**Синхронизация потоков. Оператор synchronized**

При работе потоки нередко обращаются к каким-то общим ресурсам, которые определены вне потока, например, обращение к какому-то файлу. Если одновременно несколько потоков обратятся к общему ресурсу, то результаты выполнения программы могут быть неожиданными и даже непредсказуемыми.

Синхронизация потоков – настройка взаимодействия потоков между собой.

В многопоточном программировании ввели специальное понятие мьютекс (от англ. «mutex», «mutual exclusion» — «взаимное исключение»).

Задача мьютекса — обеспечить такой механизм, чтобы доступ к объекту в определенное время был только у одного потока. Если Поток-1 захватил мьютекс объекта А, остальные потоки не получат к нему доступ, чтобы что-то в нем менять. До тех пор, пока мьютекс объекта А не освободится, остальные потоки будут вынуждены ждать.

**Synchronized**

Им помечается определенный кусок нашего кода. Если блок кода помечен ключевым словом synchronized, это значит, что блок может выполняться только одним потоком одновременно.

Синхронизацию можно реализовать по-разному. Например, создать целый синхронизированный метод:

```
public synchronized void doSomething() {

    //...логика метода
}
```

Или же написать блок кода, где синхронизация осуществляется по какому-то объекту:

```
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
        }
    }
}
```

Смысл прост. Если один поток зашел внутрь блока кода, который помечен словом synchronized, он моментально захватывает мьютекс объекта, и все другие потоки, которые попытаются зайти в этот же блок или метод вынуждены ждать, пока предыдущий поток не завершит свою работу и не освободит монитор.

Каждый объект в Java имеет ассоциированный с ним **монитор**. Монитор представляет своего рода инструмент для управления доступа к объекту. Когда выполнение кода доходит до оператора synchronized, монитор объекта блокируется, и на время его блокировки монопольный доступ к блоку кода имеет только один поток, который и произвел блокировку. После окончания работы блока кода, монитор объекта освобождается и становится доступным для других потоков.

После освобождения монитора его захватывает другой поток, а все остальные потоки продолжают ожидать его освобождения.