```
  </book>
```

```
  <book>
```

```
    <title>To Kill a Mockingbird</title>
```

```
    <author>Harper Lee</author>
```

```
    <year>1960</year>
```

```
    <genre>Fiction</genre>
```

```
  </book>
```

```
  <book>
```

```
    <title>1984</title>
```

```
    <author>George Orwell</author>
```

```
    <year>1949</year>
```

```
    <genre>Dystopian</genre>
```

```
  </book>
```

```
</library>
```

Part 3: Parsing XML in Java

XmlParser.java

```
package xml;

import org.w3c.dom.*;
import javax.xml.parsers.*;

public class XmlParser {

    public static void main(String[] args) {

        try {

            // Create a new DocumentBuilderFactory and DocumentBuilder
            DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
            DocumentBuilder builder = factory.newDocumentBuilder();

            // Parse the XML file
            Document document =
builder.parse("C:\\Users\\student\\Desktop\\EA\\XML\\src\\xml\\books.xml");

            // Normalize the document
            document.getDocumentElement().normalize();

            // Get the root element (library)
            NodeList nodeList = document.getElementsByTagName("book");

            // Loop through each book in the XML document
            for (int i = 0; i < nodeList.getLength(); i++) {

                Node node = nodeList.item(i);

                if (node.getNodeType() == Node.ELEMENT_NODE) {

                    Element element = (Element) node;

                    // Get and print the details of each book

                    String title = element.getElementsByTagName("title").item(0).getTextContent();

                    String author = element.getElementsByTagName("author").item(0).getTextContent();
```

```

        String year = element.getElementsByTagName("year").item(0).getTextContent();

        String genre = element.getElementsByTagName("genre").item(0).getTextContent();

        System.out.println("Title: " + title);

        System.out.println("Author: " + author);

        System.out.println("Year: " + year);

        System.out.println("Genre: " + genre);

        System.out.println("-----");

    }

}

} catch (Exception e) {

    e.printStackTrace();

}

```

The screenshot shows an IDE with two tabs: 'XML.java' and 'books.xml'. The active tab is 'XMLParser.java', which contains the following code:

```

1 package xml;
2 import org.w3c.dom.*;
3 import javax.xml.parsers.*;
4 public class XmlParser {
5     public static void main(String[] args) {
6         try {
7             // Create a new DocumentBuilderFactory and DocumentBuilder
8             DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
9             DocumentBuilder builder = factory.newDocumentBuilder();
10            // Parse the XML file
11            Document document = builder.parse(new File("D:/Assignment/XML/src/xml/books.xml"));
12            // Normalize the document
13            document.getDocumentElement().normalize();
14            // Get the root element (library)
15            NodeList nodeList = document.getElementsByTagName("book");
16            // Loop through each book in the XML document
17            for (int i = 0; i < nodeList.getLength(); i++) {
18                Node node = nodeList.item(i);
19                if (node.getNodeType() == Node.ELEMENT_NODE) {
20                    Element element = (Element) node;
21                    // Get and print the details of each book
22                    String title = element.getElementsByTagName("title").item(0).getTextContent();
23                    String author = element.getElementsByTagName("author").item(0).getTextContent();
24                    String year = element.getElementsByTagName("year").item(0).getTextContent();
25                    String genre = element.getElementsByTagName("genre").item(0).getTextContent();
26                    System.out.println("Title: " + title);
27                    System.out.println("Author: " + author);
28                    System.out.println("Year: " + year);
29                    System.out.println("Genre: " + genre);
30                }
31            }
32        } catch (Exception e) {
33            e.printStackTrace();
34        }
35    }
36 }

```

The 'Output' window shows the following results:

```

run:
Title: The Great Gatsby
Author: F. Scott Fitzgerald
Year: 1925
Genre: Fiction
-----
Title: To Kill a Mockingbird
Author: Harper Lee
Year: 1960
Genre: Fiction
-----
Title: 1984
Author: George Orwell
Year: 1949
Genre: Dystopian
-----
BUILD SUCCESSFUL (total time: 0 seconds)

```

The status bar at the bottom indicates '32:12' and 'INS Windows (CRLF)'.

Part 4: Modifying XML Data

XmlParser.java

```
package xml;

import java.io.File;
import org.w3c.dom.*;
import javax.xml.parsers.*;
import javax.xml.transform.Transformer;
import javax.xml.transform.TransformerFactory;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;

public class XmlParser {

    public static void main(String[] args) {

        try {

            // Create a new DocumentBuilderFactory and DocumentBuilder
            DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();

            DocumentBuilder builder = factory.newDocumentBuilder();

            // Parse the XML file
            Document document =
builder.parse("C:\\Users\\student\\Desktop\\EA\\XML\\src\\xml\\books.xml");

            // Normalize the document
            document.getDocumentElement().normalize();

            // Get the root element (library)
            NodeList nodeList = document.getElementsByTagName("book");

            // Modify the year of the first book
            Element firstBook = (Element) nodeList.item(0);

            firstBook.getElementsByTagName("year").item(0).setTextContent("2023");
```

```

// Save the modified document

TransformerFactory transformerFactory = TransformerFactory.newInstance();

Transformer transformer = transformerFactory.newTransformer();

DOMSource source = new DOMSource(document);

StreamResult result = new StreamResult(new
File("C:\\Users\\student\\Desktop\\EA\\XML\\src\\xml\\updated_books.xml"));

transformer.transform(source, result);

} catch (Exception e) {

    e.printStackTrace();

}

}

