Java 对象及其引用

关于对象与引用之间的一些基本概念。

初学 Java 时，在很长一段时间里，总觉得基本概念很模糊。后来才知道，在许多 Java 书中，把对象和对象的引用混为一谈。可是，如果我分不清对象与对象引用，

那实在没法很好地理解下面的面向对象技术。把自己的一点认识写下来，或许能让初学 Java 的朋友们少走一点弯路。

为便于说明，我们先定义一个简单的类：

```
class Vehicle {
int passengers;
int fuelcap;
int mpg;
        }
```

有了这个模板，就可以用它来创建对象：

```
Vehicle veh1 = new Vehicle();
```

通常把这条语句的动作称之为创建一个对象，其实，它包含了四个动作。

1）右边的"new Vehicle"，是以 Vehicle 类为模板，在堆空间里创建一个 Vehicle 类对象（也简称为 Vehicle 对象）。

2）末尾的()意味着，在对象创建后，立即调用 Vehicle 类的构造函数，对刚生成的对象进行初始化。构造函数是肯定有的。如果你没写，Java 会给你补上一个默认的构造函数。

3）左边的"Vehicle veh 1"创建了一个 Vehicle 类引用变量。所谓 Vehicle 类引用，就是以后可以用来指向 Vehicle 对象的对象引用。

4）"="操作符使对象引用指向刚创建的那个 Vehicle 对象。

我们可以把这条语句拆成两部分：

```
Vehicle veh1;
veh1 = new Vehicle();
```

效果是一样的。这样写，就比较清楚了，有两个实体：一是对象引用变量，一是对象本身。

在堆空间里创建的实体，与在数据段以及栈空间里创建的实体不同。尽管它们也是确确实实存在的实体，但是，我们看不见，也摸不着。不仅如此，

我们仔细研究一下第二句，找找刚创建的对象叫什么名字？有人说，它叫"Vehicle"。不对，"Vehicle"是类（对象的创建模板）的名字。

一个 Vehicle 类可以据此创建出无数个对象，这些对象不可能全叫"Vehicle"。

对象连名都没有，没法直接访问它。我们只能通过对象引用来间接访问对象。

为了形象地说明对象、引用及它们之间的关系，可以做一个或许不很妥当的比喻。对象好比是一只很大的气球，大到我们抓不住它。引用变量是一根绳， 可以用来系汽球。

如果只执行了第一条语句，还没执行第二条，此时创建的引用变量 veh1 还没指向任何一个对象，它的值是 null。引用变量可以指向某个对象，或者为 null。

它是一根绳，一根还没有系上任何一个汽球的绳。执行了第二句后，一只新汽球做出来了，并被系在 veh1 这根绳上。我们抓住这根绳，就等于抓住了那只汽球。

再来一句：

Vehicle veh2;

就又做了一根绳，还没系上汽球。如果再加一句：

veh2 = veh1;

系上了。这里，发生了复制行为。但是，要说明的是，对象本身并没有被复制，被复制的只是对象引用。结果是，veh2 也指向了 veh1 所指向的对象。两根绳系的是同一只汽球。

如果用下句再创建一个对象：

veh2 = new Vehicle();

则引用变量 veh2 改指向第二个对象。

从以上叙述再推演下去，我们可以获得以下结论：

（1）一个对象引用可以指向 0 个或 1 个对象（一根绳子可以不系汽球，也可以系一个汽球）；

（2）一个对象可以有 N 个引用指向它（可以有 N 条绳子系住一个汽球）。

如果再来下面语句：

veh1 = veh2;

按上面的推断，veh1 也指向了第二个对象。这个没问题。问题是第一个对象呢？没有一条绳子系住它，它飞了。多数书里说，它被 Java 的垃圾回收机制回收了。

这不确切。正确地说，它已成为垃圾回收机制的处理对象。至于什么时候真正被回收，那要看垃圾回收机制的心情了。

由此看来，下面的语句应该不合法吧？至少是没用的吧？

new Vehicle();

不对。它是合法的，而且可用的。譬如，如果我们仅仅为了打印而生成一个对象，就不需要用引用变量来系住它。最常见的就是打印字符串：

System.out.println(“I am Java!”);

字符串对象“I am Java!”在打印后即被丢弃。有人把这种对象称之为临时对象。

对象与引用的关系将持续到对象回收

Java 在运行时才处理别名引用