高效并发-线程安全与锁优化

# 主要是三个大问题：

## Java中的线程安全；

## 线程安全的实现方法；

## 锁优化。

# Java中的线程安全

（这里的线程安全限定于多个线程之间存在共享数据访问）

按照线程安全的“安全程度”由强到弱来排序，可以将Java中各种操作共享的数据分成以下5类：

**不可变、绝对线程安全、相对线程安全、线程兼容和线程对立。**

## 不可变

主要是通过final关键字实现。（具体见final关键字的使用方法介绍）

## 绝对线程安全

## 相对线程安全

Java中大部分线程安全类都属于相对线程安全。如Vector、HashTable等。

## 线程兼容

## 线程对立

# 线程安全的实现方法

## 互斥同步(阻塞同步)

## 非阻塞同步

## 无同步方案

### 可重入代码(Reentrant Code)

### 线程本地存储

java.lang.ThreadLocal类。

# Java中的锁优化

## 自旋锁与自适应自旋锁

## 锁消除

## 锁粗话

同步范围扩大。

场景：循环体中存在加锁和解锁，将加锁同步的范围扩展(粗话)到整个操作序列的外部。

## 轻量级锁

## 偏向锁