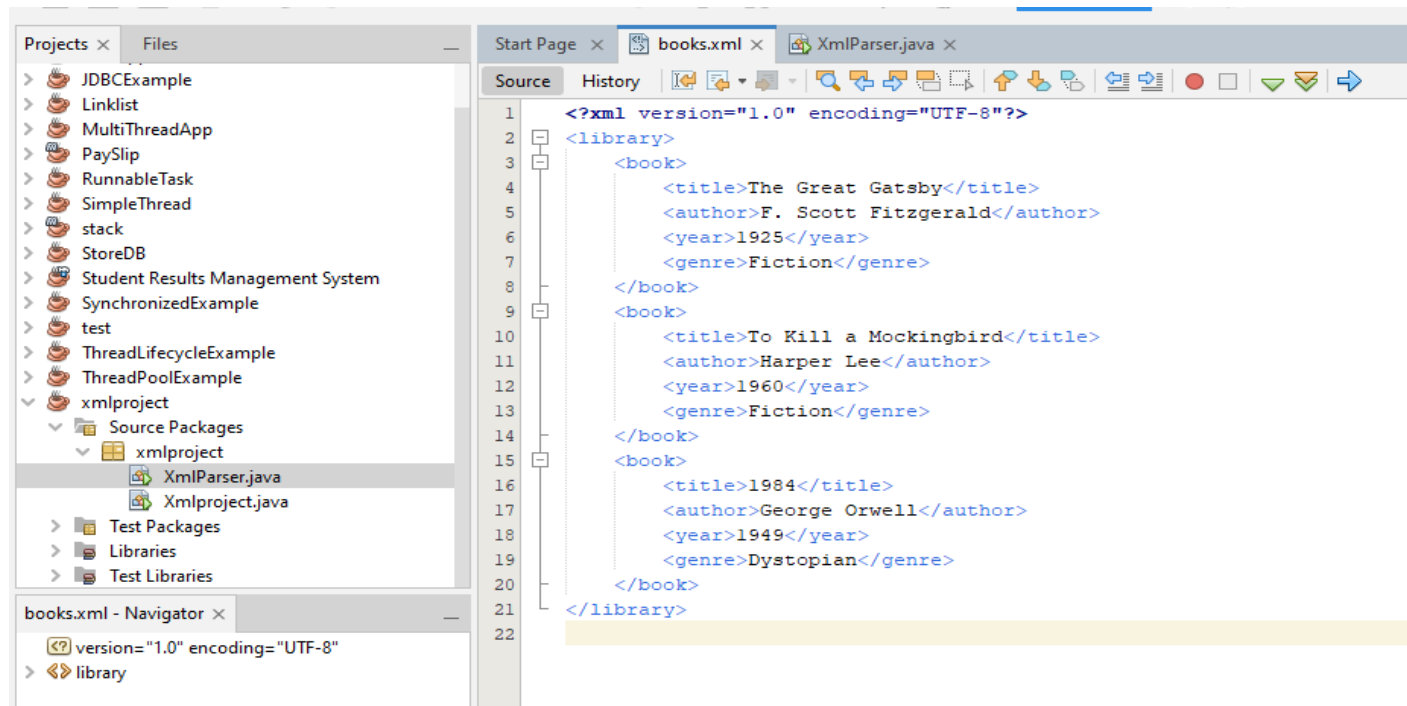


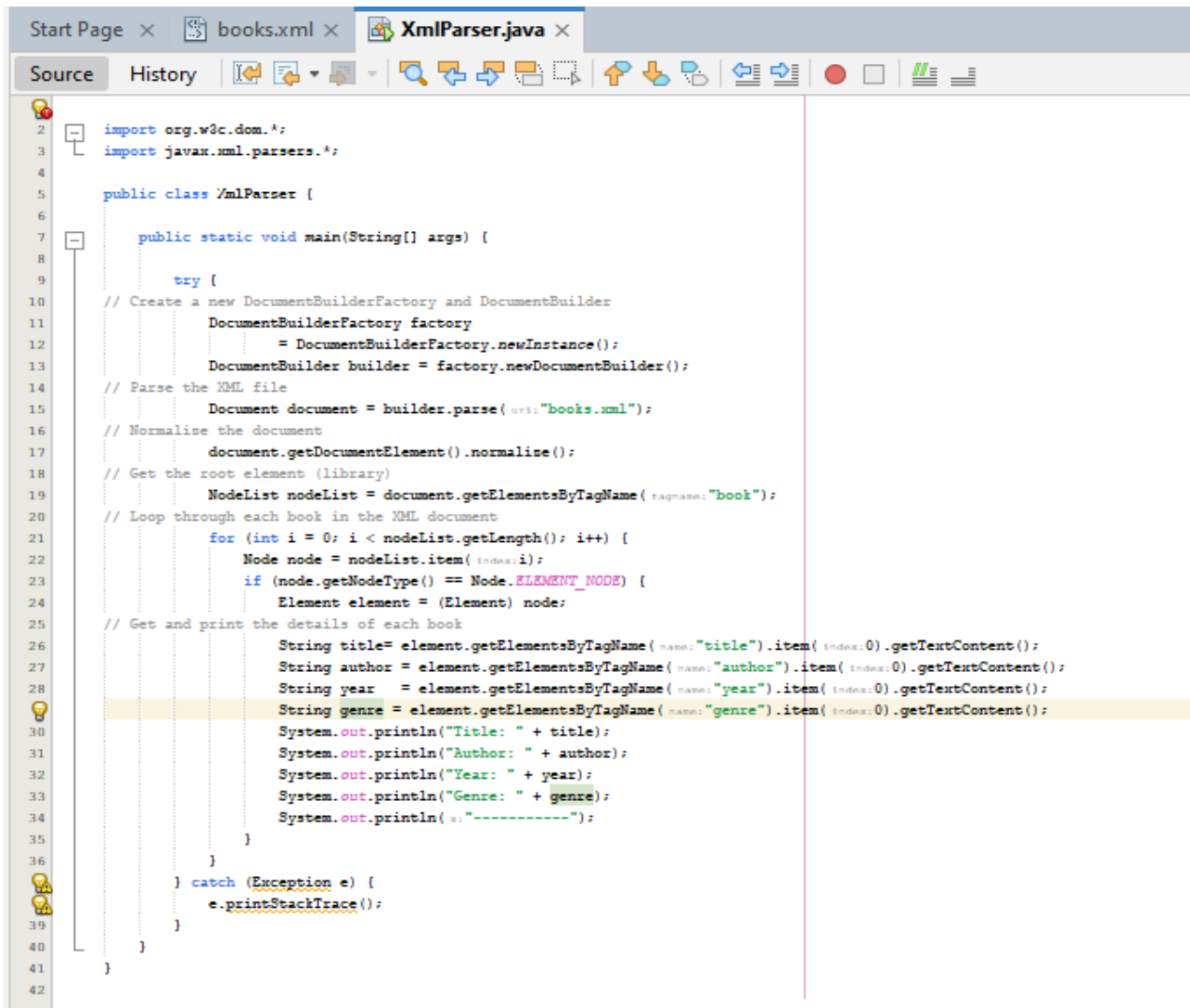
▪ Creating First XML Document

Create a new file named books.xml



- Parsing XML in Java

Create a Java Class for XML Parsing:



```
1  import org.w3c.dom.*;
2  import javax.xml.parsers.*;
3
4
5  public class XmlParser {
6
7      public static void main(String[] args) {
8
9          try {
10             // Create a new DocumentBuilderFactory and DocumentBuilder
11             DocumentBuilderFactory factory
12                 = DocumentBuilderFactory.newInstance();
13             DocumentBuilder builder = factory.newDocumentBuilder();
14             // Parse the XML file
15             Document document = builder.parse("books.xml");
16             // Normalise the document
17             document.getDocumentElement().normalize();
18             // Get the root element (library)
19             NodeList nodeList = document.getElementsByTagName("book");
20             // Loop through each book in the XML document
21             for (int i = 0; i < nodeList.getLength(); i++) {
22                 Node node = nodeList.item(i);
23                 if (node.getNodeType() == Node.ELEMENT_NODE) {
24                     Element element = (Element) node;
25                     // Get and print the details of each book
26                     String title= element.getElementsByTagName("title").item(0).getTextContent();
27                     String author = element.getElementsByTagName("author").item(0).getTextContent();
28                     String year  = element.getElementsByTagName("year").item(0).getTextContent();
29                     String genre = element.getElementsByTagName("genre").item(0).getTextContent();
30                     System.out.println("Title: " + title);
31                     System.out.println("Author: " + author);
32                     System.out.println("Year: " + year);
33                     System.out.println("Genre: " + genre);
34                     System.out.println("-----");
35                 }
36             }
37         } catch (Exception e) {
38             e.printStackTrace();
39         }
40     }
41 }
42
```

Output - xmlproject (run)



