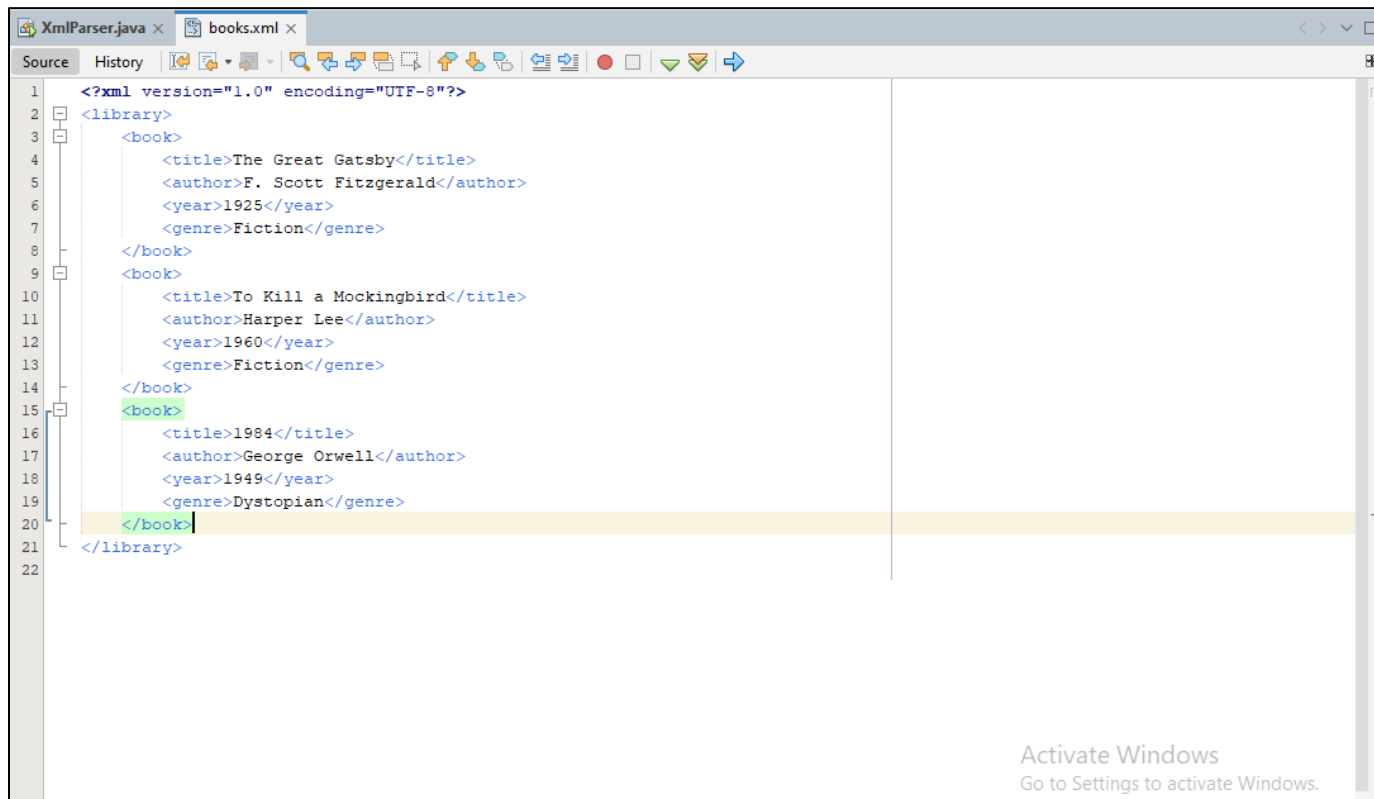


## XML Document

Create a new file named books.xml

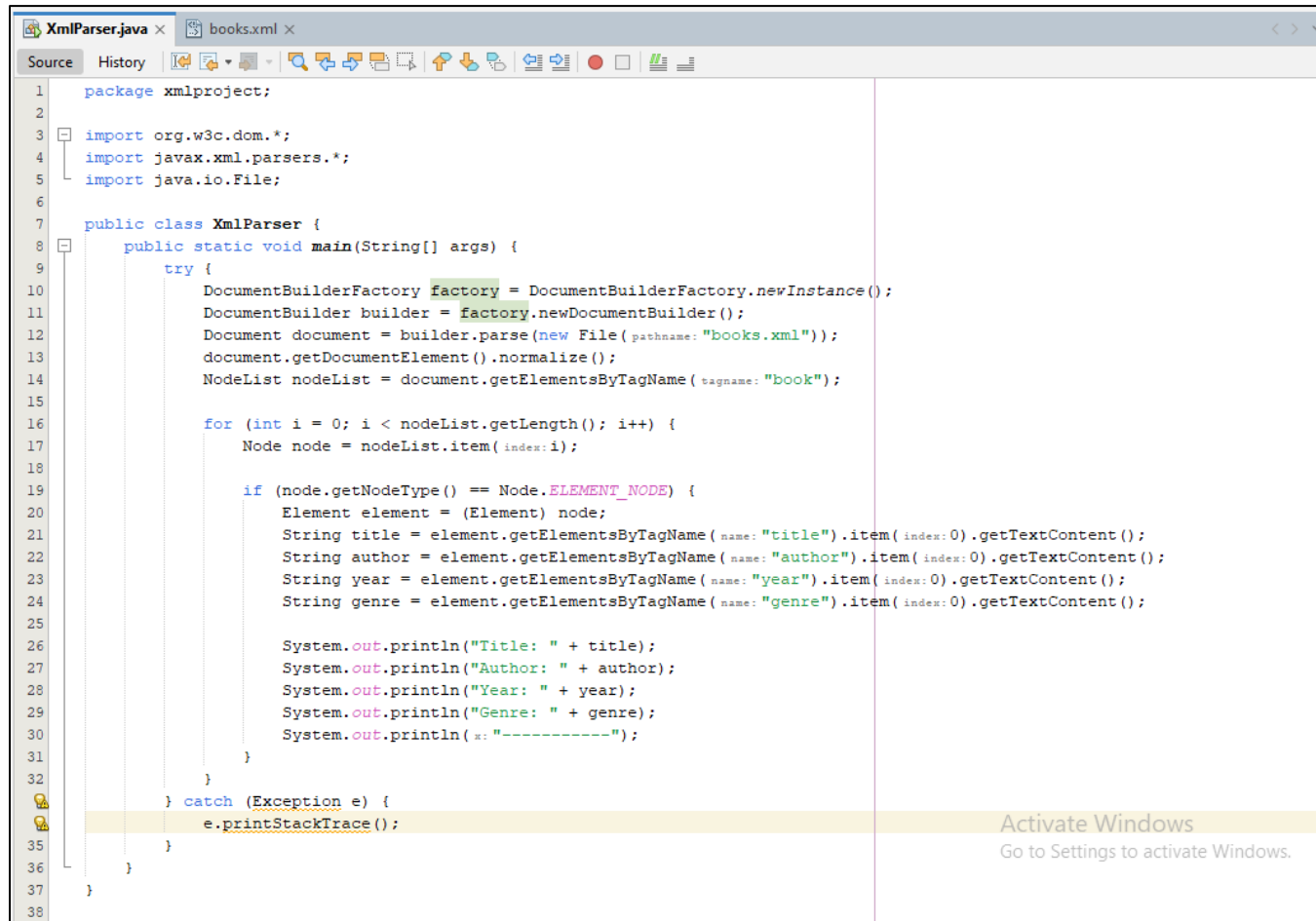


```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <library>
3   <book>
4     <title>The Great Gatsby</title>
5     <author>F. Scott Fitzgerald</author>
6     <year>1925</year>
7     <genre>Fiction</genre>
8   </book>
9   <book>
10    <title>To Kill a Mockingbird</title>
11    <author>Harper Lee</author>
12    <year>1960</year>
13    <genre>Fiction</genre>
14  </book>
15  <book>
16    <title>1984</title>
17    <author>George Orwell</author>
18    <year>1949</year>
19    <genre>Dystopian</genre>
20  </book>
21 </library>
22
```

Activate Windows  
Go to Settings to activate Windows.

