Thread in Java

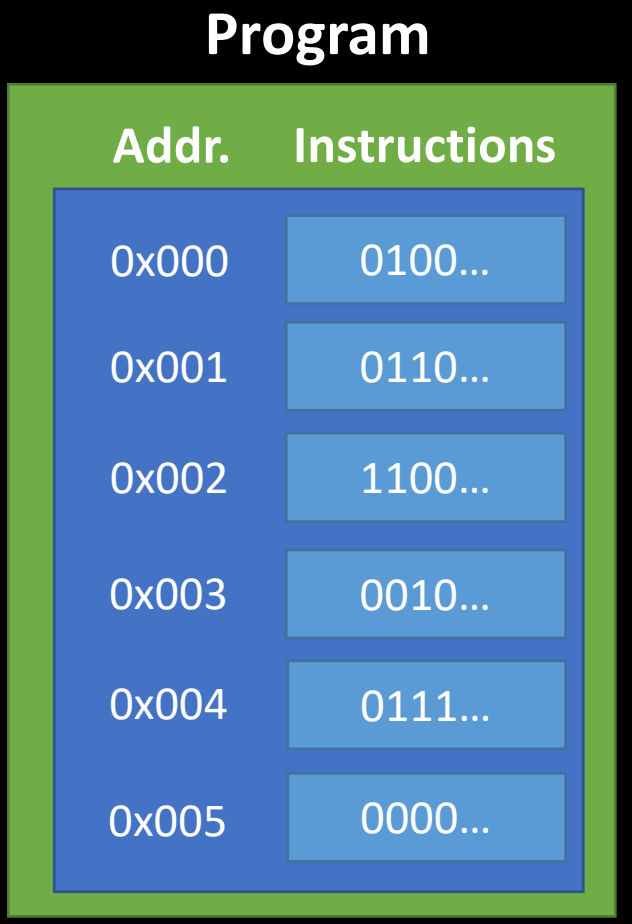
0. 목차

1. 의의

- Back-end
  - 멀티-스레드 환경에서의 개발이 필수
  - Apache Tomcat, Spring, …
  - 안정적인 서버 운영을 위해, 스레드에 대한 이해가 중요!
    - 서버 프로그램의 동작 예측
    - 발생 가능한 예외의 처리

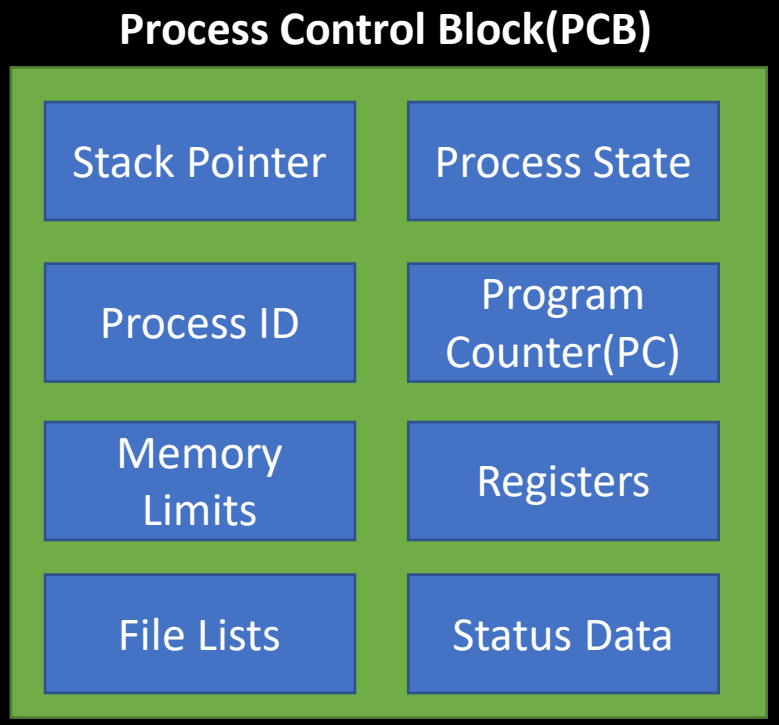
2. Program

- Program
  - 디스크(HDD, SSD)에 존재하는 명령어(Instruction)의 집합
  - 구성 요소: 코드(Code), 데이터(Data)



