Thread :

Thread is basic unit of execution which consists of its own thread ID, program counter, a stack, and a set of registers. A thread is also known as **lightweight process.** Threads are popular way to improve application through parallelism. Each thread belongs to exactly one process and no thread can exist outside a process. A thread shares with its peer threads few information like code segment, data segment and open files. The CPU switches rapidly back and forth among the threads giving illusion that the threads are running in parallel.

Types of Thread:

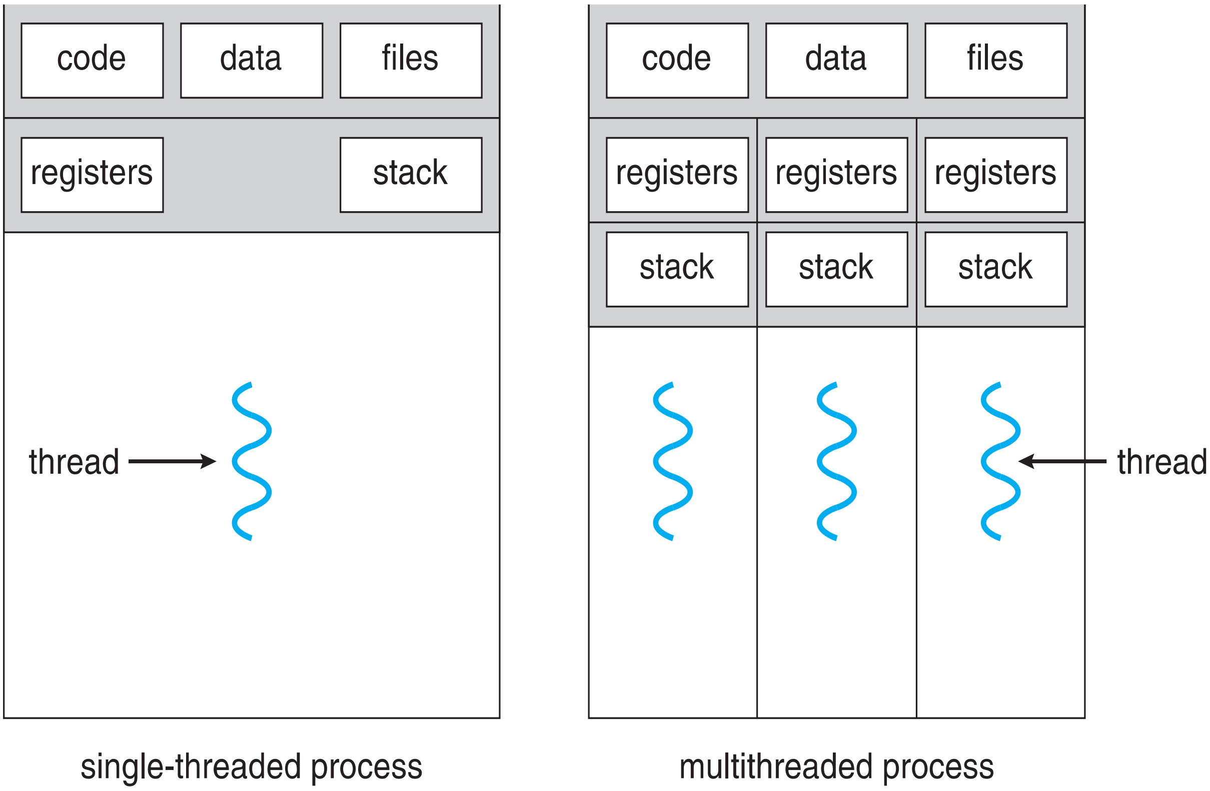
There are two types of threads:

1. User Threads
2. Kernel Threads

User threads, are above the kernel and without kernel support. These are the threads that application programmers use in their programs. The user-level threads are implemented by users and the kernel is not aware of the existence of these threads. It handles them as if they were single-threaded processes. The implementation of a user level thread is by the thread library at the user level. User-level threads are small and much faster than kernel level threads. Also, there is no kernel involvement in synchronization for user-level threads.

Kernel threads are supported within the kernel of the OS itself. All modern OSs support kernel level threads, allowing the kernel to perform multiple simultaneous tasks and/or to service multiple kernel system calls simultaneously. Kernel-level threads are handled by the operating system directly and the thread management is done by the kernel. The context information for the process as well as the process threads is all managed by the kernel. Because of this, kernel-level threads are slower than user-level threads.

Single and Multithreaded processes :



Single threaded processes contain the execution of instructions in a single sequence. So it can perform only one task at a time. But multithreaded processes. allow the execution of multiple parts of a program at the same time. If a process has multiple threads of control, it can perform more than one task at a time. In multithreaded process, the code section, data section and certain structures such as open files of the process are shared by the threads and the threads have their own program counter, stack and register.

Thus multithreaded processes are much more efficient compared to the single threaded processes.