Execution Units

Execution Units

1. Process
2. Thread

Process often means application

Task Manager

File Options View

Processes | Performance | App history | Startup | Users | Details | Services

Name	PID	Status	CPU	CPU time	Memory (p...	Description
AcroRd32.exe	3500	Running	00	0:00:11	2,492 K	Adobe Reader
AcroRd32.exe	3536	Running	00	0:14:13	9,972 K	Adobe Reader
AdobeCollabSync.exe	1352	Running	00	0:00:00	1,308 K	Adobe Collaboration Synchronizer 11.0
armsvc.exe	1580	Running	00	0:00:00	104 K	Adobe Acrobat Update Service
atieclxx.exe	1064	Running	00	0:00:01	900 K	AMD External Events Client Module
atiesrxx.exe	844	Running	00	0:00:00	304 K	AMD External Events Service Module
audiodg.exe	6892	Running	00	0:00:00	3,188 K	Windows Audio Device Graph Isolation
chrome.exe	7196	Running	00	0:00:31	42,760 K	Google Chrome
chrome.exe	2216	Running	00	0:00:29	23,412 K	Google Chrome
chrome.exe	6648	Running	00	0:00:00	8,108 K	Google Chrome
chrome.exe	776	Running	00	0:00:04	25,888 K	Google Chrome
chrome.exe	5036	Running	00	0:00:01	13,988 K	Google Chrome
chrome.exe	5816	Running	00	0:00:00	9,288 K	Google Chrome
chrome.exe	5768	Running	00	0:00:21	78,908 K	Google Chrome
chrome.exe	5828	Running	00	0:00:20	28,332 K	Google Chrome
chrome.exe	7012	Running	00	0:00:02	20,364 K	Google Chrome
chrome.exe	4984	Running	00	0:00:31	23,800 K	Google Chrome
csrss.exe	396	Running	00	0:00:07	1,244 K	Client Server Runtime Process
csrss.exe	512	Running	00	0:06:25	844 K	Client Server Runtime Process
dasHost.exe	1684	Running	00	0:00:00	152 K	Device Association Framework Provider Host
Dropbox.exe	3772	Running	00	0:00:32	15,572 K	Dropbox
dwm.exe	816	Running	00	0:22:40	15,672 K	Desktop Window Manager
explorer.exe	2688	Running	00	0:03:25	25,976 K	Windows Explorer
firefox.exe	4552	Running	00	0:00:07	191,360 K	Firefox
flux.exe	3592	Running	00	0:00:21	2,440 K	f.lux

Process Name	% CPU	CPU Time	Threads	Idle Wake Ups	PID
com.apple.wifi.proxy	0.0	0.24	2	0	237
CoreServicesUIAgent	0.0	0.65	4	0	286
CVMCompiler	0.0	0.23	2	0	12606
Dash	0.0	10.41	15	0	283
DataDetectorsDynamicData	0.0	0.25	2	0	827
dbfseventsd	0.0	3.31	1	0	6208
Deckset	0.0	47.74	22	0	12357
diagnostics_agent	0.0	1.38	2	0	259
distnoted	0.0	17.57	5	0	185
Dock	0.0	23.86	3	0	189
Dropbox	0.1	41.32	43	0	6185
EscrowSecurityAlert	0.0	0.29	3	0	2551
Finder	0.0	43.47	15	0	193
Firefox	6.9	3.25	48	3	12671
Flux	0.1	41.37	5	2	264
fmfd	0.0	0.38	2	0	219
fontd	0.0	12.84	3	0	205
Google Chrome	0.0	20.78	38	0	12342
Google Chrome Helper	0.0	0.99	10	1	12599
Google Chrome Helper	0.0	0.50	7	0	12350
Google Chrome Helper	0.0	1.24	11	1	12595
Google Chrome Helper	0.0	1.81	9	1	12564
Google Chrome Helper	0.0	2.52	9	1	12579

