Week6 Agenda (Adv Java, Java8 Features & ORM Framework)

* Adv Java Topics (Java 8 Features) [Lambda, Functional Interface, time package & Optional Class]
* Multi-Threading concepts in Java, Logging (Level), Live & Dead Lock, Synchronize keyword
* ORM Framework (Hibernate) & JPA. XML mapping & Annotation Mapping
* HQL, Native SQL, Criteria
* Named Queries & Caching

Week5 Revisit (JavaScript, Ajax )

* JS is a Multi-paradigm, loosely typed/dynamically typed, high-level, web programming language.
* JS is case sensitive.
* Except the name java and few similar java syntax & keywords, it’s no way connected to Java
* Java and JavaScript is totally a different programming language.
* Variable data types, var, let, const, variable scopes,
* JS engine is present in each browser which is responsible for running the java code.
* AJAX [Asynchronous JavaScript and XML] – Helps to perform partial page reload.
* Ways of adding java script code to HTML document. A)Inline B) Internal C) External

Interface – Is a contract which enforces some rules on the implementing classes.

Interface will help you to achieve multiple inheritance in Java.

A class can extend only one other class but it can implement multiple interfaces.

Types of Interface

1. Simple/Normal Interface (All members are by default “public” & “abstract” - Concrete methods were not allowed in Interface before Java 8)
2. Marker Interface (An interface with no method at all – Serializable, Cloneable)
3. Functional Interface (An Interface with only one abstract method - Runnable) – default & static methods are allowed

4 types of Access Modifiers

1. Private ( with in the class only) – least accessible
2. Default/package – accessible with in the package only
3. Protected – accessible in the same package as well as derived class in other packages
4. Public (access anywhere) – Most accessible

Functional Interface

1. It should have only one abstract method
2. It can have n number of default and static method
3. It’s mainly used for defining a lambda & for Method reference
4. Adding @FucntionInterface annotation is optional [It should have only one abstract method]

Lambda (Is also called as arrow function)

* Arrow function in Java (using a special symbol ->)
* This is also called as slim arrow function
* Lambda means anonymous function (Nameless)
* Using lambda, we can perform multiple operations in a single line
* Using lambda, we can able to assign a method to the Object reference.
* Lambdas are used with collection [ Filtering, Sorting & iterating through the collection]
* Return keyword is optional
* Flower bracket (curly braces) are optional if it is only one statement in the body of lambda
* Using normal brackets also optional while passing a single argument.
* Lambda’s enable functional programming in JAVA

<https://www.javatpoint.com/java-lambda-expressions>

Optional Class

* The main use of Optional Class is to avoid “NullPointerException” at runtime.

Important Features of Java8

1. Lambda
2. FunctionalInterface
3. Streams
4. Date time API

Thread – Light Weight Process

Process -- Heavy Weight Thread

Thread enables multi-tasking in Java.

Multi-tasking – Doing /performing many task at the same time. [Parallel or Asynchronous programming]

Multi-tasking

MS-Word – Is a document based Application [ Used to create different types of documents]

MS-Word is a process – Application in running state.

Each running process will have a process id.

Small tasks which is running continuously in the background – is called as a thread.

MS-Word (document edition application) – Threads [Check Spelling Thread, Auto-saving]

Creating Threads in Java,

1. Created using Thread class
2. Created using Runnable Interface

A few important methods in the Thread class include:

* getters and setters for id, name, and priority
* interrupt() to explicitly interrupt the thread
* isAlive(), isInterrupted() and isDaemon() to test the state of the thread
* join() to wait for the thread to finish execution
* start() to actually begin thread execution after instantiation

Creating a Thread using Runnable interface reference is the recommended way of creating thread.

A class can extend only one another class but it can implement n no of interfaces.

States of Thread

1. **New**: newly created thread that has not started executing
2. **Runnable**: either running or ready for execution but waiting for its resource allocation
3. **Blocked**: waiting to acquire a monitor lock to enter or re-enter a synchronized block/method
4. **Waiting**: waiting for some other thread to perform an action without any time limit
5. **Timed\_Waiting**: waiting for some other thread to perform a specific action for a specified time period
6. **Terminated**: has completed its execution

If the class contains method which is having synchronized keyword for all methods, then the class is called as thread-safe class.