链滴

谈谈强引用、软引用、弱引用、幻象引用?

作者：ibut
原文链接：https://ld246.com/article/1604654826710
来源网站：链滴

我们知道在Java中除了基础的数据类型以外，其它的都为引用类型。而Java根据其生命周期的长短将用类型又分为强引用、软引用、弱引用、幻象引用 。正常情况下我们平时基本上我们只用到强引用类，而其他的引用类型我们也就在面试中，或者平日阅读类库或其他框架源码的时候才能见到。

1.强引用

我们平日里面的用到的new了一个对象就是强引用，例如 Object obj = new Object();

当JVM的内存空间不足时，宁愿抛出OutOfMemoryError使得程序异常终止也不愿意回收具有强引用存活着的对象！记住是存活着，不可能是你new一个对象就永远不会被GC回收。

当一个普通对象没有其他引用关系，只要超过了引用的作用域或者显示的将引用赋值为null时，你的象就表明不是存活着，这样就会可以被GC回收了。当然回收的时间是不一定的具体得看GC回收策略。

2.软引用

软引用的生命周期比强引用短一些。在JDK 1.2之后，提供了SoftReference类来实现软引用。当JVM为内存空间不足时，就回去试图回收软引用指向的对象，也就是说在JVM抛出OutOfMemoryError之，会去清理软引用对象。

```
Object obj = new Object();
SoftReference softObj = new SoftReference(obj);
obj = null;  //去除强引用
```

这样就是一个简单的软引用使用方法。通过get()方法获取对象。

```
Object obj = new Object();
ReferenceQueue queue = new ReferenceQueue();
SoftReference softObj = new SoftReference(obj,queue);
obj = null;  //去除强引用
```

原文链接：谈谈强引用、软引用、弱引用、幻象引用?

软引可以与引用队列(ReferenceQueue)联合使用。

当softObj软引用的obj被GC回收之后，softObj 对象就会被塞到queue中，之后我们可以通过这个队的poll()来检查你关心的对象是否被回收了，如果队列为空，就返回一个null。反之就返回软引用对象就是softObj。软引用一般用来实现内存敏感的缓存，如果有空闲内存就可以保留缓存，当内存不足就清理掉，这样就保证使用缓存的同时不会耗尽内存。例如图片缓存框架中缓存图片就是通过软引用。

3.弱引用

回收就会死亡：它的生命周期比软引用还要短，也是通过get()方法获取对象。被弱引用关联的对象实只能生存到下一次垃圾收集发生之前。在GC的时候，不管内存空间足不足都会回收这个对象。在JDK 12之后，提供了WeakReference类来实现弱引用。

```
Object obj = new Object();
WeakReference<Object> weakObj = new WeakReference<Object>(obj
obj = null;  //去除强引用
```

同样也可以配合ReferenceQueue 使用，也同样适用于内存敏感的缓存。**ThreadLocal中的key就用了弱引用。**

4.虚引用

也称幻象引用，它是最弱的一种引用关系。在JDK 1.2之后，提供了PhantomReference类来实现虚用。一个对象实例是否有虚引用的存在，完全不会对其生存时间构成影响，也无法通过虚引用来取得个对象实例。任何时候可能被GC回收，就像没有引用一样。

```
Object obj = new Object();
ReferenceQueue queue = new ReferenceQueue();
PhantomReference<Object> phantomObj = new PhantomReference<Object>(obj , queue);
obj = null;  //去除强引用
```

无法通过虚引用访问对象的任何属性或者函数。那就要问了要它有什么用？虚引用仅仅只是提供了一确保对象被finalize以后来做某些事情的机制。比如说这个对象被回收之后发一个系统通知啊啥的。

虚引用是必须配合ReferenceQueue 使用的，具体使用方法和上面提到软引用的一样。主要用来跟对象被垃圾回收的活动。