

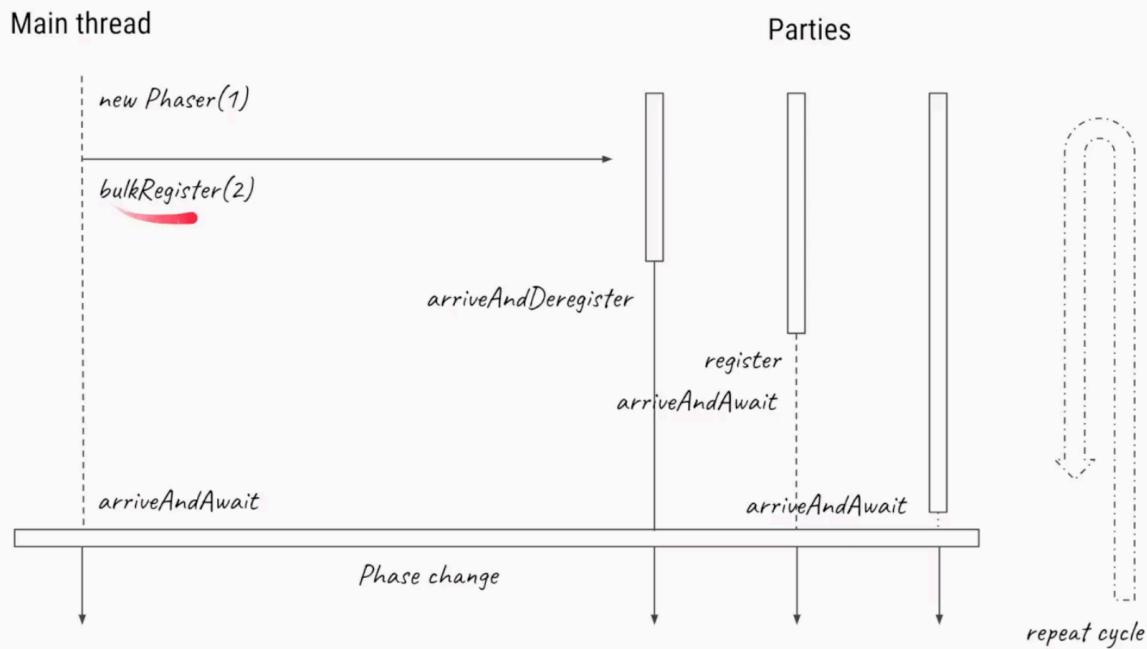
# Multithreading 6 (Part2) : Phaser vs CountDownLatch vs CyclicBarrier

Phaser : Multipal ways to register

```
public static void main(String[] args) {  
    ExecutorService executor = Executors.newFixedThreadPool( nThreads: 4 );  
    Phaser phaser = new Phaser( parties: 1 ); self-register  
    executor.submit(new Service(phaser));  
    executor.submit(new Service(phaser));  
  
    phaser.arriveAndAwaitAdvance();  
    phaser.bulkRegister( parties: 4 ); bulk register later  
}  
  
public static class Service implements Runnable {  
  
    private Phaser phaser;  
    public Service(Phaser phaser) { this.phaser = phaser; }  
  
    @Override  
    public void run() {  
        phaser.register(); Allow threads to  
        // some operations register themselves  
        phaser.arrive();  
        // other operations  
    }  
}
```

Phaser - Multiple ways to register

It could also **deregister** using **phaser.arriveAndDeregister**



## CyclicBarrier

**Phaser method:**

## Phaser methods

<code>new Phaser(partyCount)</code>	<i>Set count of participating parties</i>
<code>register</code>	<i>Allow party to register itself</i>
<code>bulkRegister(partyCount)</code>	<i>Bulk register extra parties post constructor</i>
<code>arrive</code>	<i>Parties can arrive (and continue)</i>
<code>arriveAndAwaitAdvance</code>	<i>Parties can arrive and await for all parties</i>
<code>arriveAndDeregister</code>	<i>Parties can arrive, deregister (and continue)</i>