## Parsing XML in Java

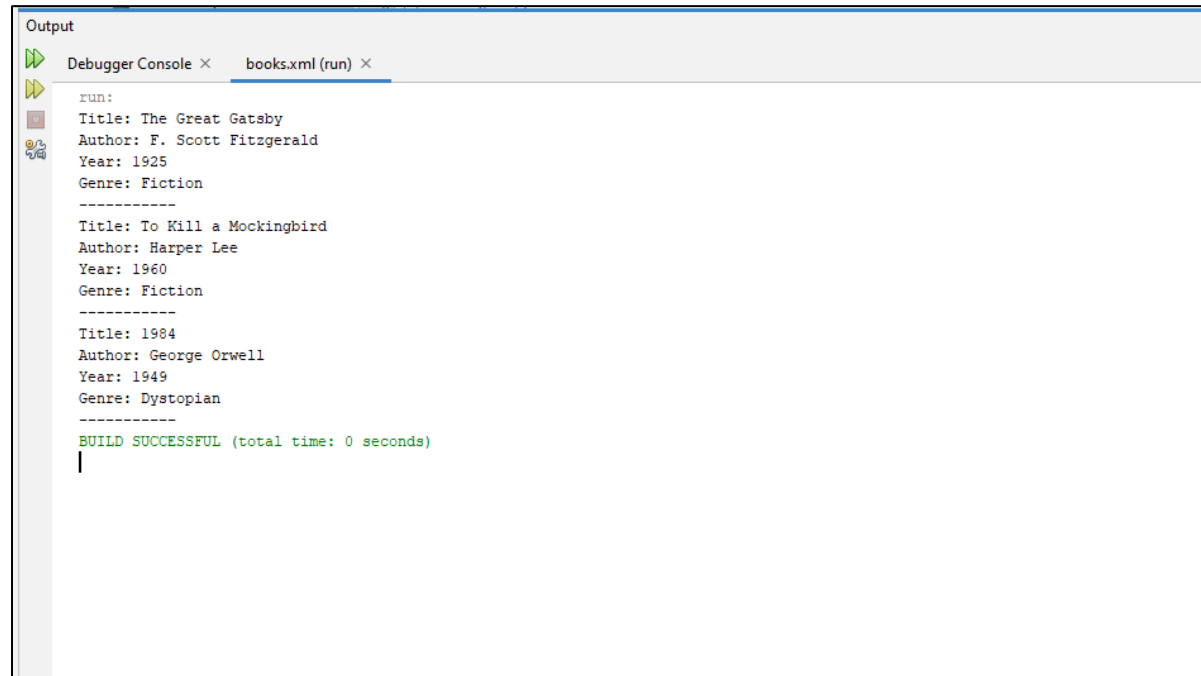
### Create a Java Class for XML Parsing:



```
1 package xmlproject;
2
3 import org.w3c.dom.*;
4 import javax.xml.parsers.*;
5 import java.io.File;
6
7 public class XmlParser {
8     public static void main(String[] args) {
9         try {
10             DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
11             DocumentBuilder builder = factory.newDocumentBuilder();
12             Document document = builder.parse(new File("books.xml"));
13             document.getDocumentElement().normalize();
14             NodeList nodeList = document.getElementsByTagName("book");
15
16             for (int i = 0; i < nodeList.getLength(); i++) {
17                 Node node = nodeList.item(i);
18
19                 if (node.getNodeType() == Node.ELEMENT_NODE) {
20                     Element element = (Element) node;
21                     String title = element.getElementsByTagName("title").item(0).getTextContent();
22                     String author = element.getElementsByTagName("author").item(0).getTextContent();
23                     String year = element.getElementsByTagName("year").item(0).getTextContent();
24                     String genre = element.getElementsByTagName("genre").item(0).getTextContent();
25
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                     System.out.println("-----");
31                 }
32             }
33         } catch (Exception e) {
34             e.printStackTrace();
35         }
36     }
37 }
38
```