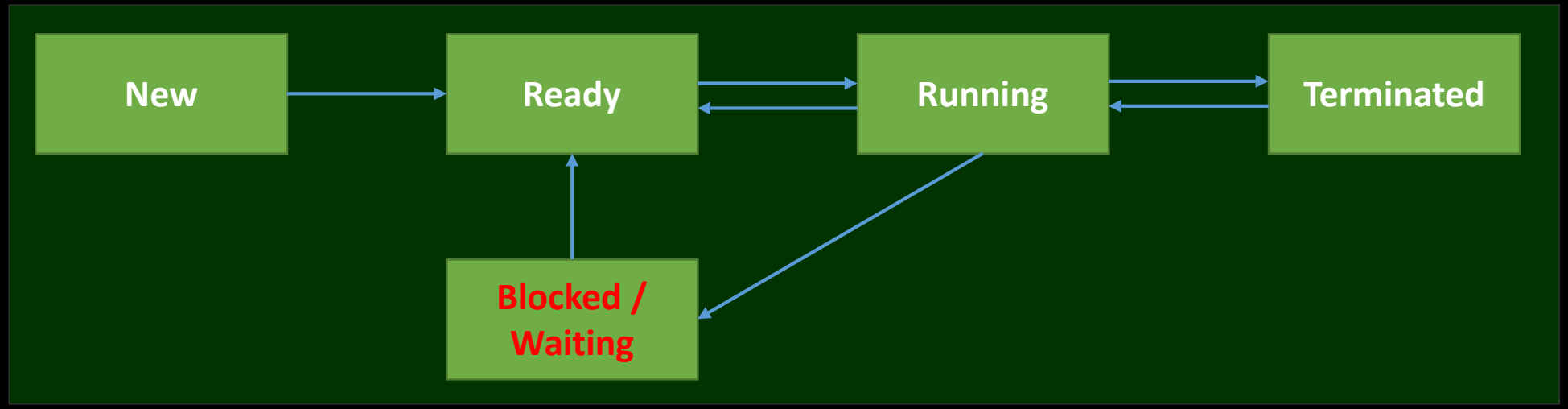
3. Process

- Process
  - CPU에 의해 실행(Execution)이 가능한 프로그램
  - 메인 메모리(RAM)에 주로 존재
  - 시스템의 자원(CPU, Memory 주소, I/O) 필요성 ↑
  - 코드(Code), 데이터(Data) + 현재 프로세스의 상태 정보(PCB)



4. Five State Process Model

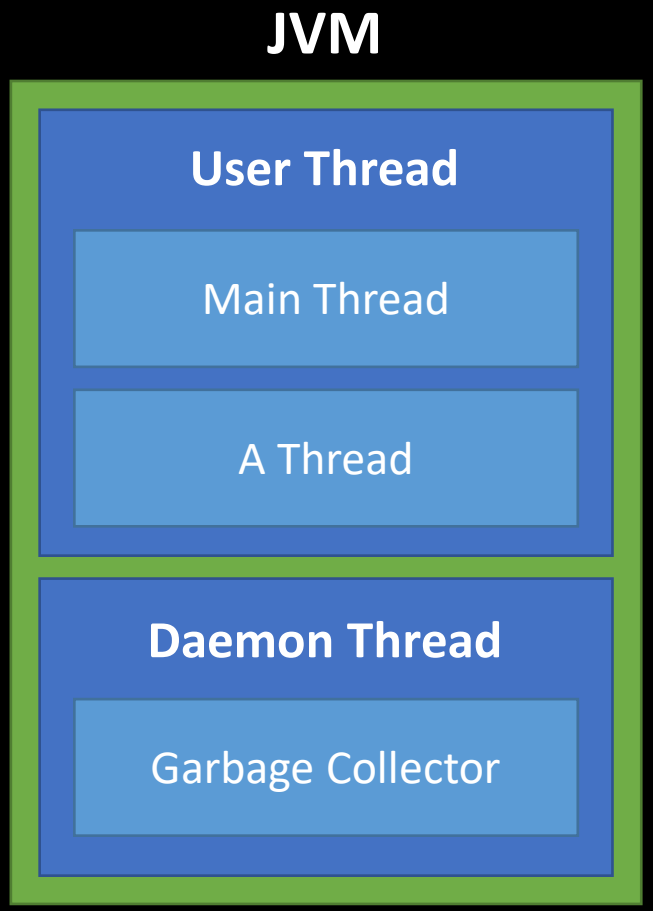
OS



- **Blocked / Waiting 상태**
  - 프로세스가 다음 명령을 실행할 수 없는 상태
    1. 접근이 불가능한 영역(Critical Section)에 접근하려 하는 경우
    2. I/O 동작을 수행해야 하는 경우
    3. …