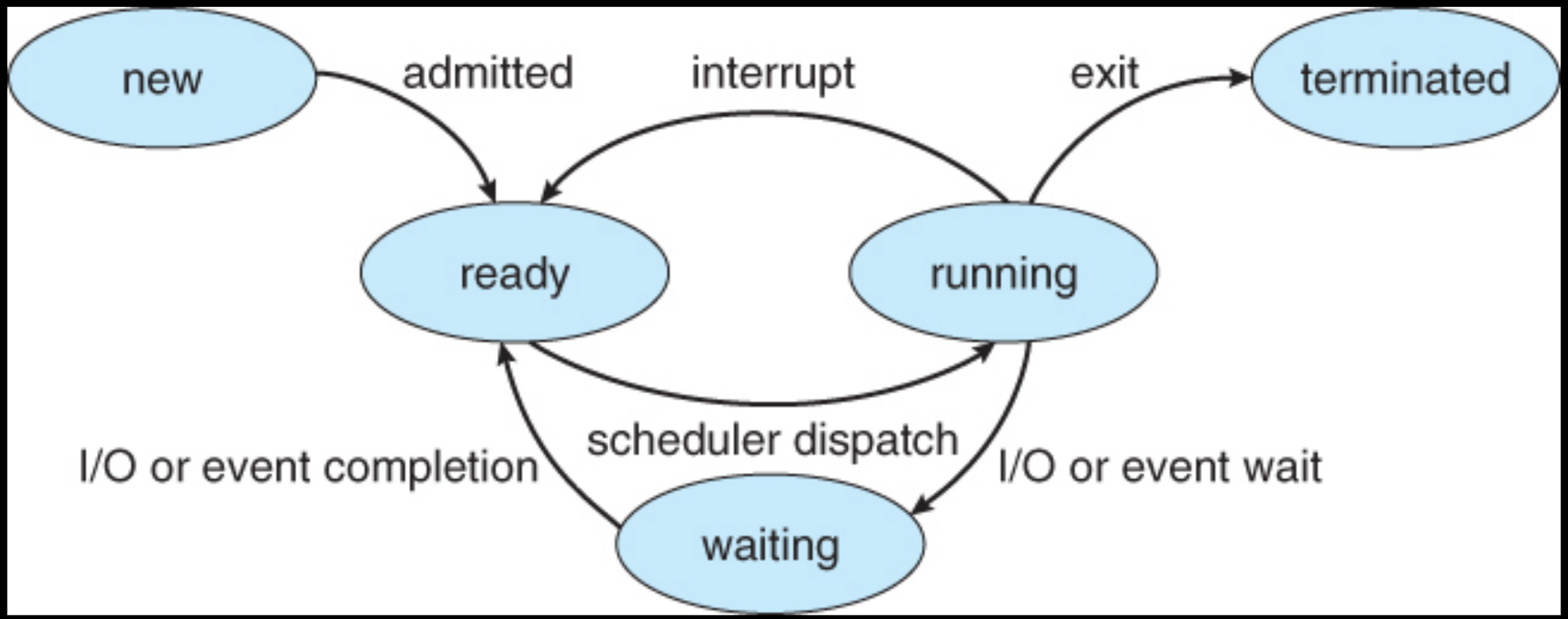
Most of applications are single process

Multiple processes application: Chrome

Process

Self-contained excution environment
- Virtual address space
- Executable code
- Privileges
- Resources (files, sockets)