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

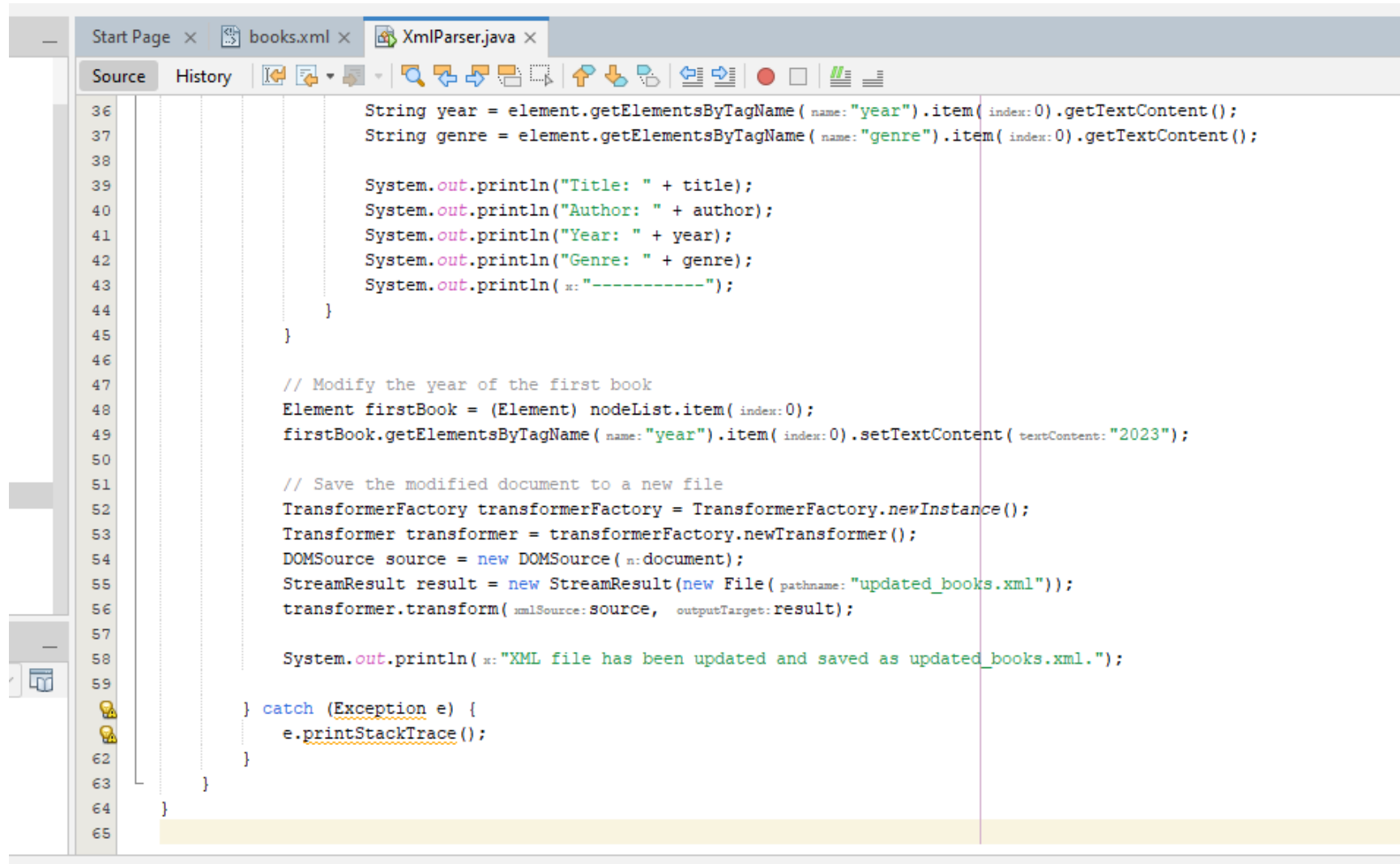
Author: George Orwell

Year: 1949

Genre: Dystopian

BUILD SUCCESSFUL (total time: 1 second)

Modifying XML Data

The image shows a screenshot of an IDE with three tabs: 'Start Page', 'books.xml', and 'XmlParser.java'. The 'XmlParser.java' tab is active, displaying Java code. The code includes imports for 'Document', 'Element', 'TransformerFactory', 'DOMSource', 'StreamResult', and 'File'. It defines a 'parse' method that reads an XML file, iterates through book elements, and prints their details. A 'modify' method is also present, which updates the year of the first book to '2023' and saves the modified document to a new file 'updated_books.xml'. The code is enclosed in a try-catch block to handle potential exceptions.

```
36 String year = element.getElementsByTagName( name: "year").item( index: 0).getTextContent();
37 String genre = element.getElementsByTagName( name: "genre").item( index: 0).getTextContent();
38
39 System.out.println("Title: " + title);
40 System.out.println("Author: " + author);
41 System.out.println("Year: " + year);
42 System.out.println("Genre: " + genre);
43 System.out.println( " : " + "-----");
44     }
45 }
46
47 // Modify the year of the first book
48 Element firstBook = (Element) nodeList.item( index: 0);
49 firstBook.getElementsByTagName( name: "year").item( index: 0).setTextContent( textContent: "2023");