5. Thread(1)

- 프로세스 내 존재하는, Execution(실행)의 분기
- 주로, OS의 스케줄링 정책에 따라 실행되는 스레드가 결정
  - JDK의 구현에 따라 상이할 수 있다.
- 스레드 간 공유되는 자원
  - 코드, 데이터, OS 자원
- 스레드의 독립적인 자원
  - Program Counter(PC), Register, Stack 공간



5. Thread(2)

- 장점
  1. 하나의 스레드에서, 다른 스레드로의 변경 비용이 가볍다
  2. 동일한 자원을 공유한다
  3. 스레드 간 데이터를 주고 받기 편리하다
  4. 시스템의 성능 향상, 자원의 효율적 관리
- 문제점
  1. Synchronization(동기화) 문제 ➔ Semaphore / Mutex

6. Semaphore

- Pseudo-code

```
function wait(semaphore):
    while(semaphore == 0)
    semaphore--;

function release(semaphore):
    semaphore++;
```

- Implementation

```
// …
wait(semaphore);
// Critical Section
release(semaphore)
// …
```

7. Thread in Java(1)

- java.lang.Thread
- implements Runnable interface


SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

compact1, compact2, compact3
java.lang

Class Thread

java.lang.Object
    java.lang.Thread

All Implemented Interfaces:

Runnable

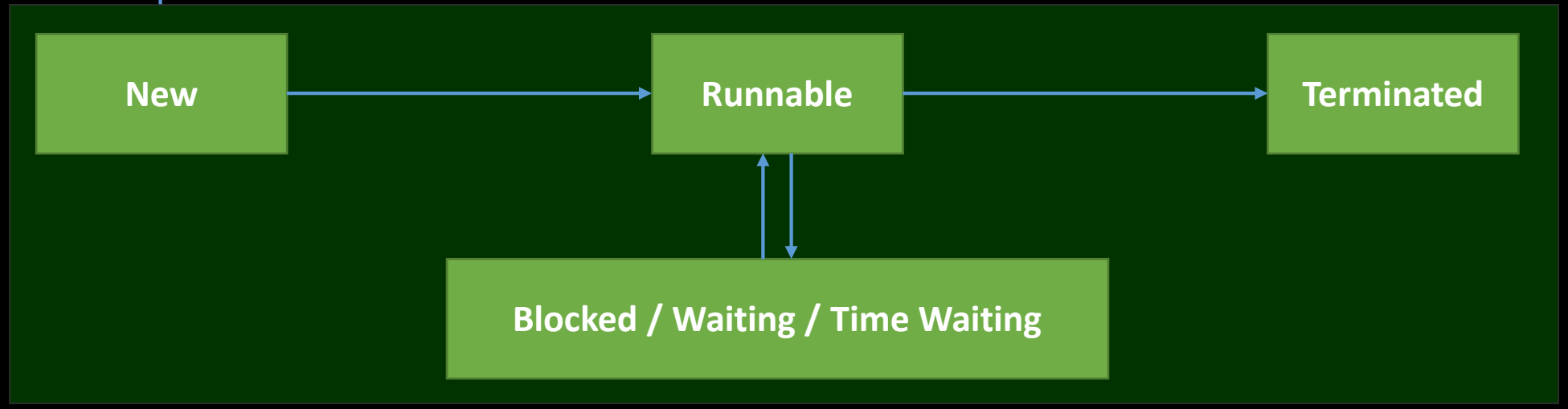
Direct Known Subclasses:

ForkJoinWorkerThread

7. Thread in Java(2)

- Thread 객체 생성
  - Thread thread = new Thread(Runnable target);