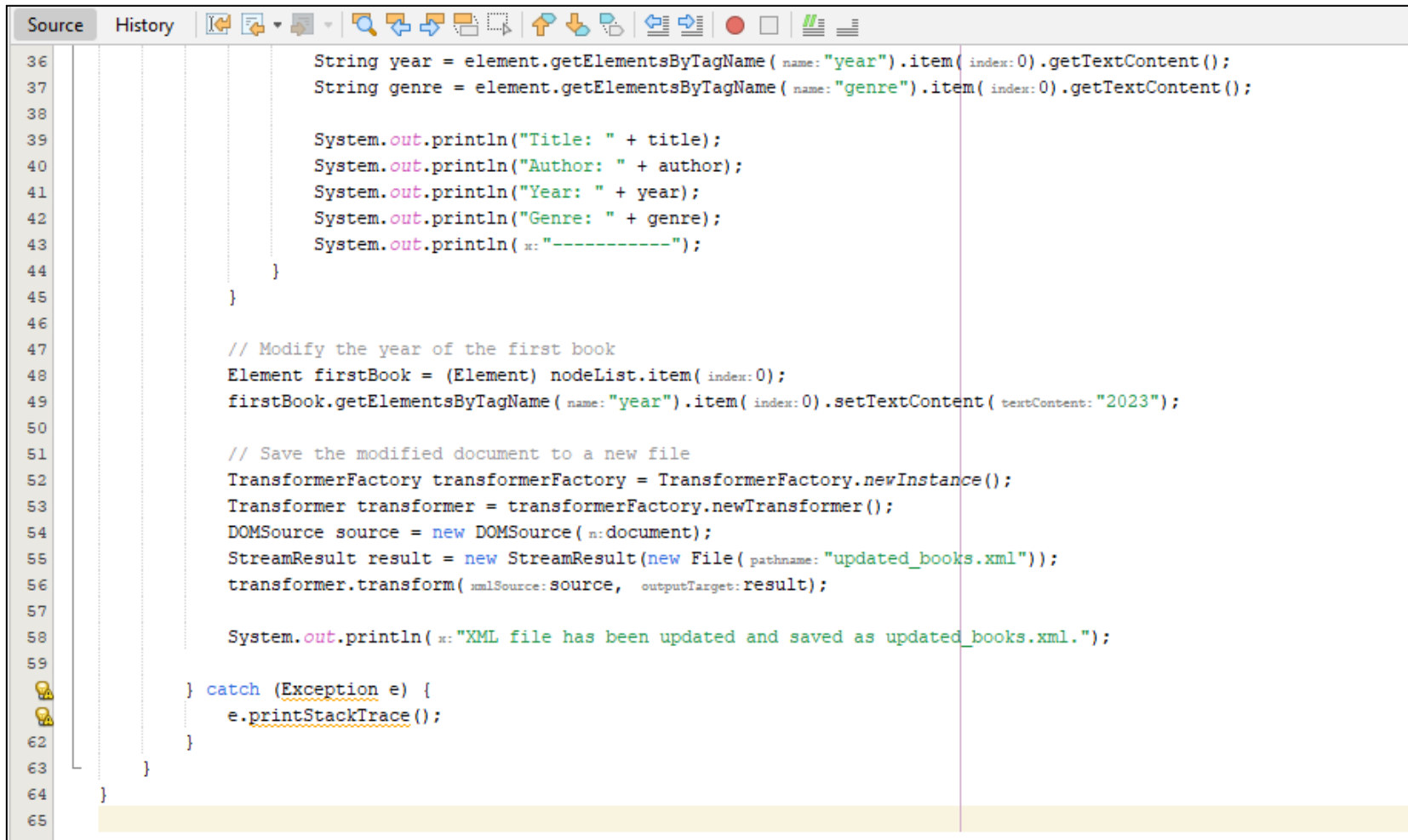
Activate Windows  
Go to Settings to activate Windows.

