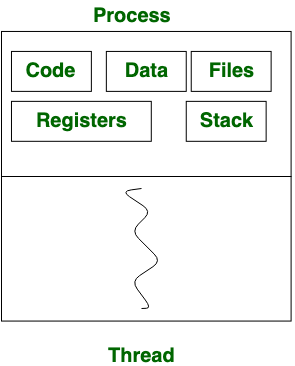
**Process-**  
Process are basically the programs which are dispatched from the ready state and are scheduled in the CPU for execution. PCB([Process Control Block](https://www.geeksforgeeks.org/process-table-and-process-control-block-pcb/)) holds the concept of process. A process can create other processes which are known as Child Processes. The process takes more time to terminate and it is isolated means it does not share the memory with any other process.

The process can have the following [states](https://www.geeksforgeeks.org/states-of-a-process-in-operating-systems/) like new, ready, running, waiting, terminated, suspended.

**Thread-**  
Thread is the segment of a process means a process can have multiple threads and these multiple threads are contained within a process. A thread has three states: Running, Ready, and Blocked.

Thread takes less time to terminate as compared to process but unlike process threads do not isolate.



*Process vs Thread*

**Difference between Process and Thread:**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **Process** | **Thread** |
| 1. | Process means any program is in execution. | Thread means segment of a process. |
| 2. | Process takes more time to terminate. | Thread takes less time to terminate. |
| 3. | It takes more time for creation. | It takes less time for creation. |
| 4. | It also takes more time for context switching. | It takes less time for context switching. |
| 5. | Process is less efficient in term of communication. | Thread is more efficient in term of communication. |
| 6. | Multi programming holds the concepts of multi process. | We don’t need multi programs in action for multiple threads because a single process consists of multiple threads. |
| 7. | Process is isolated. | Threads share memory. |
| 8. | Process is called heavy weight process. | A Thread is lightweight as each thread in a process shares code, data and resources. |
| 9. | Process switching uses interface in operating system. | Thread switching does not require to call a operating system and cause an interrupt to the kernel. |
| 10. | If one process is blocked then it will not effect the execution of other process | Second thread in the same task could not run, while one server thread is blocked. |
| 11. | Process has its own Process Control Block, Stack and Address Space. | Thread has Parents’ PCB, its own Thread Control Block and Stack and common Address space. |
| 12. | If one process is blocked, then no other process can execute until the first process is unblocked. | While one thread is blocked and waiting, a second thread in the same task can run. |
| 13. | Changes to the parent process does not affect child processes. | Since all threads of the same process share address space and other resources so any changes to the main thread may affect the behavior of the other threads of the process. |