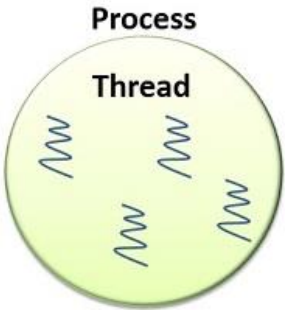
Types of Thread

User thread **Daemon Thread**

A thread is used to perform parallel execution in Java e.g. while rendering screen your program is also downloading the data from the internet in the background. There are two types of threads in Java, **user thread and daemon thread,** both of which can use to implement parallel processing in Java depending upon priority and importance of the task. **The main difference between a user thread and a daemon thread is that your Java program will not finish execution until one of the user thread is live.** JVM will wait for all active user threads to finish their execution before it shutdown itself. On the other hand, **a daemon thread doesn't get that preference, JVM will exit and close the Java program even if there is a daemon thread running in the background.** They are treated as **low priority threads** in Java, that's why they are more suitable for non-critical background jobs.

Let's see with an example:

```
class Demo1 extends Thread
{
    public void run()
    {
        for(int i=1;i<=5;i++)
        {
            System.out.println("User thread executing");
            try
            {
                Thread.sleep(1000);
            }
            catch (Exception e)
            {
                System.out.println("Some problem occured");
            }
        }
    }
}
```



```java
class Demo2 extends Thread
{
    public void run()
    {
        for(;;)
        {
            System.out.println("Daemon thread executing");
            try
            {
                Thread.sleep(1000);
            }
            catch (Exception e)
            {
                System.out.println("Some problem occured");
            }
        }
    }
}
class Demo
{
    public static void main(String[] args)
    {
        System.out.println("Main() started execution");
        Demo1 d1 = new Demo1();
        Demo2 d2 = new Demo2();
        d2.setDaemon(true);
        d1.start();
        d2.start();
        System.out.println("Main() completed execution");
    }
}
```

Output:

```
Main() started execution
Main() completed execution
User thread executing
Daemon thread executing
Daemon thread executing
User thread executing
Daemon thread executing
User thread executing
Daemon thread executing
User thread executing
Daemon thread executing
User thread executing
Daemon thread executing
Press any key to continue . . .
```

In the above example we see that the daemon thread stopped executing right after user thread completed it's execution.

Difference between User thread and Daemon thread

- **JVM doesn't wait for daemon thread to finish**
  First and foremost difference is that JVM will not wait for daemon thread to finish their work but it will wait for any active user thread.
- **JVM itself creates Daemon thread**
  Another difference between them is that daemon thread is mostly created by JVM e.g. for some garbage collection job. On the other hand user thread is usually created by the application for executing some task concurrently.
- **Daemon thread is not used for critical task**
  Another key difference between daemon and user thread is that daemons are not used for any critical task. Any important task is done by non-daemon or user thread. A daemon thread is generally used for some background tasks which are not critical.
- **Thread Priority**
  As compared to user thread, the daemon threads are low priority thread. Which means they won't get CPU as easily as a user thread can get.
- **JVM closes daemon thread**
  The user thread is closed by application or by itself but JVM will force daemon thread to terminate if all user threads have finished their execution. A daemon thread cannot keep the JVM running but a user thread can. This is the most critical difference between daemon and user thread which you should remember.