```

updated_books.xml

```

<?xml version="1.0" encoding="UTF-8" standalone="no"?><library>

<book>

    <title>The Great Gatsby</title>

    <author>F. Scott Fitzgerald</author>

    <year>2023</year>

    <genre>Fiction</genre>

</book>

<book>

    <title>To Kill a Mockingbird</title>

    <author>Harper Lee</author>

    <year>1960</year>

    <genre>Fiction</genre>

</book>

```

```
<book>

  <title>1984</title>

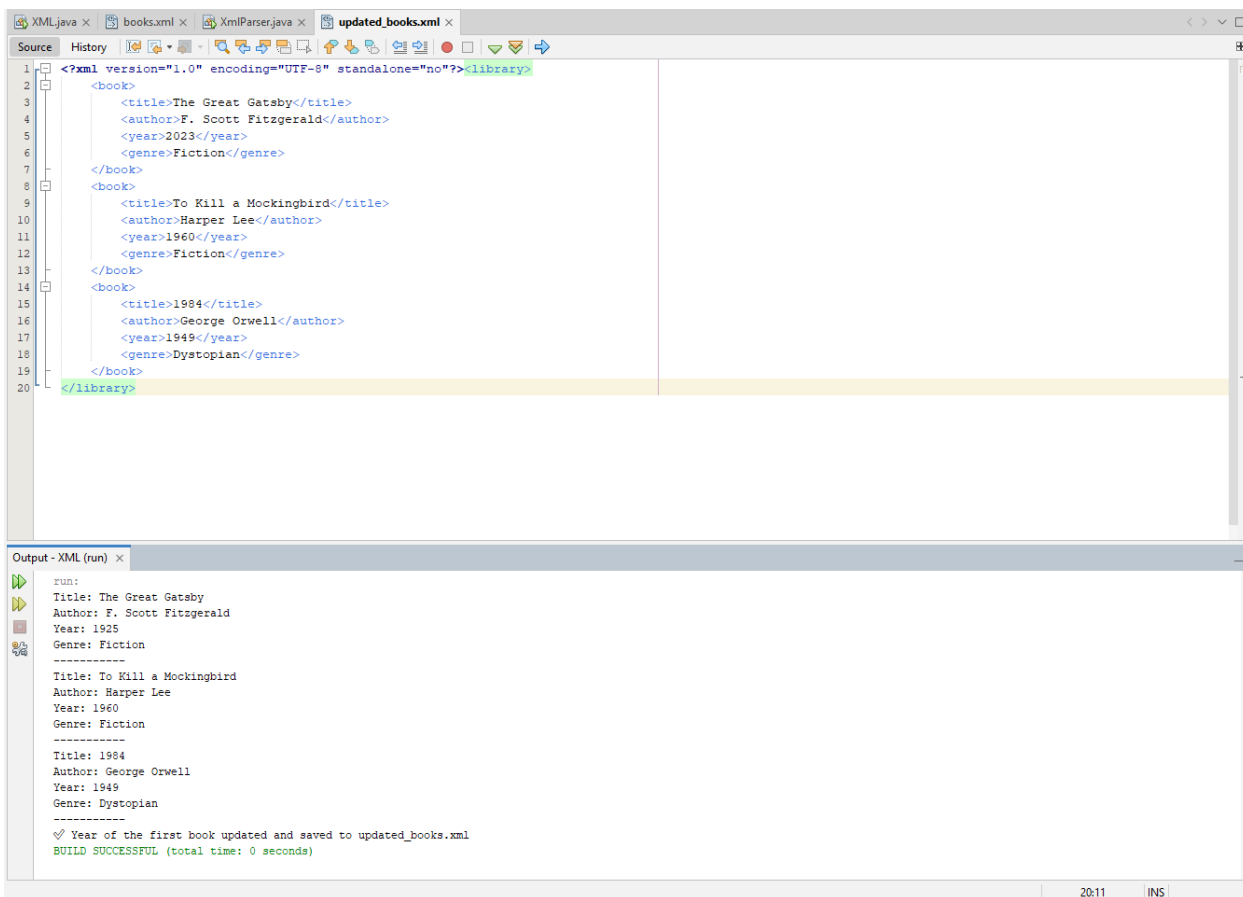
  <author>George Orwell</author>

  <year>1949</year>

  <genre>Dystopian</genre>

</book>

</library>
```



The screenshot shows an IDE with three tabs: XML.java, books.xml, and XmlParser.java. The active tab is updated_books.xml, which contains the following XML code:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?><library>
  <book>
    <title>The Great Gatsby</title>
    <author>F. Scott Fitzgerald</author>
    <year>2023</year>
    <genre>Fiction</genre>
  </book>
  <book>
    <title>To Kill a Mockingbird</title>
    <author>Harper Lee</author>
    <year>1960</year>
    <genre>Fiction</genre>
  </book>
  <book>
    <title>1984</title>
    <author>George Orwell</author>
    <year>1949</year>
    <genre>Dystopian</genre>
  </book>
</library>
```

The bottom panel shows the output of the XML parser, which is as follows:

```
run:
Title: The Great Gatsby
Author: F. Scott Fitzgerald
Year: 1925
Genre: Fiction
-----
Title: To Kill a Mockingbird
Author: Harper Lee
Year: 1960
Genre: Fiction
-----
Title: 1984
Author: George Orwell
Year: 1949
Genre: Dystopian
-----
✔ Year of the first book updated and saved to updated_books.xml
BUILD SUCCESSFUL (total time: 0 seconds)
```

The status bar at the bottom indicates the time is 20:11 and the mode is INS.