

**XML**

## **XML PARSERS**

```
package book1;

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("books.xml");
        Document document =
        builder.parse("C:\\Users\\User\\Desktop\\Book1\\src\\book1\\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();}}}
```

## **BOOK.XML**

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

```
<!--
```

Click <nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt> to change this license

Click <nbfs://nbhost/SystemFileSystem/Templates/XML/XMLDocument.xml> to edit this template

```
-->
```

```
<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>
```

# OUTPUT

The screenshot displays the Apache NetBeans IDE interface. The top toolbar includes standard menu options like File, Edit, View, and Run. The 'Projects' pane on the left shows a project named 'Book1' with sub-packages 'Source Packages' and 'book1'. The 'book1' package contains 'XmiParser.java' and 'books.xml'. The 'Source' editor shows the following Java code:

```
26 // Loop through each book in the XML document
27 for (int i = 0; i < nodeList.getLength(); i++) {
28     Node node = nodeList.item(i);
29
30     if (node.getNodeType() == Node.ELEMENT_NODE) {
31         Element element = (Element) node;
32
33         // Get and print the details of each book
34         String title = element.getElementsByTagName("title").item(0).getTextContent();
35         String author = element.getElementsByTagName("author").item(0).getTextContent();
36         String year = element.getElementsByTagName("year").item(0).getTextContent();
37         String genre = element.getElementsByTagName("genre").item(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 }
```

The 'Output - Book1 (run)' pane at the bottom shows the execution 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 Windows taskbar at the bottom shows the system clock at 07:13 PM on 07/13 PM.

**JDBC**

## **DATABASECONNECTION**

```
/*  
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this  
license  
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template  
*/  
  
package jdbcexample;  
  
  
import java.sql.Connection;  
import java.sql.DriverManager;  
import java.sql.SQLException;  
  
public class DatabaseConnection {  
    private static final String URL = "jdbc:mysql://localhost:3306/employee_db"; // Database URL  
    private static final String USER = "root"; // Your MySQL username  
    private static final String PASSWORD = "0523"; // Your MySQL password  
  
    public static Connection getConnection() throws SQLException {  
        try {  
            // Load the JDBC driver  
            Class.forName("com.mysql.cj.jdbc.Driver");  
            // Return the database connection  
            return DriverManager.getConnection(URL, USER, PASSWORD);  
        } catch (ClassNotFoundException | SQLException e) {  
            System.out.println("Connection failed: " + e.getMessage());  
            throw new SQLException("Failed to establish connection.");  
        }  
    }  
}
```

## **EMPLOYEE**

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this
 license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package jdbcexample;

public class Employee {
    private int id;
    private String name;
    private String position;
    private double salary;

    public Employee(int id, String name, String position, double salary) {
        this.id = id;
        this.name = name;
        this.position = position;
        this.salary = salary;
    }

    // Getters and setters
    public int getId() { return id; }
    public void setId(int id) { this.id = id; }

    public String getName() { return name; }
    public void setName(String name) { this.name = name; }

    public String getPosition() { return position; }
    public void setPosition(String position) { this.position = position; }

    public double getSalary() { return salary; }
    public void setSalary(double salary) { this.salary = salary; }

    @Override
    public String toString() {
        return "Employee{id=" + id + ", name=" + name + ", position=" +
        position + ", salary=" + salary + '}';
    }
}
```

## **EMPLOYEEDAO**

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this
 license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package jdbcexample;

import java.sql.*;
import java.util.ArrayList;
import java.util.List;

public class EmployeeDAO {

    // Create an employee
    public static void addEmployee(String name, String position, double salary)
    {
        String sql = "INSERT INTO employees (name, position, salary) VALUES (?, ?, ?)";

        try (Connection conn = DatabaseConnection.getConnection(); PreparedStatement stmt =
conn.prepareStatement(sql)) {

            stmt.setString(1, name);
            stmt.setString(2, position);
            stmt.setDouble(3, salary);

            int rowsAffected = stmt.executeUpdate();
            System.out.println("Employee added successfully. Rows affected: " + rowsAffected);
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    // Read all employees
    public static List<Employee> getAllEmployees() {
        List<Employee> employees = new ArrayList<>();
        String sql = "SELECT * FROM employees";

        try (Connection conn = DatabaseConnection.getConnection(); Statement stmt =
conn.createStatement(); ResultSet rs = stmt.executeQuery(sql)) {

            while (rs.next()) {
                Employee employee = new Employee(

                    rs.getInt("id"),
```



```

        rs.getString("name"),
        rs.getString("position"),
        rs.getDouble("salary")
    );
    employees.add(employee);
}
} catch (SQLException e) {
    e.printStackTrace();
}

return employees;
}

// Update an employee's information
public static void updateEmployee(int id, String name, String position,
double salary) {
    String sql = "UPDATE employees SET name = ?, position = ?, salary = ? WHERE id = ?";

    try (Connection conn = DatabaseConnection.getConnection();
        PreparedStatement stmt = conn.prepareStatement(sql)) {

        stmt.setString(1, name);
        stmt.setString(2, position);
        stmt.setDouble(3, salary);
        stmt.setInt(4, id);

        int rowsAffected = stmt.executeUpdate();
        System.out.println("Employee updated successfully. Rows affected: "
+ rowsAffected);
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

// Delete an employee
public static void deleteEmployee(int id) {
    String sql = "DELETE FROM employees WHERE id = ?";

    try (Connection conn = DatabaseConnection.getConnection();
        PreparedStatement stmt = conn.prepareStatement(sql)) {

        stmt.setInt(1, id);
        int rowsAffected = stmt.executeUpdate();
        System.out.println("Employee deleted successfully. Rows affected: "
+ rowsAffected);
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
}

