Creating Multiple Threads or Multithreading in Java

BY
M. BABY ANUSHA,
ASST.PROF IN CSE DEPT.,
RGUKT,NUZVID

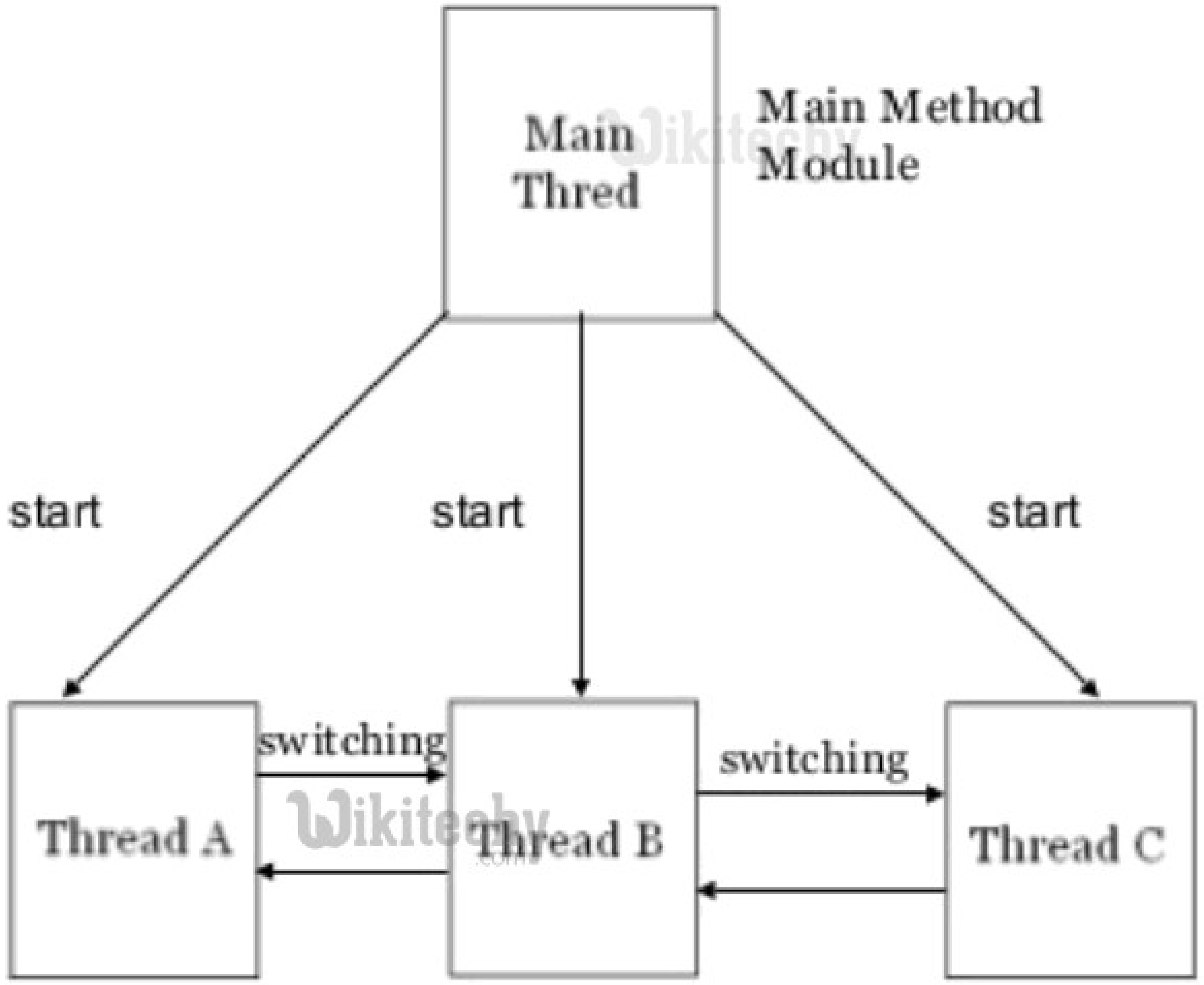
Multithreading :

- **Multithreading in** Java is a process of executing multiple threads simultaneously.
- A thread is a lightweight sub-process, the smallest unit of processing.
- Multiprocessing and multithreading, both are used to achieve multitasking.

Multithreading :

- However, we use multithreading than multiprocessing because threads use a shared memory area.
- They don't allocate separate memory area so saves memory, and context-switching between the threads takes less time than process.
- Java Multithreading is mostly used in games, animation, etc.

MultiThreading in Java



Advantages of Java Multithreading

- 1) It **doesn't block the user** because threads are independent and you can perform multiple operations at the same time.
- 2) You **can perform many operations together, so it saves time**.
- 3) Threads are **independent**, so it doesn't affect other threads if an exception occurs in a single thread.

Multitasking

- Multitasking is a process of executing multiple tasks simultaneously. We use multitasking to utilize the CPU. Multitasking can be achieved in two ways:
- Process-based Multitasking (Multiprocessing)
- Thread-based Multitasking (Multithreading)

Process-based Multitasking (Multiprocessing)

- Each process has an address in memory. In other words, each process allocates a separate memory area.
- A process is heavyweight.
- Cost of communication between the process is high.
- Switching from one process to another requires some time for saving and loading registers, memory maps, updating lists, etc.

Thread-based Multitasking (Multithreading)

- Threads share the same address space.
- A thread is lightweight.
- Cost of communication between the thread is low.
- Note: At least one process is required for each thread.

Thread Scheduler in Java

- **Thread scheduler** in java is the part of the JVM that decides which thread should run.
- There is no guarantee that which runnable thread will be chosen to run by the thread scheduler.
- Only one thread at a time can run in a single process.
- The thread scheduler mainly uses preemptive or time slicing scheduling to schedule the threads.

Daemon Thread in Java

- **Daemon thread in java** is a service provider thread that provides services to the user thread. Its life depend on the mercy of user threads i.e. when all the user threads dies, JVM terminates this thread automatically.
- There are many java daemon threads running automatically e.g. gc, finalizer etc.
- You can see all the detail by typing the jconsole in the command prompt. The jconsole tool provides information about the loaded classes, memory usage, running threads etc.

Daemon Thread in Java

- Points to remember for Daemon Thread in Java
- It provides services to user threads for background supporting tasks. It has no role in life than to serve user threads.
- Its life depends on user threads.
- It is a low priority thread.

Methods for Java Daemon thread

- Methods for Java Daemon thread by Thread class
- The java.lang.Thread class provides two methods for java daemon thread.

public void setDaemon(boolean status)

- is used to mark the current thread as daemon thread or user thread.

public boolean isDaemon()

- is used to check that current is daemon.

ThreadGroup in Java

- Java provides a convenient way to group multiple threads in a single object. In such way, we can suspend, resume or interrupt group of threads by a single method call.
- Java thread group is implemented by *java.lang.ThreadGroup* class.
- A ThreadGroup represents a set of threads. A thread group can also include the other thread group.

ThreadGroup in Java

- The thread group creates a tree in which every thread group except the initial thread group has a parent.
- A thread is allowed to access information about its own thread group, but it cannot access the information about its thread group's parent thread group or any other thread groups.

Constructors of ThreadGroup class

- There are only two constructors of ThreadGroup class.

ThreadGroup(String name)

- creates a thread group with given name.

ThreadGroup(ThreadGroup parent, String name)

creates a thread group with given parent group and name.

Thank you!
shutterstock.com · 567687052