第二十二章 新特性

22.1 Java8的新特性

22.1.1 Java8的概述

- Java8是 Java 语言的一个重要版本，该版本于2014年3月发布，是自Java5以来最具革命性的版本，这个版本包含语言、编译器、库、工具和JVM等方面的十多个新特性。

22.1.2 函数式接口

- 函数式接口主要指只包含一个抽象方法的接口，如：java.lang.Runnable、java.util.Comparator接口等。
- Java8提供@FunctionalInterface注解来定义函数式接口，若定义的接口不符合函数式的规范便会报错。
- Java8中增加了java.util.function包，该包包含了常用的函数式接口，具体如下：

接口名称	方法声明	功能介绍
Consumer	void accept(T t)	根据指定的参数执行操作
Supplier	T get()	得到一个返回值
Function<T,R>	R apply(T t)	根据指定的参数执行操作并返回
Predicate	boolean test(T t)	判断指定的参数是否满足条件

22.1.3 Lambda表达式

- Lambda 表达式是实例化函数式接口的重要方式，使用 Lambda 表达式可以使代码变的更加简洁紧凑。
- lambda表达式：参数列表、箭头符号->和方法体组成，而方法体中可以是表达式，也可以是语句块。
- 语法格式：(参数列表) -> { 方法体; }　　- 其中()、参数类型、{} 以及return关键字 可以省略。

22.1.4 方法引用

- 方法引用主要指通过方法的名字来指向一个方法而不需要为方法引用提供方法体，该方法的调用交给函数式接口执行。
- 方法引用使用一对冒号 :: 将类或对象与方法名进行连接，通常使用方式如下：
    - 对象的非静态方法引用　ObjectName :: MethodName
    - 类的静态方法引用　　ClassName :: StaticMethodName
    - 类的非静态方法引用　　ClassName :: MethodName
    - 构造器的引用　　ClassName :: new
    - 数组的引用　　TypeName[] :: new
- 方法引用是在特定场景下lambda表达式的一种简化表示，可以进一步简化代码的编写使代码更加紧凑简洁，从而减少冗余代码。

22.1.5 Stream接口

- 案例题目：

  准备一个List集合并放入Person类型的对象，将集合中所有成年人过滤出来放到另外一个集合并打印出来。

（1）基本概念

- java.util.stream.Stream接口是对集合功能的增强，可以对集合元素进行复杂的查找、过滤、筛选等操作。
- Stream接口借助于Lambda 表达式极大的提高编程效率和程序可读性，同时它提供串行和并行两种模式进行汇聚操作，并发模式能够充分利用多核处理器的优势。

（2）使用步骤

- 创建Stream，通过一个数据源来获取一个流。
- 转换Stream，每次转换返回一个新的Stream对象。
- 对Stream进行聚合操作并产生结果。

（3）创建方式

- 方式一：通过调用集合的默认方法来获取流，如：default Stream stream()
- 方式二：通过数组工具类中的静态方法来获取流，如：static IntStream stream(int[] array)
- 方式三：通过Stream接口的静态方法来获取流，如：static Stream of(T... values)
- 方式四：通过Stream接口的静态方法来获取流，static Stream generate(Supplier<? extends T> s)

（4）中间操作

- 筛选与切片的常用方法如下：

方法声明	功能介绍
Stream filter(Predicate<? super T> predicate)	返回一个包含匹配元素的流
Stream distinct()	返回不包含重复元素的流
Stream limit(long maxSize)	返回不超过给定元素数量的流
Stream skip(long n)	返回丢弃前n个元素后的流

- 映射的常用方法如下：

方法声明	功能介绍
Stream map(Function<? super T,? extends R> mapper)	返回每个处理过元素组成的流
Stream flatMap(Function<? super T,? extends Stream<? extends R>> mapper)	返回每个被替换过元素组成的流，并将所有流合成一个流

- 排序的常用方法如下：

方法声明	功能介绍
Stream sorted()	返回经过自然排序后元素组成的流
Stream sorted(Comparator<? super T> comparator)	返回经过比较器排序后元素组成的流