```

## **JDBCEXAMPLE**

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this
 * license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Main.java to edit this template
 */
package jdbcexample;

import java.util.List;

public class JDBCExample {

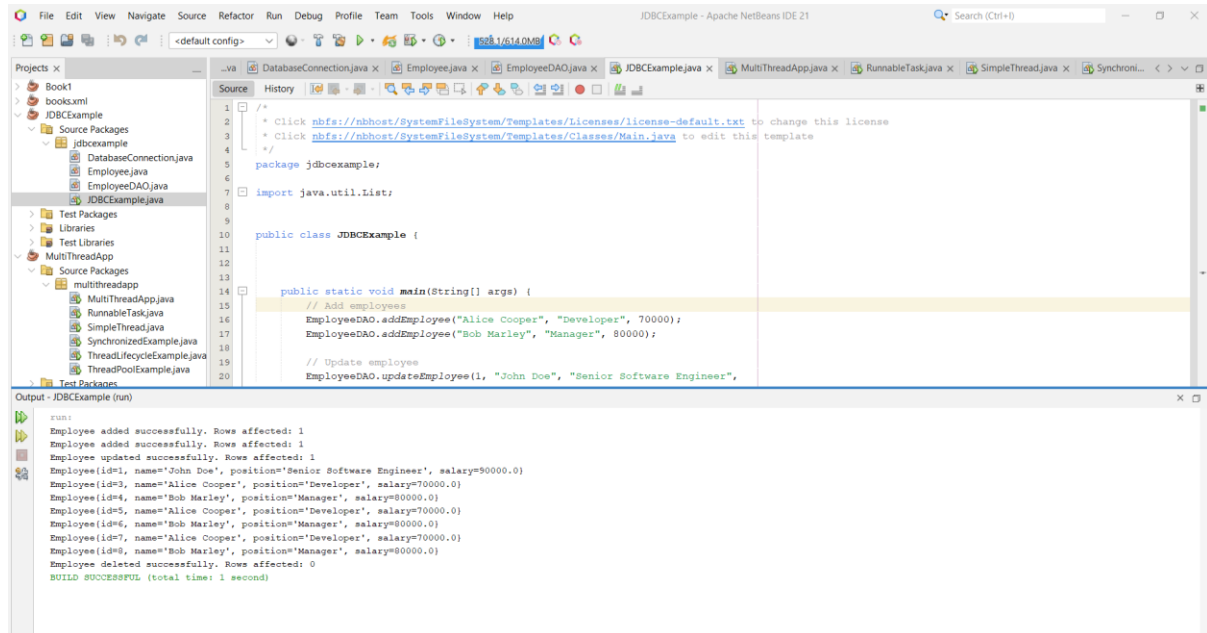
    public static void main(String[] args) {
        // Add employees
        EmployeeDAO.addEmployee("Alice Cooper", "Developer", 70000);
        EmployeeDAO.addEmployee("Bob Marley", "Manager", 80000);

        // Update employee
        EmployeeDAO.updateEmployee(1, "John Doe", "Senior Software Engineer",
90000);

        // Get all employees
        List<Employee> employees = EmployeeDAO.getAllEmployees();
        employees.forEach(System.out::println);

        // Delete employee
        EmployeeDAO.deleteEmployee(2);
    }
}
```

# OUTPUT



The screenshot displays the Apache NetBeans IDE interface. The main editor window shows the `JDBCExample.java` file. The code includes a package declaration, an import statement for `java.util.List`, and a `main` method. The `main` method contains three database operations: adding two employees, updating an employee, and deleting an employee. The output window at the bottom shows the results of these operations, including the number of rows affected and the details of the employees.

```
1  /*
2   * Click sbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click sbfs://nbhost/SystemFileSystem/Templates/Classes/Main.java to edit this template
4   */
5  package jdbcexample;
6
7  import java.util.List;
8
9
10 public class JDBCExample {
11
12
13
14     public static void main(String[] args) {
15         // Add employees
16         EmployeeDAO.addEmployee("Alice Cooper", "Developer", 70000);
17         EmployeeDAO.addEmployee("Bob Marley", "Manager", 80000);
18
19         // Update employee
20         EmployeeDAO.updateEmployee(1, "John Doe", "Senior Software Engineer",
```

Output - JDBCExample (run)

