Day 6 Revisit (Java 8 Features)

- Functional Interface – An interface with only one abstract method (All the interfaces inside the java.util.function) – Predicate, Supplier, Consumer, Function
- Lambda – (Java's Arrow function/Inline function/Anonymous Function – abstract method definition of a functional interface)
- Annotations in java is introduced in Java 1.5
- Annotation is a metadata which can be added to properties, methods, class, parameters, constructor etc.,.
- @FunctionalInterface annotation is used to mark a Interface as a FunctionalInterface
- In java8, Interface can have default and static concrete methods.
- Functional interface can have n no of static and default methods but only one abstract method.
- Method Reference (Static Method Reference, Instance Method Reference, Constructor Reference)
- Method Reference – Simplified form of lambda (syntax is : Class/Object_Name::method_Name) Ex : System.out::println)
- Method Reference will not have brackets after the method name.
- Streams – Continuous flow of data
- Streams also works with group of objects not with single object.
- Intermediate operations using stream (Filter, sort, count, stream)
- Sync/Async process – Producer & Consumer
- Optional Classes
- AutoBoxing & Unboxing

Agenda

Java 11 & 17 New features

Sealed Classes

Pattern Matching

NIO – New Input Output API

Java 11 Features

- **Java 11 adds a few new methods to the *String* class**: *isBlank*, *lines*, *strip*, *stripLeading*, *stripTrailing,* and *repeat*.
- **new *readString* and *writeString* static methods from the *Files* class**
- **Executing .java in JVM directly (No need to create .class file in JAVA11)**
-

Exception Handling (Quick Revisit)

- Exception – Anomaly, Unwanted/Unexpected Event while creating or running the Java Program
- Error – Unavoidable situation which leads the program to stop

Exception Handling prevent pre-mature ending/closing of program

Top of Collection Hierarchy – Iterable

Top of Exception Hierarchy – Throwable

Keywords related to Exception Handling – (try, catch, throw, throws, finally)

Types of Exceptions

- Runtime Exception (Unchecked/Unhandled Exceptions) – It's not compulsory to exception handling code
- Compile time Exception (Checked/Handled Exceptions) – It's compulsory to use exception handling code

Different ways of Handling Exception

1) Using throws keyword – Assigning JVM to handle the exception (It's not recommended)
2) Using try/catch block (Recommended)

Different types of try/catch block

1) Try block should have a catch block and/or finally block.
2) Try block can have more than one catch block but only one finally block
3) The code written in finally block always get executed irrespective of the exception status
4) The best practice used is, write clean up code (closing all the resources) in finally block

Download and Install JDK17 if it is not already installed in your system

https://www.oracle.com/in/java/technologies/downloads/#jdk17-windows

Sealed Class – It's introduced in Java 15.

What – New keyword called "sealed" is introduced. – sealed class is a technique that limits the number of classes that can inherit the given class.

Access Modifier (private, package/default, protected, public)

Non-Access Modifier (final, static, abstract, volatile)

Singleton – Only one instance of a class. (It's a design pattern)

Sealed class – Restricting no of inherits the class can have.

- The inherited classes from sealed classes should be non-sealed/sealed/final
- Keyword used along with sealed classes sealed/non-sealed/permits & final
- It's possible to create object of sealed classes too.

Pattern Matching

- Regular Expressions (RegEx)
- Comparision
- Matching with the given pattern.

Email, Password validations

Search Operation, Manipulate Strings

1) Pattern – Compiled representation of regex.
2) Matcher
3) PatternSyntaxException

Interface

1) MatchResult

https://www.geeksforgeeks.org/java-program-to-check-for-a-valid-mobile-number/

Java NIO, NIO2

NIO – New Input Output Features

Character data type size in Java is 2 bytes (16 bits)

128 (0-255) (-128 to 127)

ASCII Code

American Standard Code for Information Interchange

JAVA, UniCode format (each character is represented using 2 byte data)

English Lang

26 (Upper case)

26 (lower case)

0-9 (

Java.io

ByteStream (uses 1 byte data – 8 bits) – (Input/OutputStream)

CharacterStream (uses 2 byte data – 16 bits) – [Reader/Writter]

Java.nio & nio2

https://www.baeldung.com/java-io-vs-nio

https://www.geeksforgeeks.org/introduction-to-java-nio-with-examples/