Thread

- A single sequence stream within a process
- Possesses same properties as of processes, also called light weight processes.

Contents of thread

- A program counter
- A register set
- A stack space

Properties

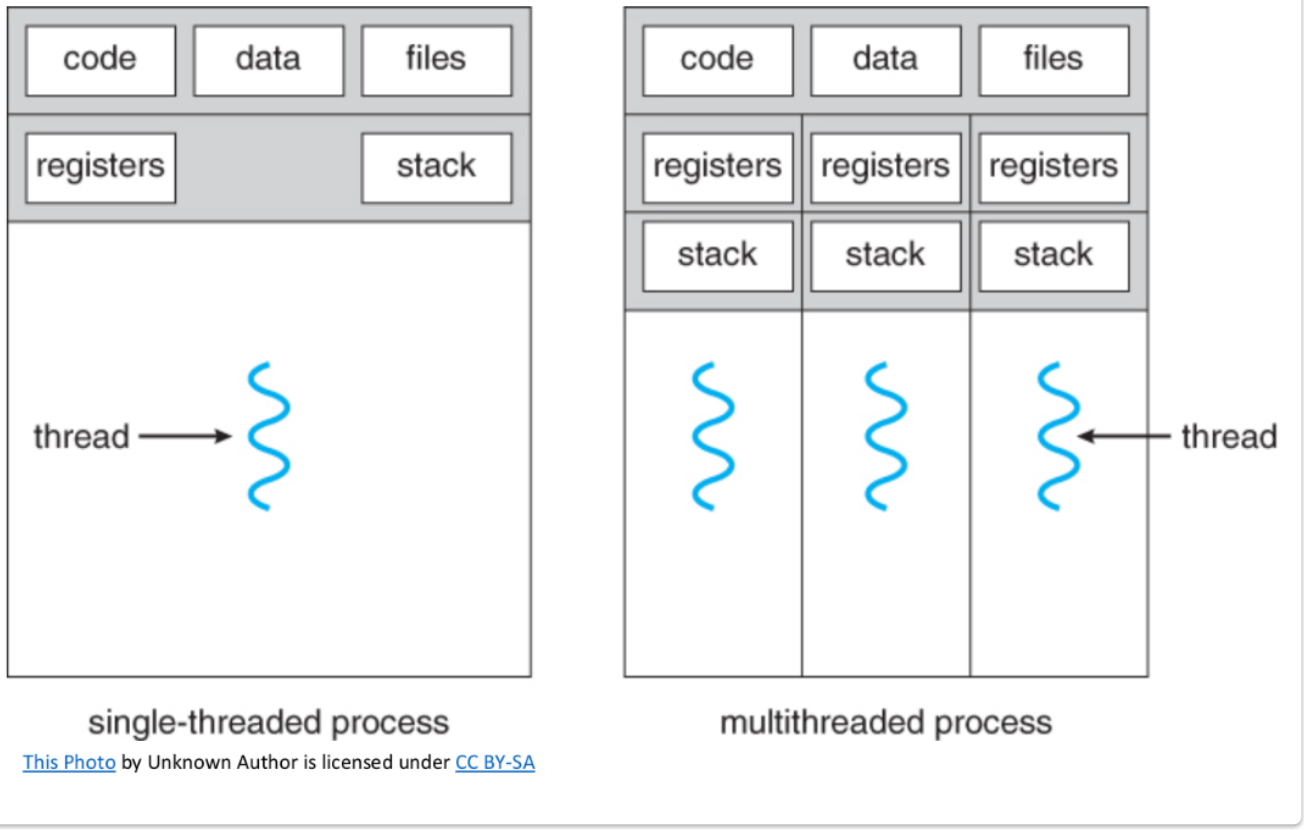
- Not independent like processes.
- Designed to assist other threads.
- They share with other threads:
  - Their code section
  - Data section
  - OS resources

Similarities

- Only one thread or process is active at a time
- Within process both execute sequentially.
- Both can create children

Advantages of threads over processes

- Responsiveness
- Faster context switch
- Effective utilization of multiprocessor system
- Resource sharing
- Communication
- Enhanced throughput



Multithreading

Can be implemented in 2 ways:

- User level threads - uses thread table
- Kernel level threads - user thread table as well as the process table

```
#include <pthread.h>
int pthread_create(pthread_t *tidp, pthread_attr_t *attr, void *(*start_rtn), void
*arg);
```

Returns 0 if OK, error number on failure
`tidp` is set to the thread ID of the newly created thread.
`attr` used to customise various thread attributes
Starts running at the address of `start_rtn` function (`run()` function) with a single argument `arg`

- No guarantee which thread will execute first.

```
tid = pthread_self();
```

Thread termination

```c
#include<pthread.h>
void pthread_exit(void *rval_ptr);
```

Joining threads

```c
#include <pthread.h>
int pthread_join(pthread_t thread, void *rval_ptr);
```

Returns 0 if OK, err number on failure
The calling thread will block until the specified thread calls `pthread_exit()`.