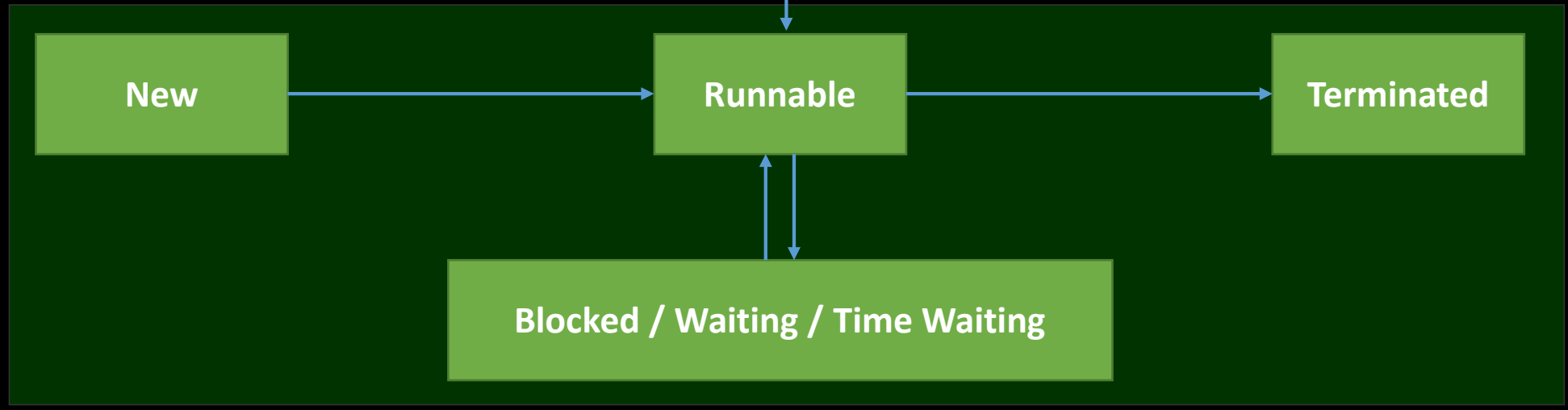
New 상태



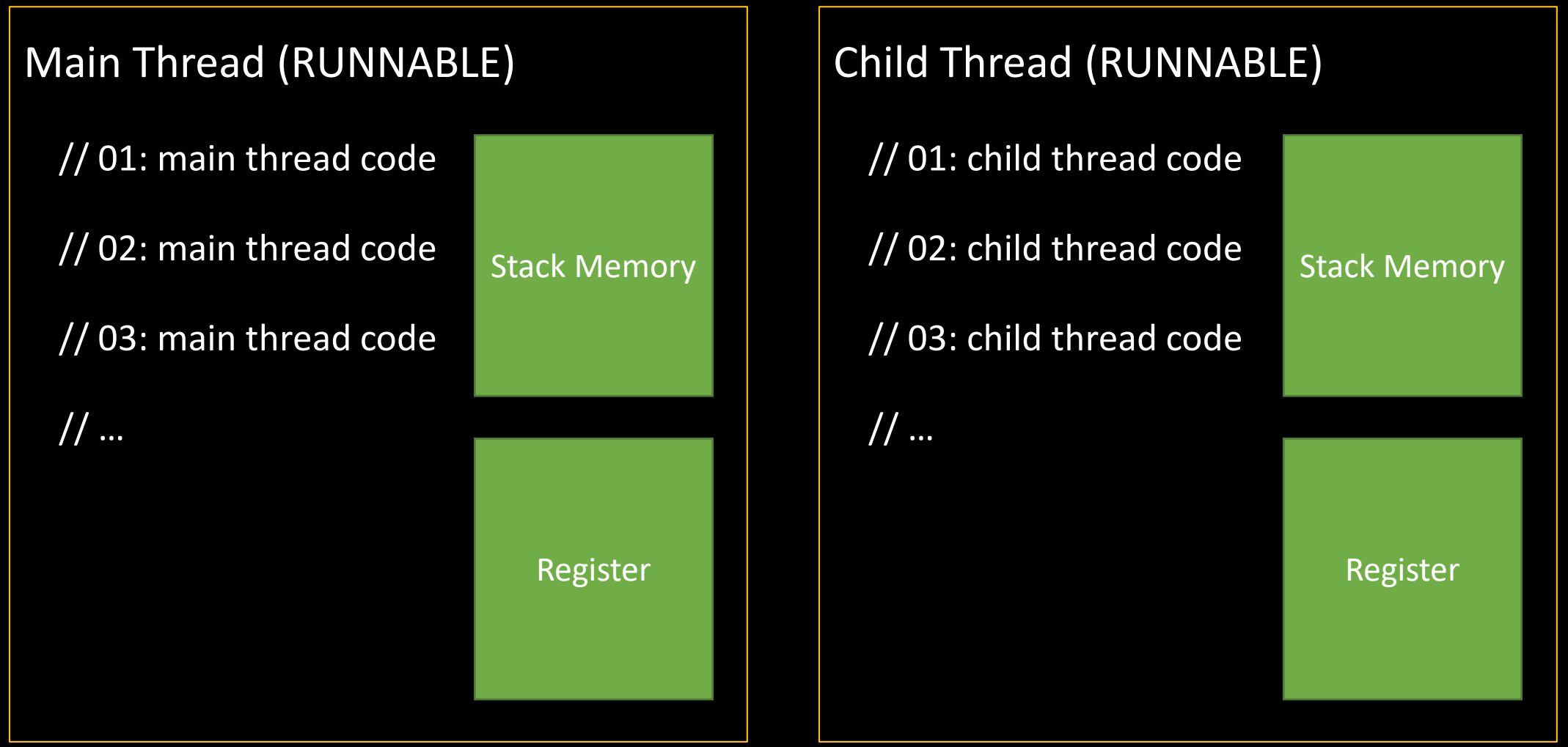
7. Thread in Java(3)

