**Thread**

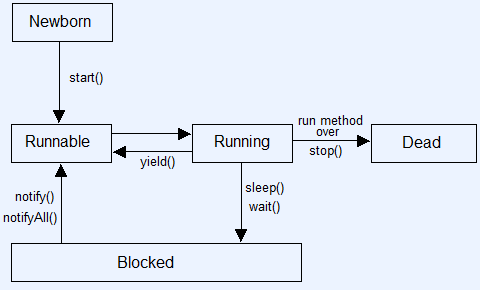
Thread can be called *lightweight process*. Thread requires less resources to create and exists in the process, thread shares the process resources.

### Process

A process is a self contained execution environment and it can be seen as a program or application. However a program itself contains multiple processes inside it. Java runtime environment runs as a single process which contains different classes and programs as processes.

Threads exist in several states. Following are those states:

* **New** – When we create an instance of Thread class, a thread is in a new state.
* **Running –**TheJava thread is in running state.
* **Suspended** – A running thread can be **suspended**, which temporarily suspends its activity. A suspended thread can then be resumed, allowing it to pick up where it left off.
* **Blocked** – A java thread can be blocked when waiting for a resource.
* **Terminated** – A thread can be terminated, which halts its execution immediately at any given time. Once a thread is terminated, it cannot be resumed.



**From JDK8- remove Runnable Block**

**From JDK 13 – remove stop().**

**New Born State:--**

* The thread enters the new born state as soon as it is created. The thread is created using the new operator.
* From the new born state the thread can go to ready to run mode or dead state.
* If start( ) method is called then the thread goes to ready to run mode. If the stop( ) method is called then the thread goes to dead state.

**Ready to run mode (Runnable Mode):--**

* If the thread is ready for execution but waiting for the CPU the thread is said to be in ready to run mode.
* All the events that are waiting for the processor are queued up in the ready to run mode and are served in FIFO manner or priority scheduling.
* From this state the thread can go to running state if the processor is available using the scheduled( ) method.
* From the running mode the thread can again join the queue of runnable threads.
* The process of allotting time for the threads is called time slicing.

**Running State:--**

* If the thread is in execution then it is said to be in running state.
* The thread can finish its work and end normally.
* The thread can also be forced to give up the control when one of the following conditions arise

1. A thread can be suspended by suspend( ) method. A suspended thread can be revived by using the resume() method.

     2. A thread can be made to sleep for a particular time by using the sleep(milliseconds) method.  
         The sleeping method re-enters runnable state when the time elapses.

     3. A thread can be made to wait until a particular event occur using the wait() method, which can  
         be run again using the notify( ) method.

**Blocked State:--**

* A thread is said to be in blocked state if it prevented from entering into the runnable state and so the running state.
* The thread enters the blocked state when it is suspended, made to sleep or wait.
* A blocked thread can enter into runnable state at any time and can resume execution.

**Dead State:--**

* The running thread ends its life when it has completed executing the run() method which is called natural dead.
* The thread can also be killed at any stage by using the stop( ) method.

**Difference between wait() and sleep() methods in thread:**  
  
1) wait is called from **synchronized**context only while sleep can be called without synchronized  
    block.  
2) wait is called on object while sleep is called on thread.  
3) waiting thread can be awake by calling notify and notifyAll while sleeping thread can not be  
     awaken by calling notify method.  
4) wait is normally done on condition, Thread wait until a condition is true while sleep is just to put  
    your thread on sleep.  
5) wait release lock on object while waiting while sleep doesn't release lock while waiting.

6) wait have 3 method Wait(), wait(ms),wait(ms,ns)

Where as sleep has 2 method sleep(ms), sleep(ms,ns)

## Thread Priorities

Every Java thread has a priority that helps the operating system determine the order in which threads are scheduled.

Java thread priorities are in the range between

MIN\_PRIORITY (a constant of 1) ,(Daemon Thread)

MAX\_PRIORITY (a constant of 10). By default, every thread is given priority (Child Thread)

NORM\_PRIORITY (a constant of 5).(main Thread)

Java provides two ways to create a thread programmatically.

1. Implementing the **java.lang.Runnable** interface.
2. Extending the **java.lang.Thread** class.

### Runnable vs Thread

If your class provides more functionality rather than just running as Thread, you should implement Runnable interface to provide a way to run it as Thread. If your class only goal is to run as Thread, you can extend Thread class.

