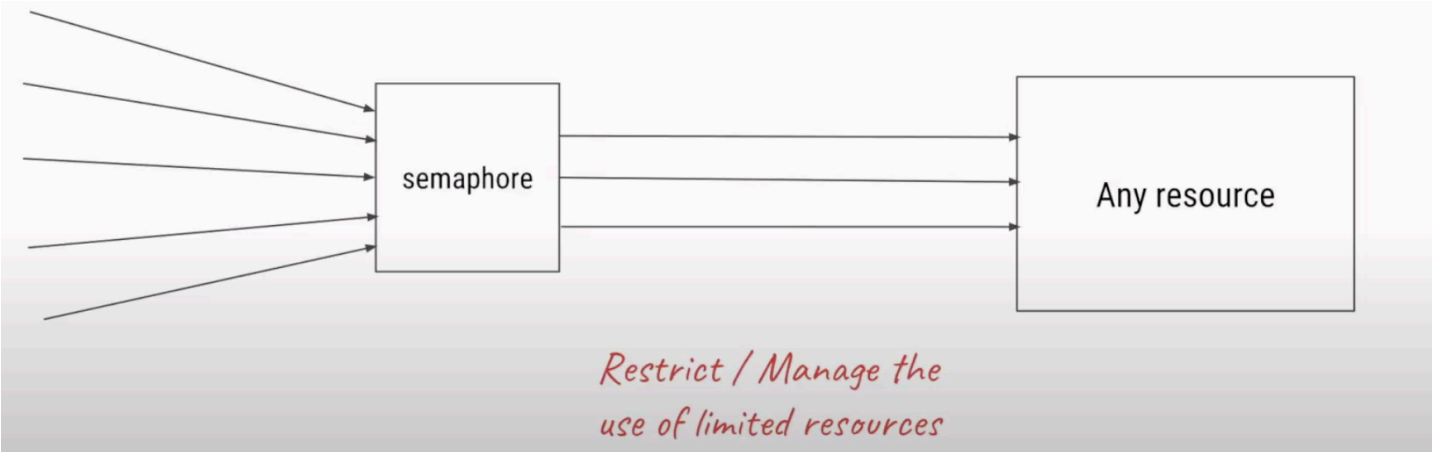
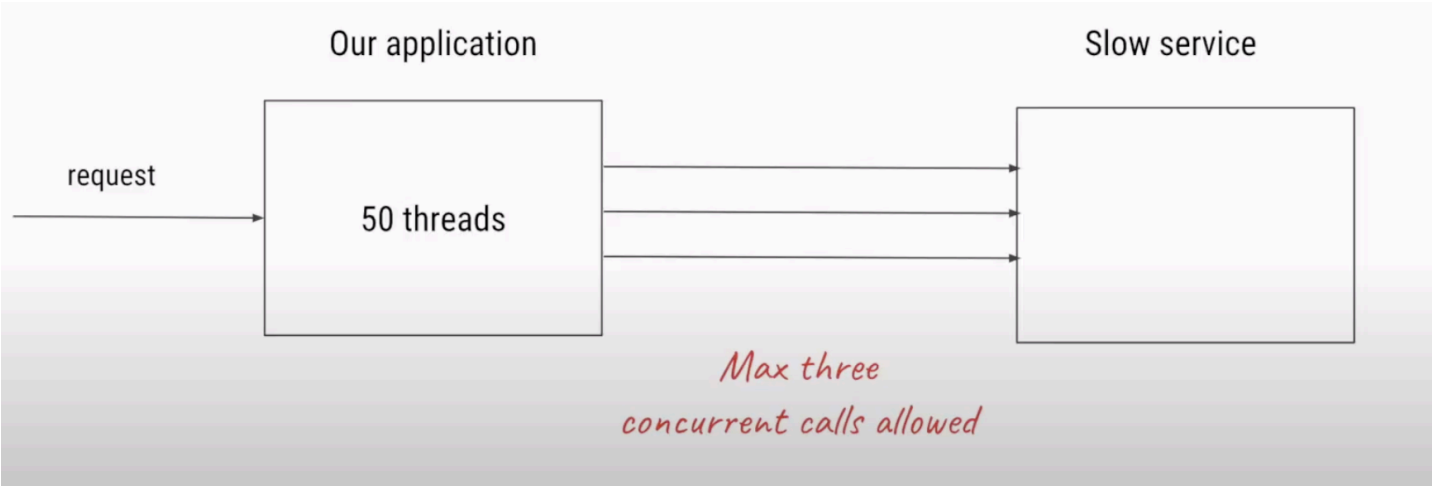
Multithreading 11: Semaphore

It is used for restrict and manage use of resources.



