Notes - Thread States

Thread States -

A thread can be in one of the following states:

NEW

A thread that is created but not yet started is in this state.

RUNNABLE

A thread executing in the Java virtual machine is in this state, internally we can think of it as a combination of two sub states *Ready* and *Running*, i.e. when you start the thread it comes to *Ready* state and wait for the CPU, and if CPU is allocated then it goes into *Running* state. Once allocated CPU time is completed, in other words when the Thread schedular suspends the thread then it goes back to the *Ready* state and waits for its turn again.

The yield() method instructs the thread schedular to pass the CPU to other waiting thread if any.

BLOCKED

A thread is blocked if it is waiting for a monitor lock is in this state. Refer **synchronized** methods and blocks.

WAITING

A thread that is waiting indefinitely for another thread

to perform a particular action is in this state.
Refer `wait()`, `join()`

TIMED_WAITING

A thread that is waiting for another thread to perform an action for up to a specified waiting time is in this state. Refer `wait(millis)`, `join(millis)`

TERMINATED

A thread that has exited i.e. it has either completed its task or terminated in the middle of the execution.

yield() method -

yield() method is important in few scenarios, suppose a thread is given 5 min of CPU time, now after a minute thread knows that it doesn't need the CPU anymore with in that time period, in such scenarios do you think that blocking the CPU for the next four minutes is a good idea ? No, it is better to pass on the control to the threads if any waiting for CPU and that is when we can use the yield() method. Usage Thread.yield(), it is a static method of the Thread class and it affects the current thread from which the method is invoked.