值类型和引用类型

C#语言的类型被分为两类——值类型和引用类型。这两种类型的对象在内存中的存储方式不同。数据类型不仅定义了存储数据需要的内存大小、组成该类型的数据成员以及该类型能执行的函数。还决定了对象在内存中的存储位置——栈或堆。

栈是一个内存数组，是一个 LIFO（last-in first-out，后进先出）的数据结构。栈存储几种类型的数据：

√变量的值；

√程序当前的执行环境；

√传递给方法的参数。

堆是一块内存区域，在堆里可以分配大块的内存用于存储某类型的数据。与栈不同，堆里的内存可以任意顺序存入和删除。虽然程序可以在堆里保存数据，但并不能显式地删除它们。CLR 的自动 GC（Garbage Collector，垃圾收集器）在判断出程序的代码将不会再访问某数据项时，自动清除无主的堆对象。

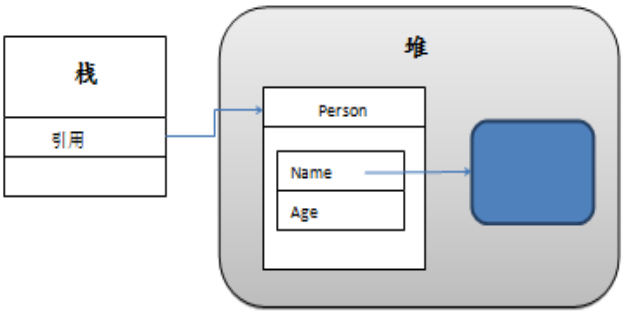
值类型只需要一段单独的内存，用于存储实际的数据。

引用类型需要两段内存：第一段存储实际的数据，它总是位于堆中。第二段是一个引用，指向数据在堆中的存放位置。

例如，假设有一个引用类型 Person 的实例，名称为 person1，它有两个成员：一个值类型成员 Age 和一个引用类型成员 Name。它将如何存储呢？是否是值类型的成员 Age 存储在栈里，而引用类型的成员 Name 在栈和堆之间分成两半呢？答案是否定的。

对于一个引用类型，其实例的数据部分始终存放在堆里。既然两个成员都是对象数据的一部分，那么它们都会被存放在堆里，无论它们是值类型还是引用类型。

尽管成员 Age 是值类型，但它也是 Person 实例数据的一部分，因此和对象的数据一起被存放在堆里。成员 Name 是引用类型，所以它的数据部分会始终存放在堆里。



引用类型成员的数据存储