USe case:

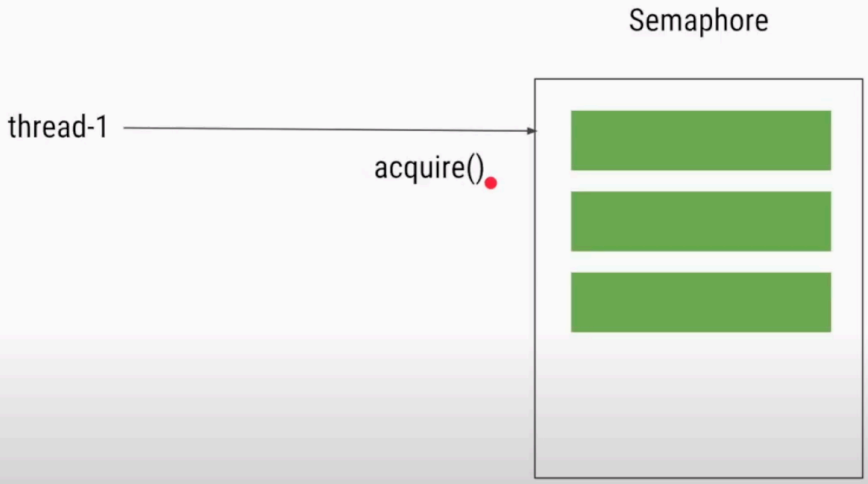


Sample code

```
public static void main(String[] args) throws InterruptedException {

    ExecutorService service = Executors.newFixedThreadPool( nThreads: 50);
    IntStream.of(1000).forEach(i -> service.execute(new Task()));

    service.shutdown();
    service.awaitTermination( timeout: 1, TimeUnit.MINUTES);
}

static class Task implements Runnable {

    @Override
    public void run() {
        // some processing
        // IO call to the slow service
        // rest of processing
    }
}
```

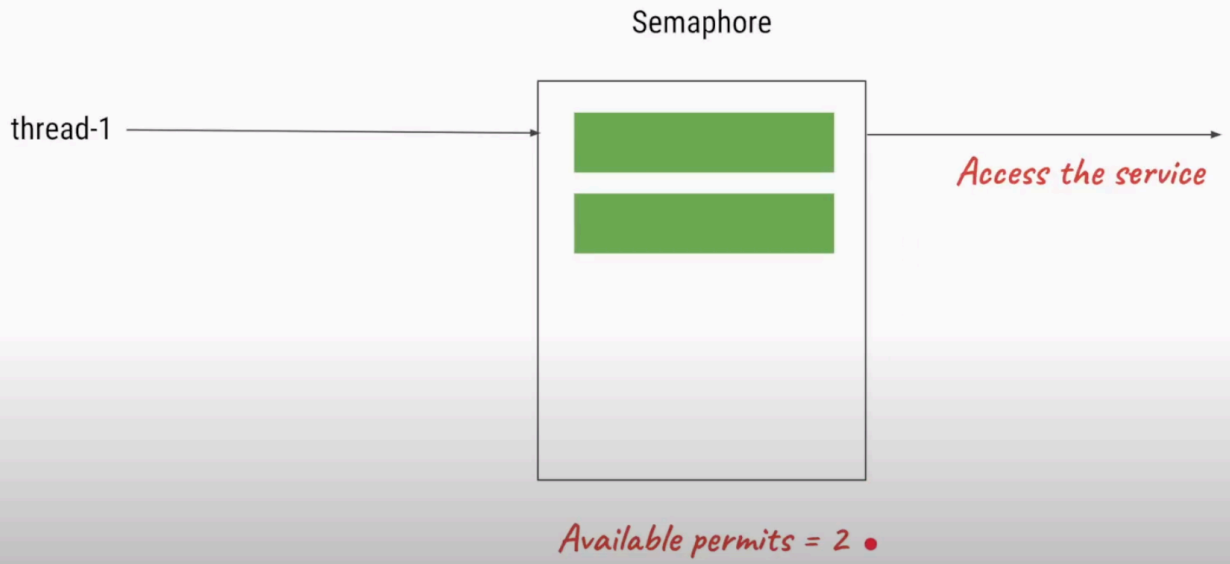
This might be called 50 times concurrently!!