## Output



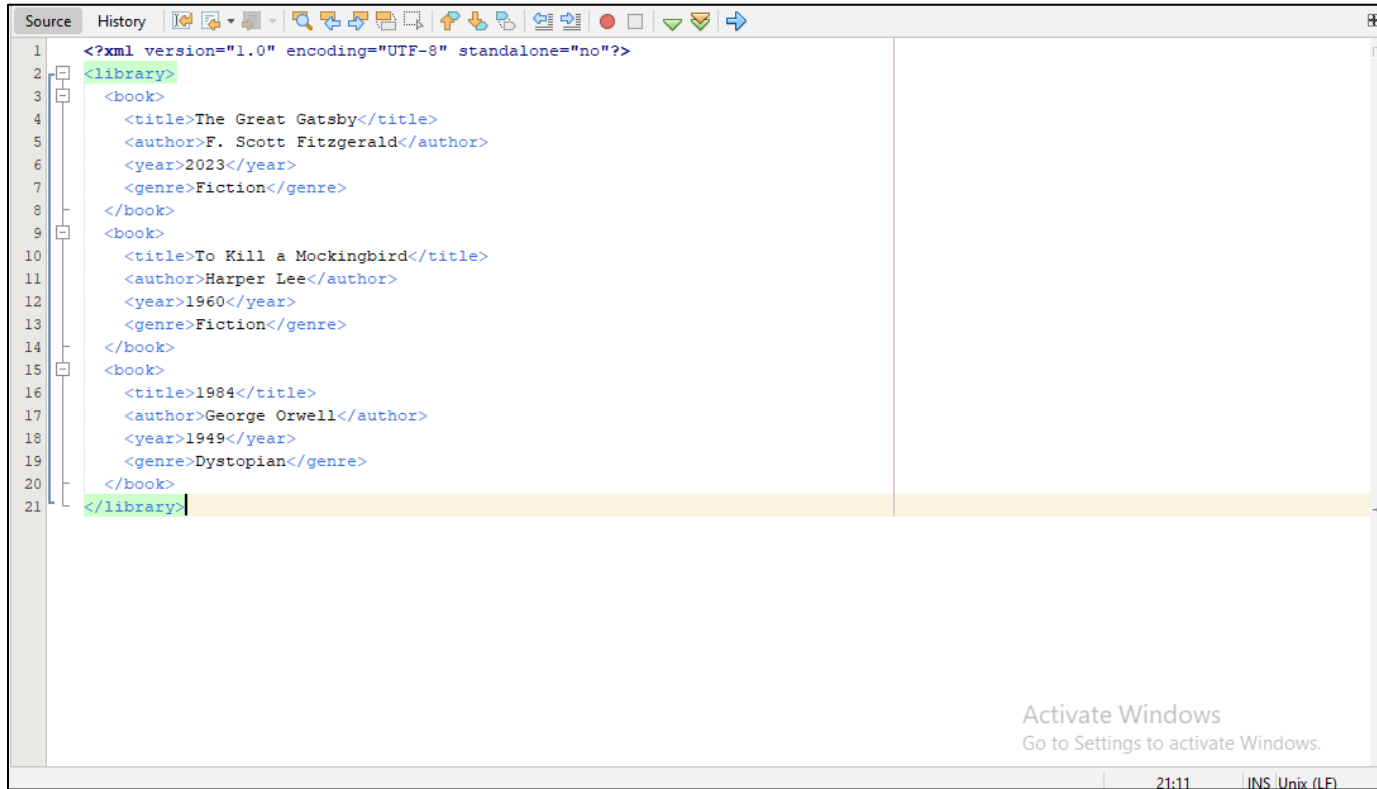
```
run:
Title: The Great Gatsby
Author: F. Scott Fitzgerald
Year: 1925
Genre: Fiction
-----
Title: To Kill a Mockingbird
Author: Harper Lee
Year: 1960
Genre: Fiction
-----
Title: 1984
Author: George Orwell
Year: 1949
Genre: Dystopian
-----
BUILD SUCCESSFUL (total time: 0 seconds)
|
```

