1: Setting Up JUnit

pom.xml

<?xml version="1.0" encoding="UTF-8"?>  
<project xmlns="http://maven.apache.org/POM/4.0.0"  
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  
 <modelVersion>4.0.0</modelVersion>  
  
 <groupId>org.example</groupId>  
 <artifactId>unit-test</artifactId>  
 <version>1.0-SNAPSHOT</version>  
  
 <properties>  
 <maven.compiler.source>24</maven.compiler.source>  
 <maven.compiler.target>24</maven.compiler.target>  
 <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>  
 </properties>  
 <dependencies>  
 <dependency>  
 <groupId>junit</groupId>  
 <artifactId>junit</artifactId>  
 <version>4.13.2</version>  
 <scope>test</scope>  
 </dependency>  
 </dependencies>  
</project>

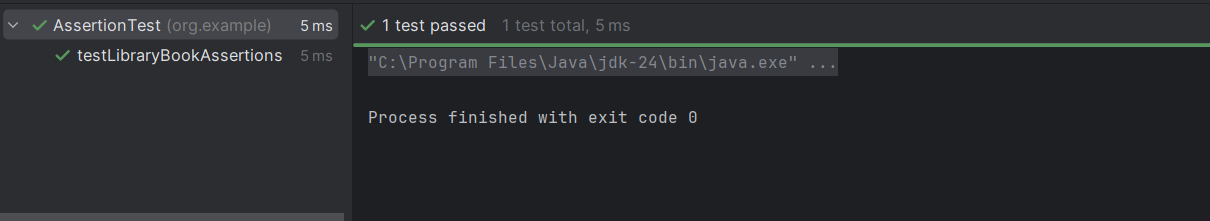
2. LibraryBook.java

package org.example;  
  
public class LibraryBook {  
 private String title;  
 private boolean checkedOut;  
 private String borrower;  
  
 public LibraryBook(String title) {  
 this.title = title;  
 this.checkedOut = false;  
 this.borrower = null;  
 }  
  
 public boolean isAvailable() {  
 return !checkedOut;  
 }  
  
 public String getTitle() {  
 return title;  
 }  
  
 public String getBorrower() {  
 return borrower;  
 }  
  
 public boolean checkOut(String borrowerName) {  
 if (checkedOut || borrowerName == null || borrowerName.isEmpty()) {  
 return false;  
 }  
 checkedOut = true;  
 borrower = borrowerName;  
 return true;  
 }  
  
 public boolean returnBook() {  
 if (!checkedOut) {  
 return false;  
 }  
 checkedOut = false;  
 borrower = null;  
 return true;  
 }  
}

3: Assertions in JUnit

AssertionsTest.java

package org.example;  
  
import org.junit.Test;  
import static org.junit.Assert.\*;  
  
public class AssertionTest {  
  
 @Test  
 public void testLibraryBookAssertions() {  
 LibraryBook book = new LibraryBook("Clean Code");  
  
 *assertTrue*(book.isAvailable());  
 *assertNull*(book.getBorrower());  
  
 *assertTrue*(book.checkOut("Alice"));  
 *assertFalse*(book.isAvailable());  
 *assertEquals*("Alice", book.getBorrower());  
  
  
 *assertFalse*(book.checkOut("Bob"));  
  
  
 *assertTrue*(book.returnBook());  
 *assertTrue*(book.isAvailable());  
 *assertNull*(book.getBorrower());  
  
 // Assert that returning again fails  
 *assertFalse*(book.returnBook());  
 }  
}



4. Arrange-Act-Assert (AAA) Pattern, Test Fixtures, Setup and Teardown Methods in JUnit

LibraryBookTest.java

package org.example;  
  
import org.junit.After;  
import org.junit.Before;  
import org.junit.Test;  
import static org.junit.Assert.\*;  
  
public class LibraryBookTest {  
 private LibraryBook book;  
  
 @Before  
 public void setUp() {  
 // Arrange  
 book = new LibraryBook("The Pragmatic Programmer");  
 }  
  
 @After  
 public void tearDown() {  
 // Teardown:   
 book = null;  
 }  
  
 @Test  
 public void testCheckOutSuccess() {  
 // Arrange done in setUp()  
  
 // Act  
 boolean result = book.checkOut("Alice");  
  
 // Assert  
 *assertTrue*(result);  
 *assertFalse*(book.isAvailable());  
 *assertEquals*("Alice", book.getBorrower());  
 }  
  
 @Test  
 public void testCheckOutFailsIfAlreadyCheckedOut() {  
 // Arrange  
 book.checkOut("Alice");  
  
 // Act  
 boolean result = book.checkOut("Bob");  
  
 // Assert  
 *assertFalse*(result);  
 *assertEquals*("Alice", book.getBorrower());  
 }  
  
 @Test  
 public void testReturnBookSuccess() {  
 // Arrange  
 book.checkOut("Alice");  
  
 // Act  
 boolean result = book.returnBook();  
  
 // Assert  
 *assertTrue*(result);  
 *assertTrue*(book.isAvailable());  
 *assertNull*(book.getBorrower());  
 }  
  
 @Test  
 public void testReturnBookFailsIfNotCheckedOut() {  
 // Act  
 boolean result = book.returnBook();  
  
  
 *assertFalse*(result);  
 *assertTrue*(book.isAvailable());  
 }  
}

