Java

Многопоточность. Часть 2

ConcurrentModification

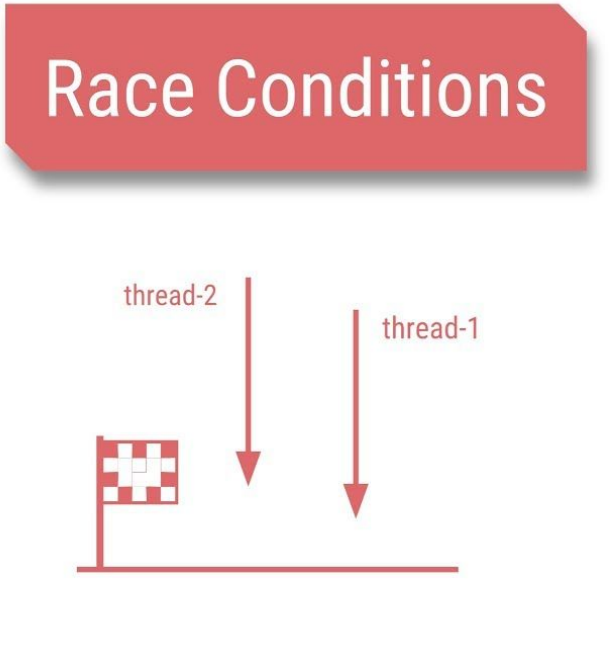
- Можно итерироваться по копии

Объект счётчик и два потока

- Создадим класс со счётчиком
- Создадим runnable вызывающий increment
- Создадим и запустим два потока
- Посмотрим результат

Race Condition

- Несколько потоков зависят друг от друга



Атомарные операции

- Чтение и запись примитивов (int, boolean), кроме long и double
- Чтение и запись ссылок
- Специальные атомарные операции

Compare and Swap (CAS)

```
public class CompareAndSwapCounter
{
    private SimulatedCAS value = new SimulatedCAS();

    public int getCount()
    {
        return value.get();
    }

    public int increment()
    {
        int oldValue = value.get();
        while (value.compareAndSet(oldValue,  newValue: oldValue + 1) == false)
            oldValue = value.get();
        return oldValue + 1;
    }
}
```

Критическая секция

Критическая секция в Java

- Object содержит Mutex
  - Mutex - условно “замок”
  - Имеет состояние (заблокирован или нет)
  - Имеет владельца
- Находится в заголовке класса
- На нём сделан монитор

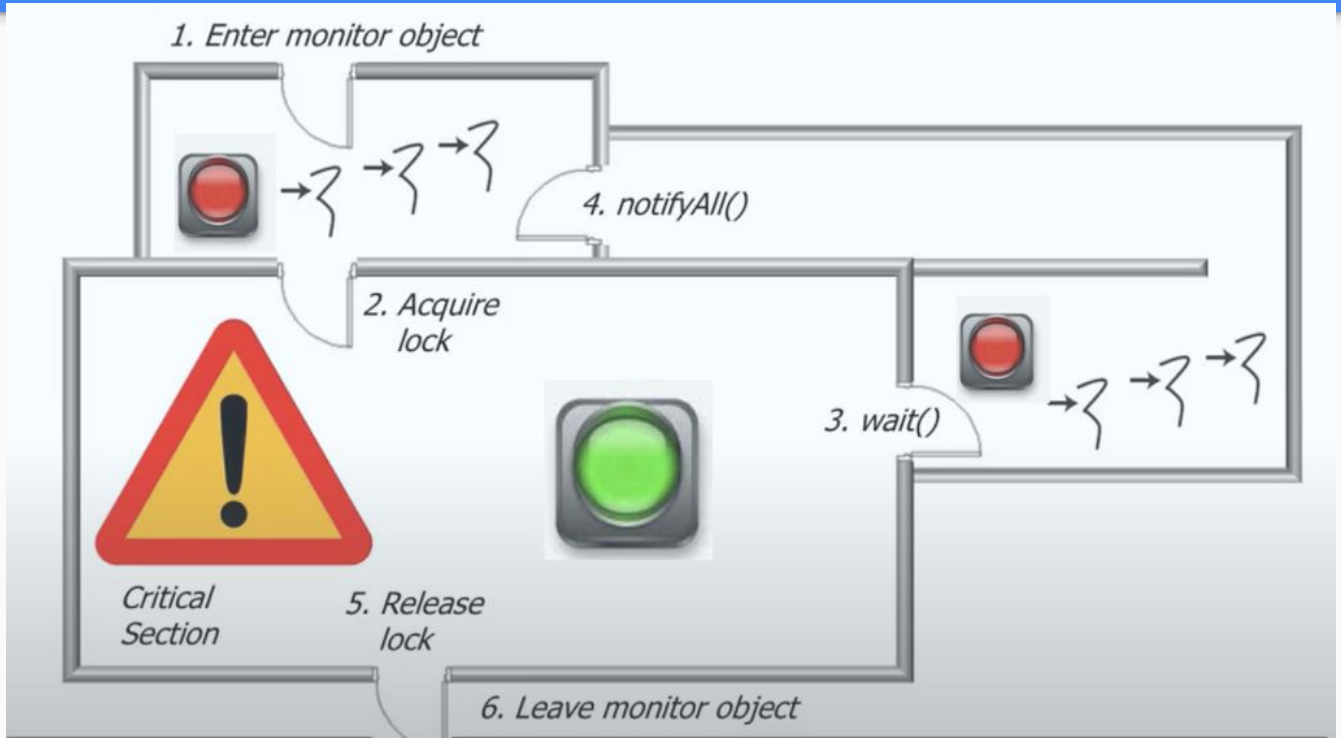
synchronized

```
public class Main {

    private Object obj = new Object();

    public void doSomething() {
        //...какая-то логика, доступная для всех потоков
        synchronized (obj) {
            //логика, которая одновременно доступна только для одного потока
        }
    }
}
```

