Java Unsafe 类

简单介绍

Unsafe 类使 Java 拥有了像 C 语言的指针一样**操作内存空间的能力**，同时也带来了指针的问题。

过度的使用 Unsafe 类会使得出错的几率变大，因此 Java 官方并不建议使用的，官方文档也几乎没有。

Oracle 正在计划从 Java 9 中去掉 Unsafe 类

主要功能有三个

操纵对象属性

操纵对象属性，主要落在这个方法上：

```
public native long objectFieldOffset(Field f);
```

通过此方法，可以获取对象中某个属性的内存偏移地址，然后可根据偏移地址直接对属性进行修改，属性是否可读都无所谓，都能修改。

```
Field name = user.getClass().getDeclaredField("name");
long nameOffset = unsafe.objectFieldOffset(name);
unsafe.putObject(user, nameOffset, "jim");
```

操纵数组元素

操纵数组元素，主要涉及两个接口。

```
public native int arrayBaseOffset(Class arrayClass);
public native int arrayIndexScale(Class arrayClass);
```

arrayBaseOffset，获取数组第一个元素的偏移地址。

arrayIndexScale，获取数组中元素的增量地址。

arrayBaseOffset 与 arrayIndexScale 配合起来使用，就可以定位数组中每个元素在内存中的位置

例如：索引为 i 的元素可以使用如下代码定位：

```
int baseOffset = unsafe.arrayBaseOffset(array.getClass());
int indexScale = unsafe.arrayIndexScale(array.getClass());
baseOffset + i*indexScale
```

线程挂起与恢复、CAS

如果我们使用 ReentrantLock 进行多线程开发，当一个线程抢占锁失败时，线程将被挂起，实现线程挂起的正是 Unsafe 类。

将一个线程进行挂起是通过 park 方法实现的，调用 park 后，线程将一直阻塞直到超时或者中断等条件出现。unpark 可以终止一个挂起的线程，使其恢复正常。整个并发框架中对线程的挂起操作被封装在 LockSupport 类中，LockSupport 类中有各种版本 pack 方法，但最终都调用了 Unsafe.park()方法。

CAS，一种乐观锁机制，如果对象当前值和期望值一样，那么则将对象的值设置成新值。和悲观锁不一样，它不需要抢占锁，它是一种尝试性的，能有效地提高效率，它的全称是 compareAndSwap ，依赖于硬件的原子操作实现。详细的 CAS 请参考 线程安全 一文。