Scala Multithreading

Multithreading is a process of executing multiple threads simultaneously. It allows you to perform multiple operations independently.

You can achieved multitasking by using Multithreading. Threads are lightweight sub-processes which occupy less memory. Multithreading are used to develop concurrent applications in Scala.

Scala does not provide any separate library for creating thread. If you are familiar with multithreading concept of Java, you will come to know that it is similar except the syntax of Scala language itself.

You can create thread either by extending Thread class or Runnable interface. Both provide a run method to provide specific implementation.

Scala Thread Life Cycle

Thread life cycle is a span of time in which thread starts and terminates. It has various phases like new, runnable, terminate, block etc. Thread class provides various methods to monitor thread's states.

The Scala thread states are as follows:

New

Runnable

Non-Runnable (Blocked)

Running

Terminated

Scala Multithreading 1

1) New

This is the first state of thread. It is just before starting of new thread.

2) Runnable

This is the state when thread has been started but the thread scheduler has not selected it to be the running thread.

3) Running

The thread is in running state if the thread scheduler has selected it.

4) Non-Runnable (Blocked)

This is the state when the thread is still alive, but is currently not eligible to run due to waiting for input or resources.

5) Terminated

A thread is in terminated or dead state when its run() method exits.