Implementing Runnable is preferred because java supports implementing multiple interfaces. If you extend Thread class, you can’t extend any other classes.

|  |  |
| --- | --- |
| **Volatile Keyword** | **Synchronization Keyword** |
| Volatile keyword is a field modifier. | Synchronized keyword modifies code blocks and methods. |
| The thread cannot be blocked for waiting in case of volatile. | Threads can be blocked for waiting in case of synchronized. |
| It improves thread performance. | Synchronized methods degrade the thread performance. |
| It synchronizes the value of one variable at a time between thread memory and main memory. | It synchronizes the value of all variables between thread memory and main memory. |
| Volatile fields are not subject to compiler optimization. | Synchronize is subject to compiler optimization. |

**Constructor**

**Method of Runnable Interface**

**Run();**

**Constructor of Thread class**

|  |
| --- |
| [**Thread**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#Thread--)()  Allocates a new Thread object. |
| [**Thread**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#Thread-java.lang.Runnable-)(**[Runnable](file:///C:\\Users\\Gyan\\Desktop\\crrr\\doc\\java\\lang\\Runnable.html" \o "interface in java.lang)** target)  Allocates a new Thread object. |
| [**Thread**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#Thread-java.lang.Runnable-java.lang.String-)(**[Runnable](file:///C:\\Users\\Gyan\\Desktop\\crrr\\doc\\java\\lang\\Runnable.html" \o "interface in java.lang)** target, [**String**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\String.html) name)  Allocates a new Thread object. |
| [**Thread**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#Thread-java.lang.String-)([**String**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\String.html) name)  Allocates a new Thread object. |
| [**Thread**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#Thread-java.lang.ThreadGroup-java.lang.Runnable-)(**[ThreadGroup](file:///C:\\Users\\Gyan\\Desktop\\crrr\\doc\\java\\lang\\ThreadGroup.html" \o "class in java.lang)** group, **[Runnable](file:///C:\\Users\\Gyan\\Desktop\\crrr\\doc\\java\\lang\\Runnable.html" \o "interface in java.lang)** target)  Allocates a new Thread object. |
| [**Thread**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#Thread-java.lang.ThreadGroup-java.lang.Runnable-java.lang.String-)(**[ThreadGroup](file:///C:\\Users\\Gyan\\Desktop\\crrr\\doc\\java\\lang\\ThreadGroup.html" \o "class in java.lang)** group, **[Runnable](file:///C:\\Users\\Gyan\\Desktop\\crrr\\doc\\java\\lang\\Runnable.html" \o "interface in java.lang)** target, [**String**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\String.html) name)  Allocates a new Thread object so that it has target as its run object, has the specified name as its name, and belongs to  the thread group referred to by group. |
| [**Thread**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#Thread-java.lang.ThreadGroup-java.lang.Runnable-java.lang.String-long-)(**[ThreadGroup](file:///C:\\Users\\Gyan\\Desktop\\crrr\\doc\\java\\lang\\ThreadGroup.html" \o "class in java.lang)** group, **[Runnable](file:///C:\\Users\\Gyan\\Desktop\\crrr\\doc\\java\\lang\\Runnable.html" \o "interface in java.lang)** target, [**String**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\String.html) name, long stackSize)  Allocates a new Thread object so that it has target as its run object, has the specified name as its name, and belongs to the  thread group referred to by group, and has the specified *stack size*. |
| [**Thread**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#Thread-java.lang.ThreadGroup-java.lang.String-)(**[ThreadGroup](file:///C:\\Users\\Gyan\\Desktop\\crrr\\doc\\java\\lang\\ThreadGroup.html" \o "class in java.lang)** group, [**String**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\String.html) name)  Allocates a new Thread object. |