Enter Semaphore - The permit machine

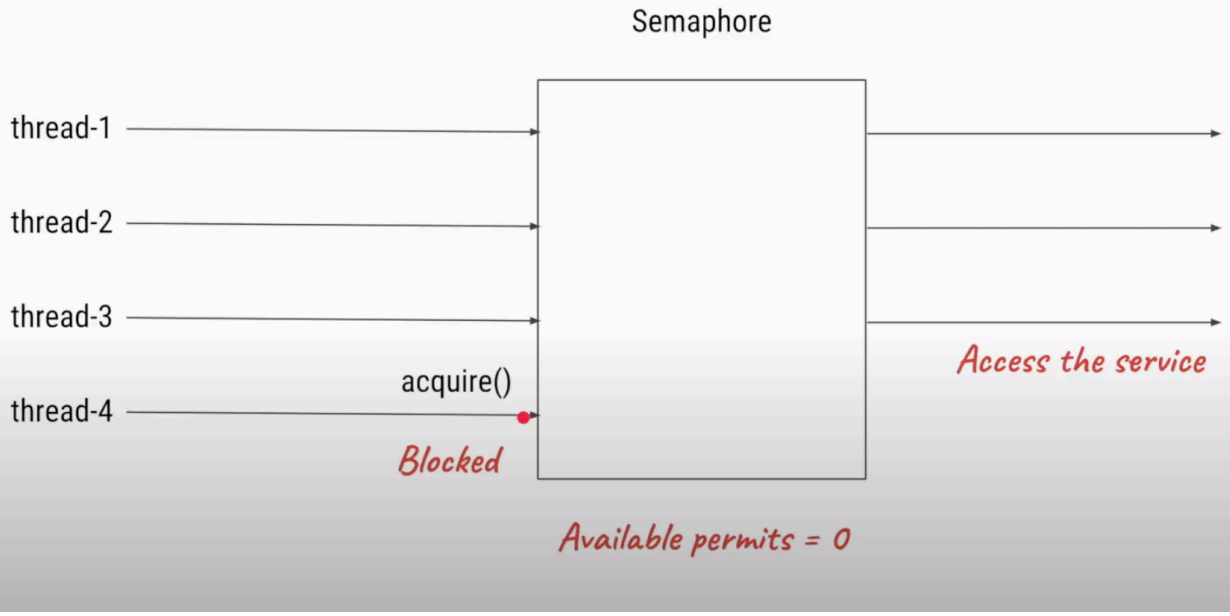


Available permits = 3

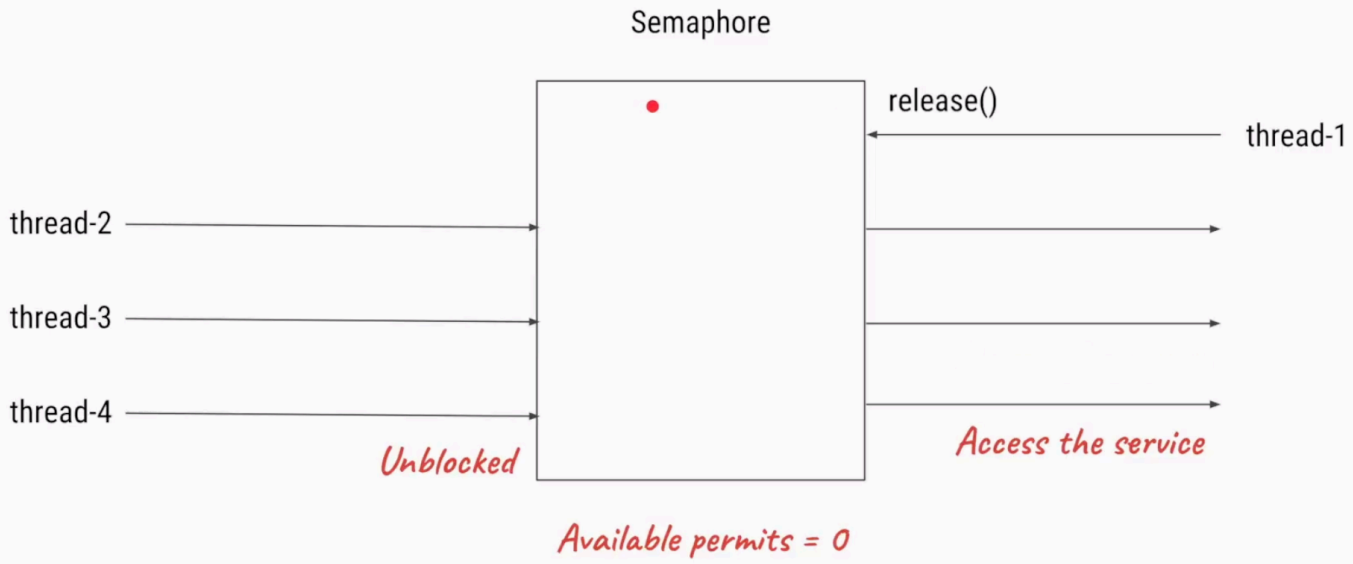
Enter Semaphore - The permit machine



Enter Semaphore - The permit machine



Enter Semaphore - The permit machine



Restrict access

```
public static void main(String[] args) throws InterruptedException {

    Semaphore semaphore = new Semaphore( permits: 3);

    ExecutorService service = Executors.newFixedThreadPool( nThreads: 50);
    IntStream.of(1000).forEach(i -> service.execute(new Task(semaphore)));

    service.shutdown();
    service.awaitTermination( timeout: 1, TimeUnit.MINUTES);
}

static class Task implements Runnable {

    @Override
    public void run() {
        // some processing

        semaphore.acquireUninterruptibly();
        // IO call to the slow service
        semaphore.release();

        // rest of processing
```

Only 3 threads can acquire at a time

Thread can acquire multiple permits

```
static class Task implements Runnable {

    @Override
    public void run() {
        // some processing

        semaphore.acquireUninterruptibly( permits: 2);
        // IO call to the slow service
        semaphore.release( permits: 2);

        // rest of processing
    }
```

Other methods

Method	Meaning
tryAcquire	Try to acquire, if no permit available, do not block. Continue doing something else.
tryAcquire (timeout)	Same as above but with timeout
availablePermits	Returns count of permits available
new Semaphore (count, fairness)	FIFO. Fairness guarantee for threads waiting the longest.