Process States



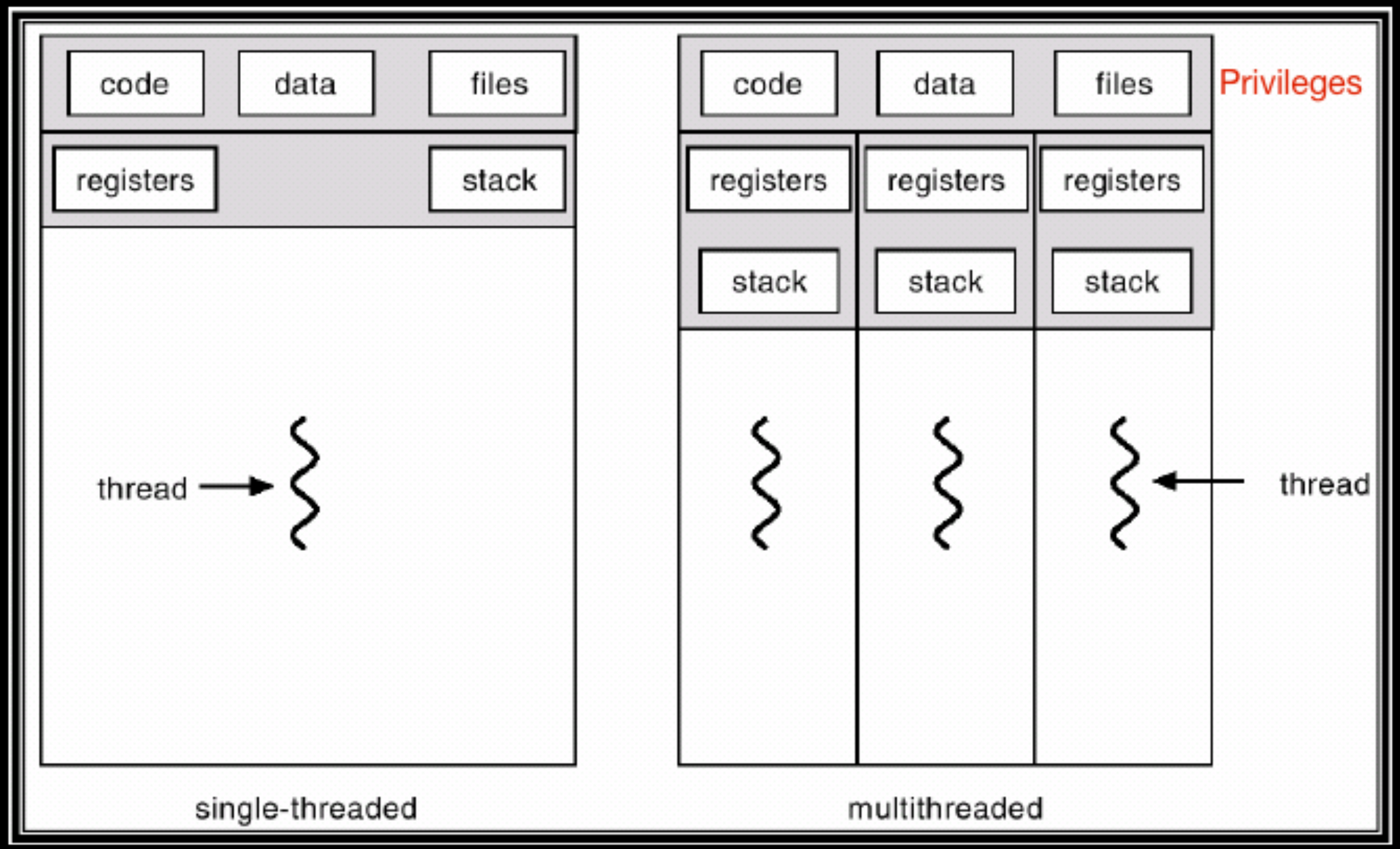
Create Process

- Path to application
- Application arguments
- Environment variables
- Privileges (access rights)