**Method of Thread Classs**

|  |  |
| --- | --- |
| Static int | [**activeCount**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#activeCount--)()  Returns an estimate of the number of active threads in the current  thread's [**thread group**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\ThreadGroup.html) and its subgroups. |
| void | [**checkAccess**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#checkAccess--)()  Determines if the currently running thread has permission to  modify this thread. |
| protected [**Object**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Object.html) | [**clone**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#clone--)()  Throws CloneNotSupportedException as a Thread can not be  meaningfully cloned. |
| int | [**countStackFrames**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#countStackFrames--)()  **Deprecated.**  The definition of this call depends on [**suspend()**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#suspend--), which is  deprecated. Further, the results of this call were never well-defined. |
| static [**Thread**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html) | [**currentThread**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#currentThread--)()  Returns a reference to the currently executing thread object. |
| void | [**destroy**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#destroy--)()  **Deprecated.**  This method was originally designed to destroy this thread without  any cleanup. Any monitors it held would have remained locked. |
|  |  |
| static int | [**enumerate**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#enumerate-java.lang.Thread:A-)([**Thread**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html)[] tarray)  Copies into the specified array every active thread in the current thread's thread group and its subgroups. |
| static java.util.Map<**[Thread](file:///C:\\Users\\Gyan\\Desktop\\crrr\\doc\\java\\lang\\Thread.html" \o "class in java.lang)**,[**StackTraceElement**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\StackTraceElement.html)[]> | [**getAllStackTraces**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#getAllStackTraces--)()  Returns a map of stack traces for all live threads. |
| [**ClassLoader**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\ClassLoader.html) | [**getContextClassLoader**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#getContextClassLoader--)()  Returns the context ClassLoader for this Thread. |
| static **[Thread.UncaughtExceptionHandler](file:///C:\\Users\\Gyan\\Desktop\\crrr\\doc\\java\\lang\\Thread.UncaughtExceptionHandler.html" \o "interface in java.lang)** | [**getDefaultUncaughtExceptionHandler**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#getDefaultUncaughtExceptionHandler--)()  Returns the default handler invoked when a thread abruptly  terminates due to an uncaught exception. |
| long | [**getId**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#getId--)()  Returns the identifier of this Thread. |
| [**String**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\String.html) | [**getName**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#getName--)()  Returns this thread's name. |
| int | [**getPriority**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#getPriority--)()  Returns this thread's priority. |
| [**StackTraceElement**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\StackTraceElement.html)[] | [**getStackTrace**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#getStackTrace--)()  Returns an array of stack trace elements representing the  stack dump of this thread. |
| [**Thread.State**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.State.html) | [**getState**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#getState--)()  Returns the state of this thread. |
| [**ThreadGroup**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\ThreadGroup.html) | [**getThreadGroup**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#getThreadGroup--)()  Returns the thread group to which this thread belongs. |
| [**Thread.UncaughtExceptionHandler**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.UncaughtExceptionHandler.html) | [**getUncaughtExceptionHandler**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#getUncaughtExceptionHandler--)()  Returns the handler invoked when this thread abruptly terminates  due to an uncaught exception. |
| static boolean | [**holdsLock**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#holdsLock-java.lang.Object-)([**Object**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Object.html) obj)  Returns true if and only if the current thread holds the monitor  lock on the specified object. |
| void | [**interrupt**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#interrupt--)()  Interrupts this thread. |
| static boolean | [**interrupted**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#interrupted--)()  Tests whether the current thread has been interrupted. |
| boolean | [**isAlive**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#isAlive--)()  Tests if this thread is alive. |
| boolean | [**isDaemon**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#isDaemon--)()  Tests if this thread is a daemon thread. |
| boolean | [**isInterrupted**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#isInterrupted--)()  Tests whether this thread has been interrupted. |
| void | [**join**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#join--)()  Waits for this thread to die. |
| void | [**join**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#join-long-)(long millis)  Waits at most millis milliseconds for this thread to die. |
| void | [**join**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#join-long-int-)(long millis, int nanos)  Waits at most millis milliseconds plus nanos nanoseconds for this thread to die. |
| void | [**resume**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#resume--)()  **Deprecated.**  This method exists solely for use with [**suspend()**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#suspend--),  which has been deprecated because it is deadlock-prone. |
| void | [**run**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#run--)()  If this thread was constructed using a separate Runnable run  object, then thatRunnable object's run method is called;  otherwise, this method does nothing and returns. |
| void | [**setContextClassLoader**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#setContextClassLoader-java.lang.ClassLoader-)(**[ClassLoader](file:///C:\\Users\\Gyan\\Desktop\\crrr\\doc\\java\\lang\\ClassLoader.html" \o "class in java.lang)** cl)  Sets the context ClassLoader for this Thread. |
| void | [**setDaemon**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#setDaemon-boolean-)(boolean on)  Marks this thread as either a [**daemon**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#isDaemon--) thread or a user thread. |
|  |  |
| void | [**setName**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#setName-java.lang.String-)([**String**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\String.html) name)  Changes the name of this thread to be equal to the argument name. |
| void | [**setPriority**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#setPriority-int-)(int newPriority)  Changes the priority of this thread. |
|  |  |
| static void | [**sleep**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#sleep-long-)(long millis)  Causes the currently executing thread to sleep (temporarily cease  execution) for the specified number of milliseconds, subject to the  precision and accuracy of system timers and schedulers. |
| static void | [**sleep**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#sleep-long-int-)(long millis, int nanos)  Causes the currently executing thread to sleep (temporarily cease  execution) for the specified number of milliseconds plus the  specified number of nanoseconds, subject to the precision and  accuracy of system timers and schedulers. |
| void | [**start**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#start--)()  Causes this thread to begin execution; the Java Virtual Machine  calls the run method of this thread. |
| void | [**stop**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#stop--)()  **Deprecated.**  This method is inherently unsafe. Stopping a thread with Thread.  stop causes it to unlock all of the monitors that it has locked (as a natural consequence of the unchecked ThreadDeath exception propagating up the stack). |
|  |  |
|  |  |
|  |  |
| static void | [**yield**](file:///C:\Users\Gyan\Desktop\crrr\doc\java\lang\Thread.html#yield--)()  A hint to the scheduler that the current thread is willing to yield its current use of a processor. |