《Java8 实战》- 读书笔记第一章（01）

作者：Not-Found
原文链接：https://ld246.com/article/1533019322470
来源网站：链滴

Blog:http://www.codedream.xin

最近一直想写点什么东西，却不知该怎么写，所以就写写关于看《Java8实战》的笔记吧。

第一章内容较多，因此打算分几篇文章来写。

为什么要关心Java8

自1996年JDK（1.0）发布以来，Java已经受到了学生、项目经理和 程序员等一大批活跃的用户的欢。这一语言极富活力，不断被用在大大小小的项目里。从Java1.1（1997）年一直到Java7（2011），Java通过增加新功能，不断得到良好的升级。Java8则是在2014年3月发布的。那么问题来了：为么你应该关心Java8?

是的，Java一直被吐槽写起来太啰嗦，没有IDE的快捷键和自动生成，简直就是在用生命写代码，因太浪费时间。

例如，最简单的HelloWorld：

```
public class HelloWorld {
    public static void main (String[] args) {
        System.out.println("HelloWorld");
    }
}
```

都要这写么多代码，不使用快捷键或者语法模板去生成，真的很浪费时间。（初学者请忽略）

所以，为了解决这个问题，Java8中推出了核心新特性之一：Lambda（匿名函数）

Lambda表达式，是一个很不错很实用的一个新特性，如果你使用了这个新特性，也许会爱不释手。

举个例子，比如我们对苹果进行按照重量进行排序，也许我们会这样写：

```
private static List<Apple> apples = Arrays.asList(new Apple(100, "red"),
    new Apple(101, "green"), new Apple(132, "green"),
    new Apple(90, "green"), new Apple(122, "red")
);

Collections.sort(apples, new Comparator<Apple>() {
    public int compare(Apple o1, Apple o2) {
        return o1.getWeight() < o2.getWeight() ? -1 :
                ((o1.getWeight() == o2.getWeight()) ? 0 : 1);
    }
});
```

在Java8里，你可以这样写，这样写看起来更接近问题的描述：

```
apples.sort(Comparator.comparing(Apple::getWeight));
```

是不是有点心动啊，本来需要五六行解决的排序的代码，现在只要一行即可！趁热打铁，继续吧。

Java8里面将代码传递给方法的功能（同时也能够放回代码并将其包含在数据结构中），还让我们能使用一整套技巧，通常称为函数式编程。

现在你需要筛选一个目录中的所有隐藏文件，你会怎么做?

大部分人立马会想到，File类中不就是有一个isHidden的方法吗？使用这个方法就可以判断哪些是隐

文件啦。

是的，如下所示：

```
File[] files = new File("D:\\.").listFiles(new FileFilter() {
    @Override
    public boolean accept(File pathname) {
        return pathname.isHidden();
    }
});
```

看起来很简单，很明了嘛！那还可以不可以继续优化简短一下呢？答案是当然可以的。

如下所示：

```
File[] files = new File("D:\\.").listFiles(File::isHidden);
```

太酷了，有了函数isHidden，因此只需要使用Java8的方法引用语法::（即“把这个方法作为值”）将传给listFiles方法就可以了。

代码传递：一个例子

来看看一个例子，看看它是如何帮你你写程序的。依旧使用刚刚对苹果排序的代码。现在，要做的是选出所有的绿苹果，也许你会这一个这样的方法filterGreenApples：

```
public static List<Apple> filterGreenApples(List<Apple> apples) {
    List<Apple> result = new ArrayList<>();
    for (Apple apple : apples) {
        if ("green".equals(apple.getColor())) {
            result.add(apple);
        }
    }
    return result;
}
```

在Java8之前，基本上都是这样写的，看起来也没什么毛病。但是，现在又要筛选一下重量超过120的苹果。哦，一想很简单嘛，把上面的代码复制、粘贴改一下就好啦：

```
public static List<Apple> filterHeavyApples(List<Apple> apples) {
    List<Apple> result = new ArrayList<>();
    for (Apple apple : apples) {
        if (apple.getWeight() > 120) {
            result.add(apple);
        }
    }
    return result;
}
```

虽然简单，但是还是出现了一些重复地方，看起来不太好。这两段代码的差异只是条件不同，那么只要把接受重量的上下限作为参数传递给filter就可以了，使用Java8来优化一下这些代码：

```
    public static void main (String[] args) {
        // 筛选出绿色苹果
        List<Apple> greenApples = filterApples(apples, FilterApples::isGreenApple);
        System.out.println(greenApples);
```

```
        // 筛选重量大于120克的苹果
        List<Apple> heavyApples = filterApples(apples, FilterApples::isHeavyApple);
        System.out.println(heavyApples);
    }

    public static boolean isGreenApple(Apple apple) {
        return "green".equals(apple.getColor());
    }

    public static boolean isHeavyApple(Apple apple) {
        return apple.getWeight() > 120;
    }

    public static List<Apple> filterApples(List<Apple> apples, Predicate<Apple> predicate) {
        List<Apple> result = new ArrayList<>();
        for (Apple apple : apples) {
            if (predicate.test(apple)) {
                result.add(apple);
            }
        }
        return result;
    }

    interface Predicate<T> {
        /**
         * 根据给定的参数计算此谓词
         *
         * @param t
         * @return
         */
        boolean test(T t);
    }
```

在这段代码中，自定义了一个接口Predicate<T>，中文意思是谓词：

什么是谓词?

前面的代码传递了方法Apple::isGreenApple（它接受参数Apple并返回一个boolean）给filterApple。后者希望接受一个Predicate<Apple>参数。谓词（predicate）在数学上常常用来代表一个类似函的东西，它接受一个参数值，并返回true或false。

当然，Java8中已经有了一个Predicate<T>接口，因此，我们也不需要去定义一个这样的接口啦。

使用自定义的Predicate接口中的方法，你创建了一个方法引用，你无须去关注test方法是如何实现的你只要知道你引用的某个方法即可。

读书笔记系列的文章，因为我的文笔不是很好，一些观点可能描述的不是那么的好，文章中有误的地还请各位读者指正，非常感谢。