Inter Process Communication

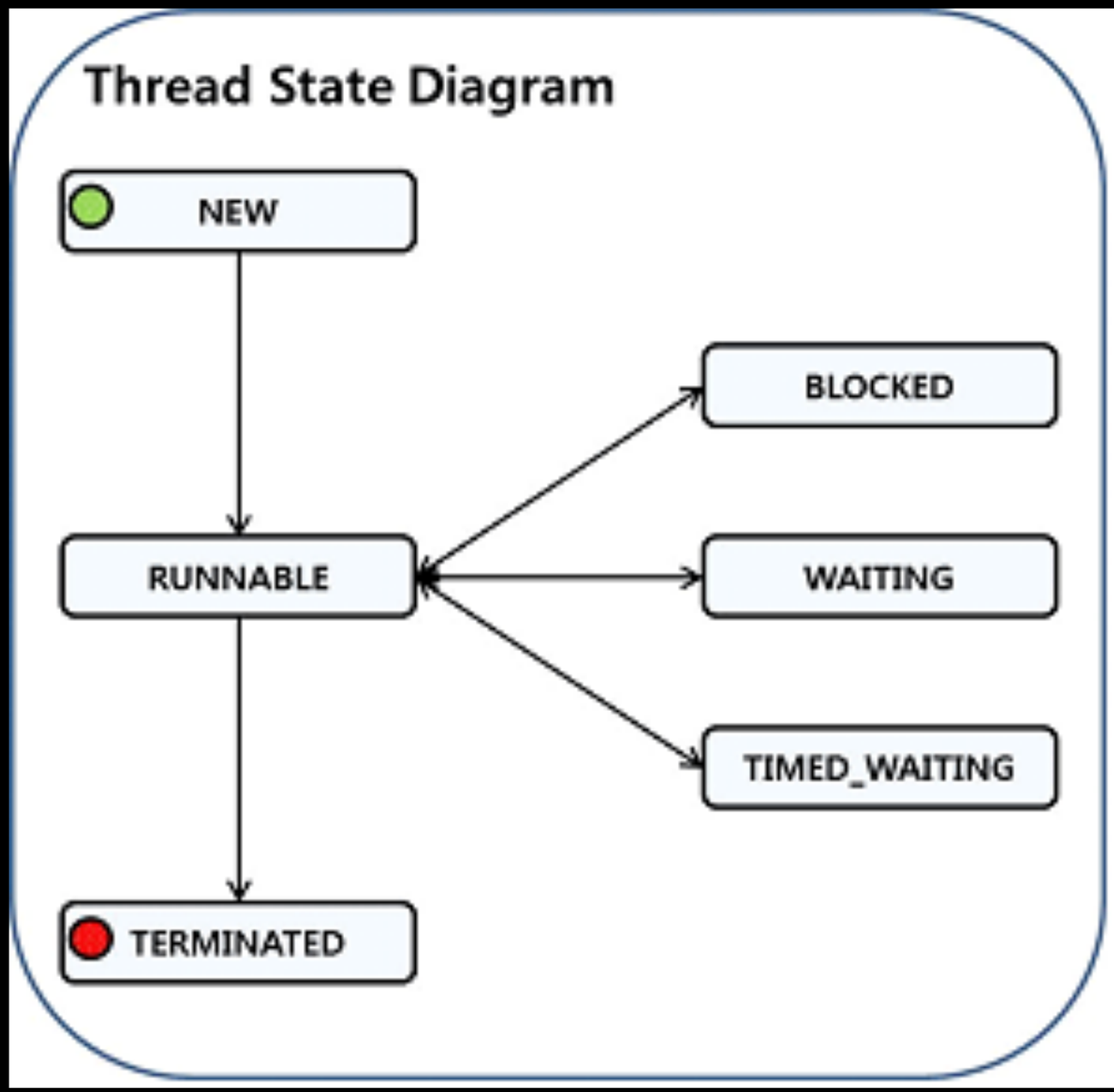
- Socket
- Shared Memory
- Pipe / Unix Socket

Multithreaded Process



Thread State

WAITING is TIMED_WAITING with infinite timeout (0 means INFINITE)



Steps to create threads

1. Extends `Thread class` and override `run method`
2. Or implements `Runnable Interface` and implements `run method`
3. Create new execution as a regular object
4. Call `start` to start a thread