## Modifying XML Data

A screenshot of an IDE window titled 'Source' with a 'History' tab. The code is in Java and uses the DOM API to modify an XML document. It includes comments for each major step: finding the first book, updating its year to 2023, and saving the document to a new file named 'updated\_books.xml'. The code also includes a try-catch block for handling exceptions. The line numbers 36 through 65 are visible on the left side of the editor.

```
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( "\n-----");
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( "\nXML file has been updated and saved as updated_books.xml.");
59
60 } catch (Exception e) {
61     e.printStackTrace();
62 }
63
64 }
65
```

## Updated books.xml

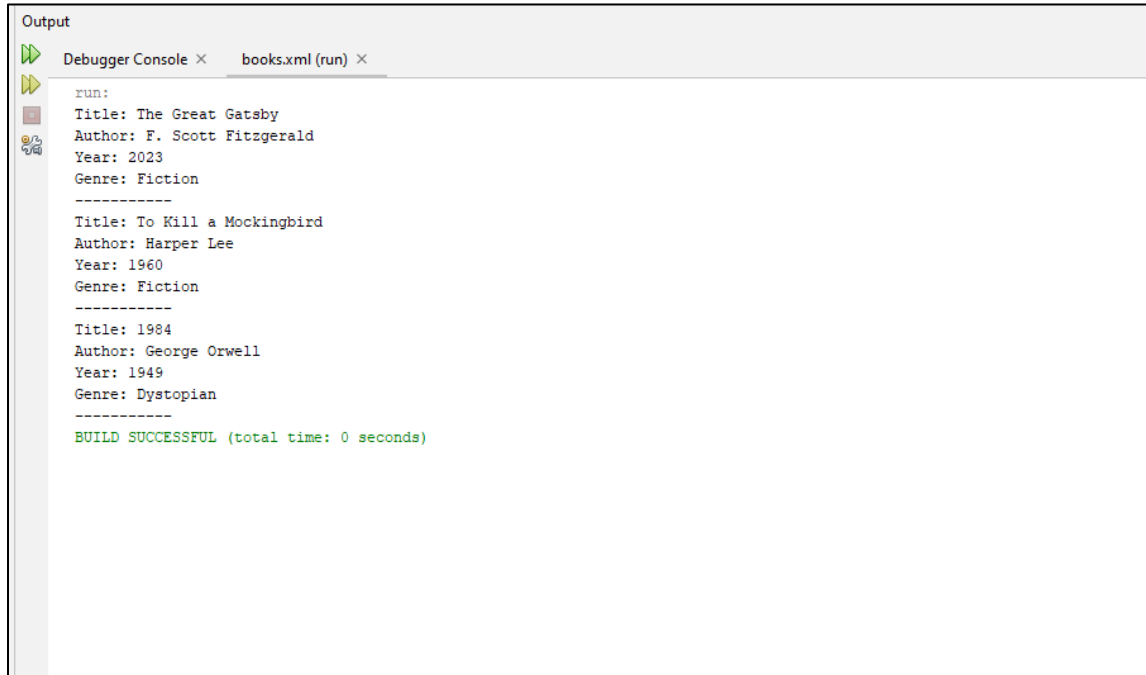


```
1 <?xml version="1.0" encoding="UTF-8" standalone="no"?>
2 <library>
3   <book>
4     <title>The Great Gatsby</title>
5     <author>F. Scott Fitzgerald</author>
6     <year>2023</year>
7     <genre>Fiction</genre>
8   </book>
9   <book>
10    <title>To Kill a Mockingbird</title>
11    <author>Harper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

Activate Windows  
Go to Settings to activate Windows.

21:11 INS Unix (LF)

## Output



```
run:
Title: The Great Gatsby
Author: F. Scott Fitzgerald
Year: 2023
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)
```