50
51 // Save the modified document to a new file
52 TransformerFactory transformerFactory = TransformerFactory.newInstance();
53 Transformer transformer = transformerFactory.newTransformer();
54 DOMSource source = new DOMSource( n: document);
55 StreamResult result = new StreamResult( new File( pathname: "updated_books.xml"));
56 transformer.transform( xmlSource: source, outputTarget: result);
57
58 System.out.println( " : XML file has been updated and saved as updated_books.xml.");
59
60 } catch (Exception e) {
61     e.printStackTrace();
62 }
63
64 }
65
```

Output

Output - xmlproject (run)



run:

Updated XML content:

```
<?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>BUILD SUCCESSFUL (total time: 1 second)
```

updated_books.xml

```
18 public class StoreDB {  
1 <?xml version="1.0" encoding="UTF-8" standalone="no"?><library>  
2   <book>  
3     <title>The Great Gatsby</title>  
4     <author>F. Scott Fitzgerald</author>  
5     <year>2023</year>  
6     <genre>Fiction</genre>  
7   </book>  
8   <book>  
9     <title>To Kill a Mockingbird</title>  
10    <author>Harper Lee</author>  
11    <year>1960</year>  
12    <genre>Fiction</genre>  
13  </book>  
14  <book>  
15    <title>1984</title>  
16    <author>George Orwell</author>  
17    <year>1949</year>  
18    <genre>Dystopian</genre>  
19  </book>  
20 </library>
```