synchronized

- Синхронизация на переданном объекте
- Синхронизация на this
- Синхронизация метода
- Как синхронизировать методы разных инстансов?
- ThreadSafe в документации

Работа synchronized



Критическая секция и synchronize

```
public class Main
{
    private Object object = new Object();

    public void doSomething() throws InterruptedException {
        // Логика, доступная для всех потоков
        // Ниже идёт логика, доступная только одному потоку
        // До тех пор пока мьютекс занят - любой другой поток спит
        while (object.getMutex().isBusy()) {
            Thread.sleep( millis: 1);
        }

        object.getMutex().setBusy(true);

        object.doImportandStuff();

        object.getMutex().setBusy(false);
    }
}
```

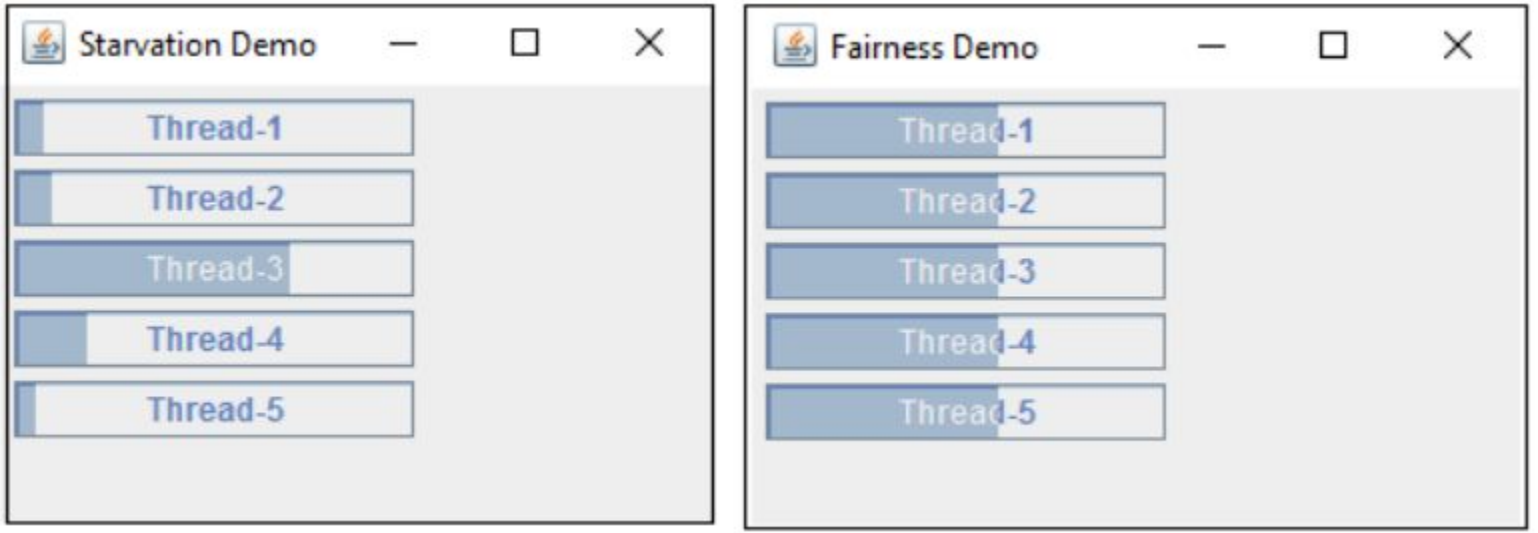
Lock

```
public void doSomething() {
    ReentrantLock lock = new ReentrantLock();
    lock.lock();

    try {
        // Some code
    } finally {
        lock.unlock();
    }
}
```