- Thread 실행 가능
  - thread.start();

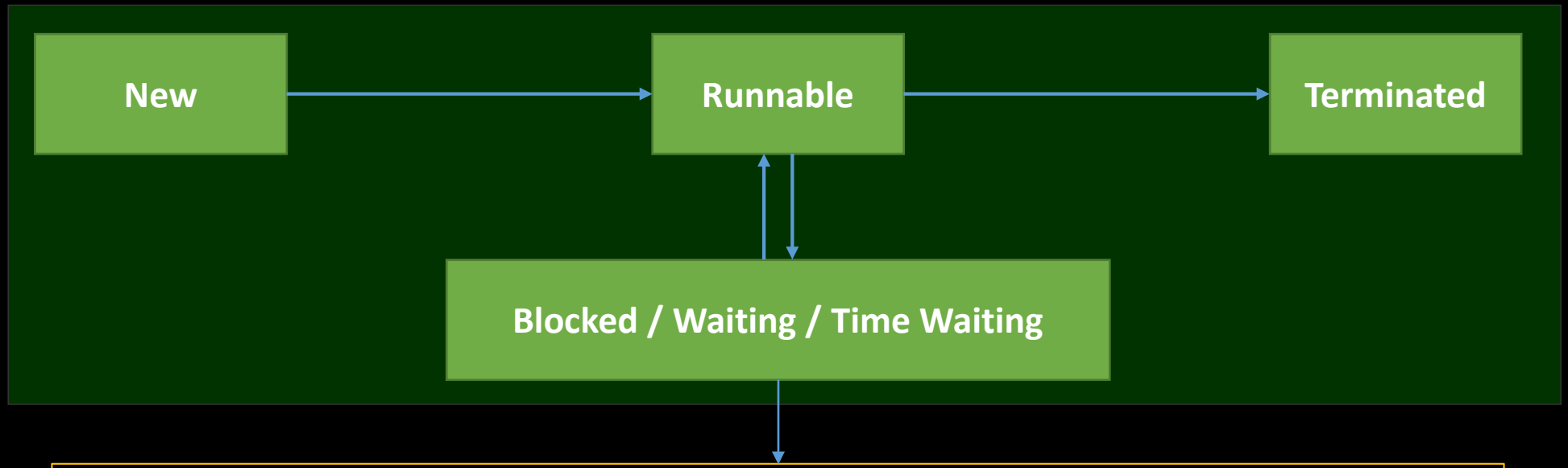
Runnable 상태



7. Thread in Java(4)



7. Thread in Java(5)



- 현재 Runnable하지 않은 스레드
  - 접근이 불가능한 Critical Section에 접근해야 하는 경우
  - 다른 스레드의 종료를 기다려야 하는 경우
  - 특정 시간만큼 sleep하는 경우...

8. ETC

- Synchronization (동기화)
- Thread Pool (스레드 풀)

9. Reference

- https://www.geeksforgeeks.org/multithreading-in-java/
- https://www.geeksforgeeks.org/lifecycle-and-states-of-a-thread-in-java/
- https://www.geeksforgeeks.org/thread-in-operating-system/
- https://www.geeksforgeeks.org/semaphores-in-process-synchronization/
- https://www.geeksforgeeks.org/5-state-process-model-in-operating-system/
- https://docs.oracle.com/javase/8/docs/api/