**Difference Between Sleep and Yeild**

|  |  |
| --- | --- |
| **Sleep** | **Yeild** |
| If a thread doesn’t want to perform any operation for a particular amount of time then we should go for the sleep method  The sleep method is overloaded.  This method throws  Interrupted Exception  This method doesn’t cause the currently executing thread to give up monitors. | If a thread wants to pause its execution to give chance for the remaining thread of the same priority then we should go for the yield method.  This method is not overloaded  This method doesn’t throw an exception  This method gives up the monitors. |

**Deamon Thread**

* Daemon thread in Java is a low-priority thread that runs in the background to perform tasks such as garbage collection. Daemon thread in Java is also a service provider thread that provides services to the user thread. Its life depends on the mercy of user threads i.e. when all the user threads die, JVM terminates this thread automatically.

**Dead Lock**

*Deadlock* describes a situation where two or more threads are blocked forever, waiting for each other.

**When Do You Interrupt a Thread And How To Do it**

* An interrupt is an indication to a thread that it should stop what it is doing and do something else.
* It's up to the programmer to decide exactly how a thread responds to an interrupt, but it is very common for the thread to terminate.

Using interrupted() method which is static

**Why Threads Are Light Weighted And Heavy Weighted**

* A process is said to be heavy weight because **OS has to do lot of housekeeping to create a process** . It has to allocate lot of buffers, make a note of entry in different files and notify schedulers , memory management etc about this new entry. Hence creating a new process is slow and it is heavy weight.
* Threads are sometimes called lightweight processes **because they have their own stack but can access shared data**. Because threads share the same address space as the process and other threads within the process,

Diffrerence Between notify and notifyAll

|  |  |
| --- | --- |
| Notify() | notifyAll() |
| In the case of the multiThreading, notify() method sends the notification to only one thread among the multiple waiting threads which are waiting for the send lock.  it is sure that which of those waiting threads is going to receive the lock.  the risk of thread missing is high as notification is sent only a single thread, and if it misses that, then no other thread would get a notification  no concept of thread Interchangeable | While notifyAll() methods in the same context send notifications to all waiting threads instead of a single thread.  it is not clear which of the thread is going to receive the lock.  it sends a notification to all the waiting threads, Hence the risk is less.   if all your waiting threads are interchangeable (the order they wake up doesn’t matter).- |