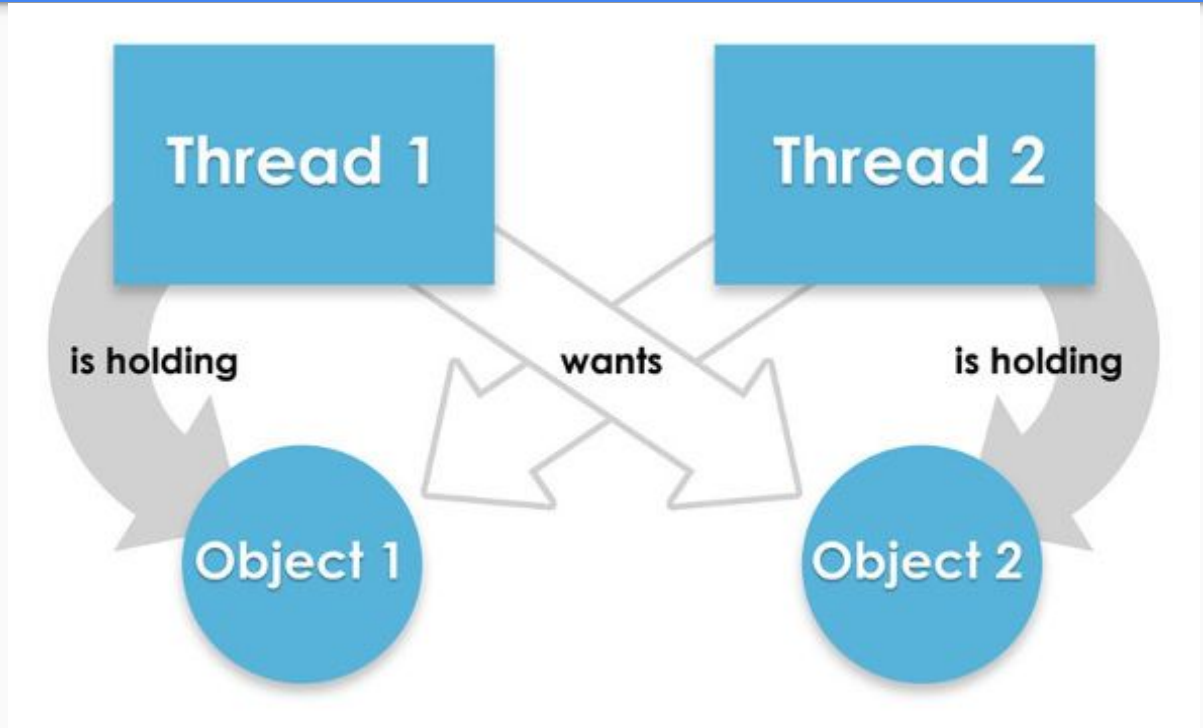
Проблемы

- Race Condition
- Starvation
- Deadlock

Starvation


Starvation Demo
Thread-1
Thread-2
Thread-3
Thread-4
Thread-5
Fairness Demo
Thread-1
Thread-2
Thread-3
Thread-4
Thread-5