Extends Thread class

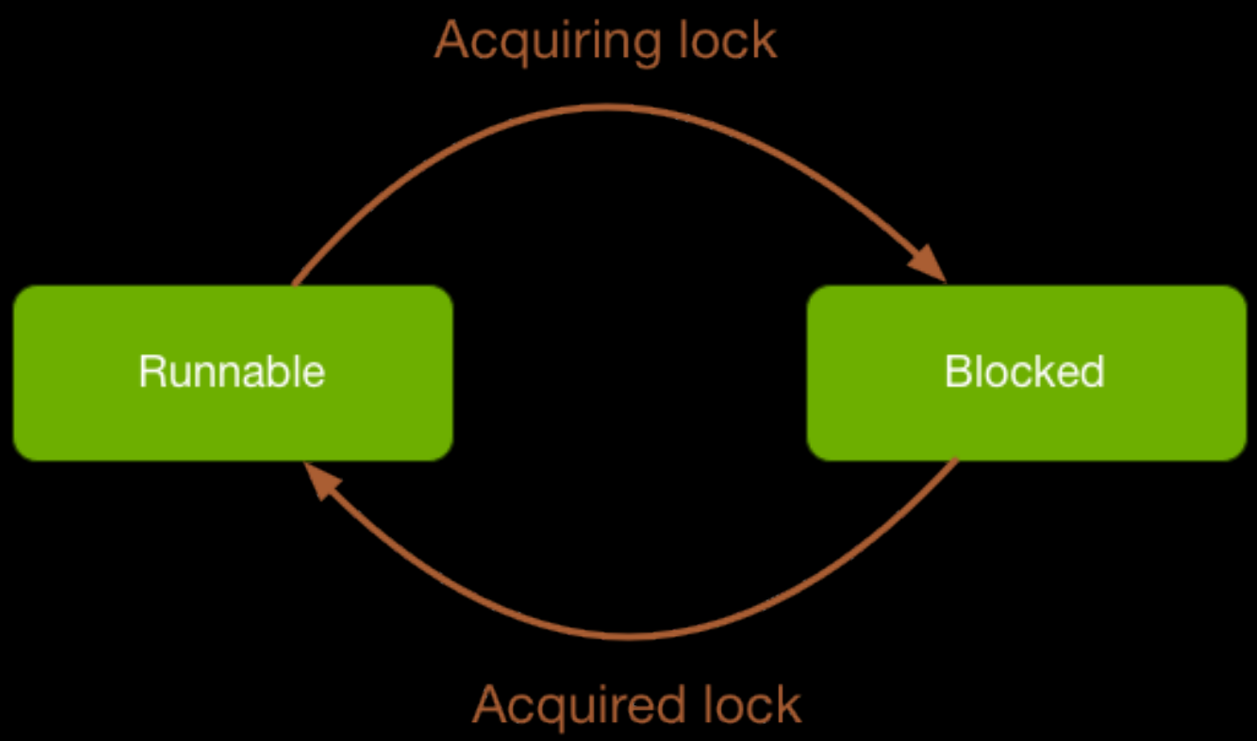
```
class PrimeCalculation extends Thread {
    long minPrime;

    PrimeThread(long minPrime) {
        this.minPrime = minPrime;
    }

    public void run() {
    // compute primes larger than minPrime
    }
}

PrimeCalculation p = new PrimeCalculation(143);
p.start();
```

Implements Runnable

```
class PrimeCalculation implements Runnable {
    long minPrime;

    PrimeThread(long minPrime) {
        this.minPrime = minPrime;
    }

    public void run() {
    // compute primes larger than minPrime
    }
}

PrimeCalculation p = new PrimeCalculation(143);
p.start();
```

Blocked State

Calling `synchronized` method/block is to acquire an intrinsic lock.



Timed Wait

- `sleep` until timeout
- `wait(timeout)` until other threads call `notify` or `notifyAll`
- `join` wait until that thread die

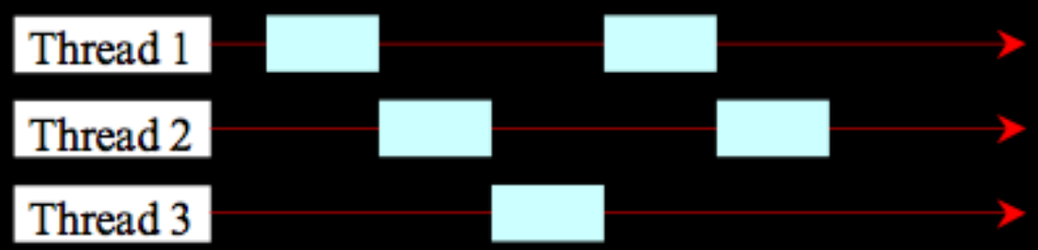
Important Methods

- `Thread.start` to start a thread
- `Object.wait` to wait until timeout or other thread wakes up
- `Object.notify` to wake up one of waiting threads
- `Object.notifyAll` to wake up all waiting threads
- `Thread.sleep` to sleep until timeout

CPU Cycles

How to schedule threads?

Multiple processer (Parallel)



Single processor (Concurrent)

Scheduling

Time-slicing: interrupts the running thread periodically to give other threads a chance to run.

Other schedulers consider thread priority

Context Switch

Context: CPU Registers + Program Counter

Yield

`Thread.yield` to temporarily pause executing thread and allow other threads to execute

Executor

- Manage thread is still an expensive operation.
- Thread pool or executor contains lightweight exection.

Executor Example

```
ExecutorService executor = Executors.newFixedThreadPool(poolSize);
class PrimeCalculation implements Runnable {
    long minPrime;

    PrimeThread(long minPrime) {
        this.minPrime = minPrime;
    }

    public void run() {
    // compute primes larger than minPrime
    }
}
excutor.execute(new PrimeCalculation(100));
```