Name of the Course : Complete Java SE8 Developer Bootcamp

Level : Easy

Tool Stack : Java11 and Junit5

Problem Statement : Provide a code solution to validate PanCard using regular expression.

Description : Create two classes one PanCard class with panCard field with a parameterized constructor and another MainClass with two static methods

1. public static String validatePAN(PanCard panCard), which accepts PanCard object against following rules:

1. There must be eight characters.
2. First three letters must be alphabets followed by four digit number and ends with alphabet
3. All alphabets should be in capital case.

and returns “valid” or “invalid” after validating using regular expression.

2. public static void main method, for reading the PanCard Number from input devices and call the validatePAN method to test it.

Code:

**package** yaksha;

**public** **class** PanCard {

**private** String panCard;

**public** PanCard(String panCard) {

**super**();

**this**.panCard = panCard;

}

**public** String getPanCard() {

**return** panCard;

}

**public** **void** setPanCard(String panCard) {

**this**.panCard = panCard;

}

}

**package** yaksha;

**import** java.util.Scanner;

**public** **class** MainClass {

**public** **static** String validatePAN(PanCard panCard) {

**if** (panCard.getPanCard().matches("[A-Z]{3}[0-9]{4}[A-Z]{1}"))

**return** "valid";

**else**

**return** "invalid";

}

**public** **static** **void** main(String[] args) {

Scanner in = **new** Scanner(System.***in***);

System.***out***.println("Enter PanCard Number");

String s1 = in.nextLine();

PanCard panCard = **new** PanCard(s1);

String result = MainClass.*validatePAN*(panCard);

**if** (result.equals("valid")) {

System.***out***.println("Valid PanCard");

} **else** {

System.***out***.println("InValid PanCard");

}

in.close();

}

}

Junit Testing

**package** yaksha;

**import** **static** yaksha.TestUtils.*businessTestFile*;

**import** **static** yaksha.TestUtils.*currentTest*;

**import** **static** yaksha.TestUtils.*yakshaAssert*;

**import** org.junit.jupiter.api.Test;

**class** MainClassTest {

@Test

**void** testValidatePAN() **throws** Exception {

// Test will pass

PanCard panCard1 = **new** PanCard("ALD3245E");

*yakshaAssert*(*currentTest*(), (MainClass.*validatePAN*(panCard1).equals("valid") ? "true" : "false"),

*businessTestFile*);

}

}

**package** yaksha;

**import** java.io.File;

**import** java.io.FileWriter;

**import** java.io.IOException;

// boiler-plate code

**public** **class** TestUtils {

**public** **static** File *businessTestFile*;

**public** **static** File *boundaryTestFile*;

**public** **static** File *exceptionTestFile*;

**static** {

*businessTestFile* = **new** File("./output\_revised.txt");

*businessTestFile*.delete();

*boundaryTestFile* = **new** File("./output\_boundary\_revised.txt");

*boundaryTestFile*.delete();

*exceptionTestFile* = **new** File("./output\_exception\_revised.txt");

*exceptionTestFile*.delete();

}

**public** **static** **void** yakshaAssert(String testName, Object result, File file) **throws** IOException {

System.***out***.println("\n" + testName + "=" + result);

FileWriter writer = **new** FileWriter(file, **true**);

writer.append("\n" + testName + "=" + result);

writer.flush();

writer.close();

}

**public** **static** String currentTest() {

**return** Thread.*currentThread*().getStackTrace()[2].getMethodName();

}

}

pom.xml

<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 https://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>iiht.yaksha.pancard</groupId>

<artifactId>PanCardValidationJ11EQ1</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>PanCardValidationJ11EQ1</name>

<description>PanCardValidationJ11EQ1</description>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

<maven.compiler.source>1.8</maven.compiler.source>

<maven.compiler.target>${maven.compiler.source}</maven.compiler.target>

<junit.jupiter.version>5.5.2</junit.jupiter.version>

<junit.platform.version>1.5.2</junit.platform.version>

</properties>

<dependencies>

<!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->

<dependency>

<groupId>org.projectlombok</groupId>

<artifactId>lombok</artifactId>

<version>1.18.12</version>

<scope>provided</scope>

</dependency>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-engine</artifactId>

<version>${junit.jupiter.version}</version>

<scope>test</scope>

</dependency>

<dependency>

<groupId>org.junit.platform</groupId>

<artifactId>junit-platform-runner</artifactId>

<version>${junit.platform.version}</version>

<scope>test</scope>

</dependency>

</dependencies>

<build>

<plugins>

<plugin>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.8.1</version>

</plugin>

<plugin>

<artifactId>maven-surefire-plugin</artifactId>

<version>2.22.2</version>

</plugin>

</plugins>

</build>

</project>

output\_revised.txt

testValidatePAN=true

testing-PanCardValidationJ11EQ1.xml

<test-cases>

<cases xsi:type="java:com.assessment.data.TestCase">

<test-case-type>Functional</test-case-type>

<expected-ouput>true</expected-ouput>

<name>testValidatePAN</name>

<weight>10</weight>

<mandatory>true</mandatory>

<desc>Test to validate PanCard Number</desc>

</cases>

</test-cases>

Test Data1

Enter PanCard Number

ALD3245E

Valid PanCard

Learning outcome: Participant could able to learn how to use regular expression.