```
run:
Employee added successfully. Rows affected: 1
Employee added successfully. Rows affected: 1
Employee updated successfully. Rows affected: 1
Employee(id=1, name='John Doe', position='Senior Software Engineer', salary=80000.0)
Employee(id=3, name='Alice Cooper', position='Developer', salary=70000.0)
Employee(id=4, name='Bob Marley', position='Manager', salary=80000.0)
Employee(id=5, name='Alice Cooper', position='Developer', salary=70000.0)
Employee(id=6, name='Bob Marley', position='Manager', salary=80000.0)
Employee(id=7, name='Alice Cooper', position='Developer', salary=70000.0)
Employee(id=8, name='Bob Marley', position='Manager', salary=80000.0)
Employee deleted successfully. Rows affected: 0
BUILD SUCCESSFUL (total time: 1 second)
```

# **JAVA THREAD**

## **RUNNABLE TASK**

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this
 * license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package multithreadapp;

/**
 *
 * @author User
 */
public class RunnableTask implements Runnable {
    @Override
    public void run() {
        System.out.println(Thread.currentThread().getId() + " is executing the runnable task.");
    }

    public static void main(String[] args) {
        RunnableTask task1 = new RunnableTask();
        RunnableTask task2 = new RunnableTask();

        Thread thread1 = new Thread(task1);
        Thread thread2 = new Thread(task2);

        thread1.start(); // Starts thread1
        thread2.start(); // Starts thread2 }}
}
```

## **MULTITHREADAPP**

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this
 * license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package multithreadapp;

/**
 *
 * @author User
 */

    public class SimpleThread extends Thread {
@Override
public void run() {
System.out.println(Thread.currentThread().getId() + " is executing the thread.");
}
public static void main(String[] args) {

SimpleThread thread1 = new SimpleThread();
SimpleThread thread2 = new SimpleThread();

thread1.start(); // Starts thread1
thread2.start(); // Starts thread2
}
}
```

## **SYNCHRONIZEDEXAMPLE**

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this
 license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package multithreadapp;

/**
 *
 * @author User
 */
class Counter {
    private int count = 0;
    // Synchronized method to ensure thread-safe access to the counter
    public synchronized void increment() {
        count++;
    }
    public int getCount() {
        return count;
    }
}

public class SynchronizedExample extends Thread {
    private Counter counter;
    public SynchronizedExample(Counter counter) {
        this.counter = counter;
    }
    @Override
    public void run() {
        for (int i = 0; i < 1000; i++) {
            counter.increment();
        }
    }
    public static void main(String[] args) throws InterruptedException {
        Counter counter = new Counter();

        // Create and start multiple threads
        Thread thread1 = new SynchronizedExample(counter);
        Thread thread2 = new SynchronizedExample(counter);
        thread1.start();
        thread2.start();
        // Wait for threads to finish
        thread1.join(); thread2.join(); System.out.println("Final counter value: " + counter.getCount());
    }
}
```

## **ThreadPoolExample**

```
package multithreadapp;

/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this
license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */

/**
 *
 * @author User
 */
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
class Task implements Runnable {
    private int taskId;
    public Task(int taskId) {
        this.taskId = taskId;
    }
    @Override
    public void run() {
        System.out.println("Task " + taskId + " is being processed by " +
Thread.currentThread().getName());
    }
}
public class ThreadPoolExample {
    public static void main(String[] args) {
        // Create a thread pool with 3 threads
        ExecutorService executorService = Executors.newFixedThreadPool(3);
        // Submit tasks to the pool
        for (int i = 1; i <= 5; i++) {
            executorService.submit(new Task(i));
        }
        // Shutdown the thread pool
        executorService.shutdown();
    }
}
```



# Output

The screenshot displays the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, and Help. The title bar indicates the project is 'MultiThreadApp - Apache NetBeans IDE 21'. The toolbar shows various icons for file operations and running the application.

The left sidebar shows the 'Projects' view with a tree structure:

- Book1
- books.xml
- JDBCExample
- MultiThreadApp
  - Source Packages
    - multithreadapp
      - MultiThreadApp.java
      - RunnableTask.java
      - SimpleThread.java
      - SynchronizedExample.java
      - ThreadLifecycleExample.java
      - ThreadPoolExample.java
  - Test Packages
  - Libraries
  - Test Libraries
- servlet

The main editor window shows the source code of 'MultiThreadApp.java' in the 'Source' tab. The code is as follows:

```
1 package multithreadapp;
2
3 /*
4  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
5  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
6  */
7
8 /**
9  *
10  * @author User
11  */
12 import java.util.concurrent.ExecutorService;
13 import java.util.concurrent.Executors;
14 class Task implements Runnable {
15     private int taskId;
16     public Task(int taskId) {
17         this.taskId = taskId;
18     }
19     @Override
```

The bottom panel shows the 'Output - MultiThreadApp (run)' window with the following output:

```
run:
Task 3 is being processed by pool-1-thread-3
Task 1 is being processed by pool-1-thread-1
Task 2 is being processed by pool-1-thread-2
Task 4 is being processed by pool-1-thread-3
Task 5 is being processed by pool-1-thread-1
BUILD SUCCESSFUL (total time: 0 seconds)
```

The status bar at the bottom indicates '26:31/1:17' and 'INS'.