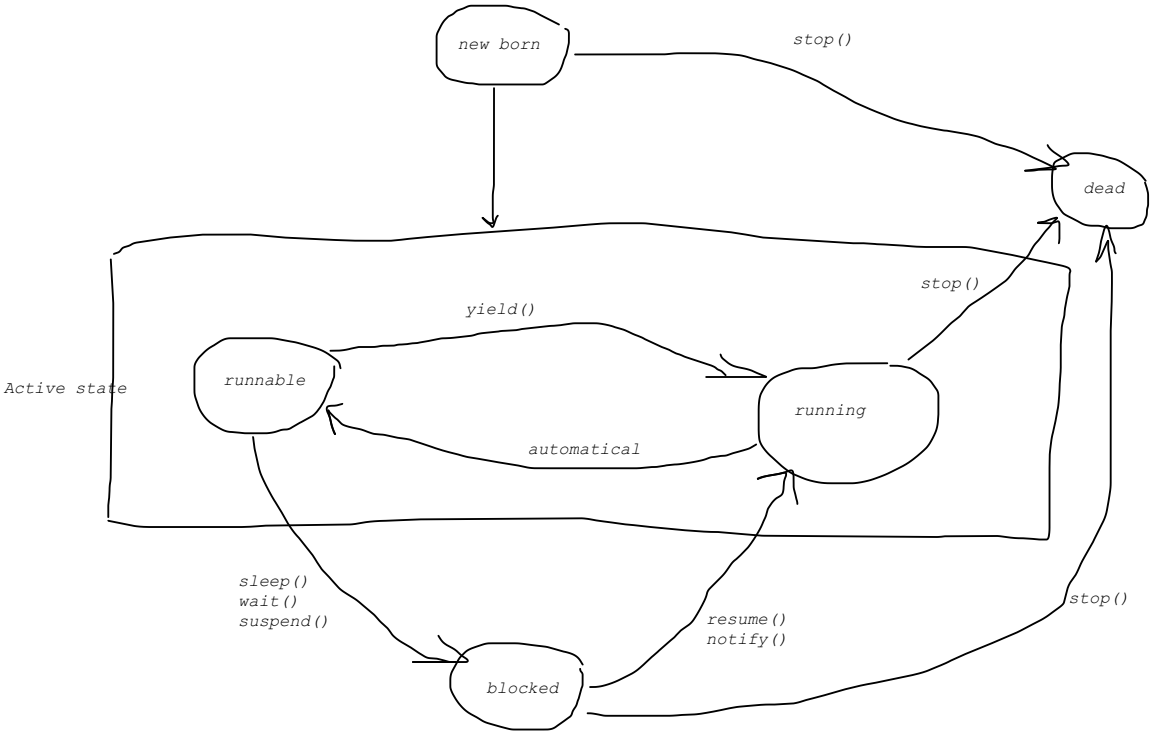
Life cycle of a Thread

During the life cycle of a thread, there are many states that it can enter.

1. new born state
2. runnable state
3. running state
4. blocked state
5. dead state

following is the state transition diagram of a thread



controlling the execution of a thread

three operations can be used to block a thread

suspend(); -block until further order, resume() will take the thread back to running state

sleep(<time in milliseconds>); -block the thread for a specific time

wait(); - block until certain condition is satisfied

In the case if sleep() the thread will return to the runnable state when the specified time is passed. The resume() method is used to make a thread runnable that is blocked by suspend() method, and notify() is called in the case of wait().

using thread methods example:

```
class FirstThread extends Thread{
	public void run(){
		for (int i=0; i<=5; i++){
			if (i==3){
				yield();
				System.out.println("From thread A i ="+i);
				}
			System.out.println("Exit thread A");
			}
		}
	}

class SecondThread extends Thread{
	public void run(){
		for (int i=0; i<=5; i++){
			System.out.println("From thread B i ="+i);
			}
		System.out.println("Exit thread B");
		}
	}


class ThirdThread extends Thread{
	public void run(){
		for (int i=0; i<=5; i++){
			if (i==1){
				try{
					sleep(1000);
					}
				catch(Exception e){
					System.out.println(e);
					}
				}
			}
			System.out.println("Exit thread C");
		}
	}


public class Test{
	public static void main(String args[]){
		FirstThread f = new FirstThread();
		SecondThread s = new SecondThread();
		ThirdThread t= new ThirdThread();

		System.out.println("Starting threads");
		t.start();
		s.start();
		f.start();
		//System.out.println("Exit from main");
		}
	}
```

Run the above code and understand the behavior