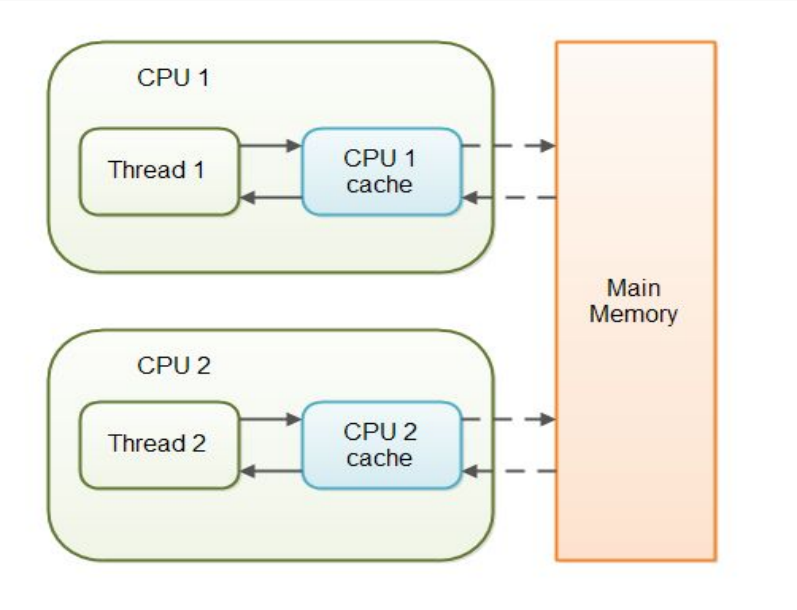
Deadlock

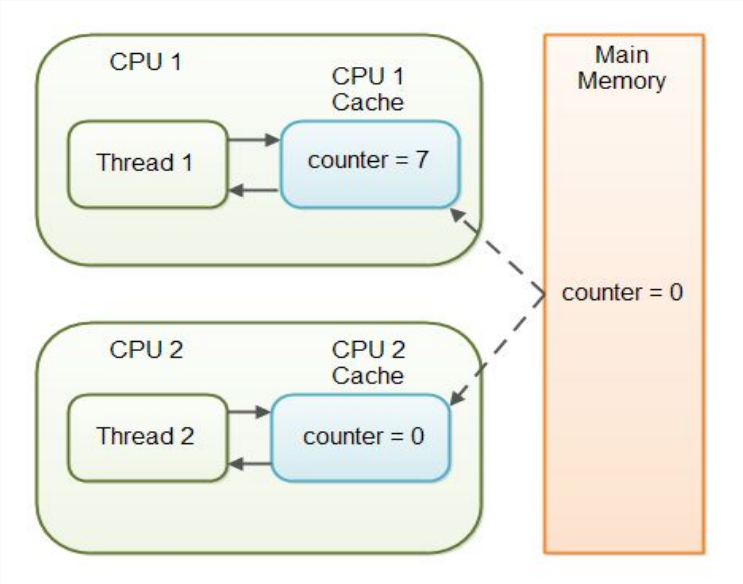

Thread 1
Thread 2
is holding
wants
is holding
Object 1
Object 2

Семафор

Задача о философах

Livelock: Volatile





ReadWriteLock

ReentrantReadWriteLock:

- readLock()
- writeLock()

Полезные ссылки

Просто и коротко о потоках:

- https://javarush.ru/groups/posts/1992-mnogopotochnostjh-v-java-sutjh-pljusih-i-chastihe-lovushki- - многопоточность в Java: суть, «плюсы» и частые ловушки
- https://javarush.ru/groups/posts/1993-mnogopotochnostjh-chto-delajut-metodih-klassa-thread - многопоточность: что делают методы класса Thread
- https://javarush.ru/groups/posts/2174-v-chem-raznica-mezhdu-mjhjuteksom-monitorom-i-semaforom - в чем разница между мьютексом, монитором и семафором
- https://javarush.ru/groups/posts/1994-sinkhronizacija-potokov-operator-synchronized - синхронизация потоков. Оператор synchronized

Сложно и подробно о потоках:

- https://habr.com/ru/post/164487/ - многопоточность в Java
- https://habr.com/ru/post/326146/ - многопоточность на корабликах (интересный материал)
- https://habr.com/ru/post/326146/ - пул потоков