（5）终止操作

- 匹配与查找的常用方法如下：

方法声明	功能介绍
Optional findFirst()	返回该流的第一个元素
boolean allMatch(Predicate<? super T> predicate)	返回所有元素是否匹配
boolean noneMatch(Predicate<? super T> predicate)	返回没有元素是否匹配
Optional max(Comparator<? super T> comparator)	根据比较器返回最大元素
Optional min(Comparator<? super T> comparator)	根据比较器返回最小元素
long count()	返回元素的个数
void forEach(Consumer<? super T> action)	对流中每个元素执行操作

- 规约的常用方法如下：

方法声明	功能介绍
Optional reduce(BinaryOperator accumulator)	返回结合后的元素值

- 收集的常用方法如下：

方法声明	功能介绍
<R,A> R collect(Collector<? super T,A,R> collector)	使用收集器对元素进行处理

22.1.6 Optional类

- 案例题目

  判断字符串是否为空，若不为空则打印字符串的长度，否则打印0。

（1）基本概念

- java.util.Optional类可以理解为一个简单的容器，其值可能是null或者不是null，代表一个值存在或不存在。
- 该类的引入很好的解决空指针异常，不用显式进行空值检测。

（2）常用的方法

方法声明	功能介绍
static Optional ofNullable(T value)	根据参数指定数值来得到Optional类型的对象
Optional map(Function<? super T,? extends U> mapper)	根据参数指定规则的结果来得到Optional类型的对象
T orElse(T other)	若该值存在就返回，否则返回other的数值。

22.2 Java9的新特性

22.2.1 Java9的概述

- Java9发布于2017年9月发布，带来了很多新特性，其中最主要的变化是模块化系统。
- 模块就是代码和数据的封装体，模块的代码被组织成多个包，每个包中包含Java类和接口，模块的数据则包括资源文件和其他静态信息。

22.2.2 模块化的使用

（1）语法格式

- 在 module-info.java 文件中，我们可以用新的关键词module来声明一个模块，具体如下：
  module 模块名称 {
  }

（2）模块化的优势

- 减少内存的开销。
- 可简化各种类库和大型应用的 开发和维护。
- 安全性，可维护性，提高性能。

22.2.3 钻石操作符的使用升级

- 在Java9中允许在匿名内部类的使用中使用钻石操作符。

22.2.4 集合工厂方法

（1）基本概念

- Java9的List、Set和Map集合中增加了静态工厂方法of实现不可变实例的创建。
- 不可变体现在无法添加、修改和删除它们的元素。
- 不允许添加null元素对象。

（2）实际意义

- 保证线程安全：在并发程序中既保证线程安全性，也大大增强了并发时的效率。
- 被不可信的类库使用时会很安全。
- 如果一个对象不需要支持修改操作，将会节省空间和时间的开销。
- 可以当作一个常量来对待，并且这个对象在以后也不会被改变。

22.2.5 InputStream的增强

- InputStream类中提供了transferTo方法实现将数据直接传输到OutputStream中。

22.3 Java10的新特性

22.3.1 Java10的概述

- Java10于2018年3月发布，改进的关键点包括一个本地类型推断、一个垃圾回收的增强。

- Java10计划只是一个短期版本，因此公开更新将在六个月内结束，9月份发布的Java11将是Java的长期支持（LTS）版本，LTS版本的发布每三年发布一次。

22.3.2 局部变量类型推断

（1）基本概念

- Java10可以使用var作为局部变量类型推断标识符，此符号仅适用于局部变量，增强for循环的索引，以及传统for循环的本地变量。
- 它不能使用于方法形式参数，构造函数形式参数，方法返回类型，字段，catch形式参数或任何其他类型的变量声明。

（2）实际意义

- 标识符var不是关键字，只是一个保留的类型名称。这意味着var用作变量，方法名或包名的代码不会受到影响，但var不能作为类或则接口的名字。
- 避免了信息冗余。
- 对齐了变量名。
- 更容易阅读。

22.4 Java11的新特性

22.4.1 Java11的概述

- Java11于2018年9月正式发布，这是 Java 大版本周期变化 后的第一个长期支持版本，非常值得关注。

22.4.2 简化的编译运行操作

- 在Java11中可以使用java命令一次性进行编译和运行操作。
- 执行源文件中的第一个类必须包含主方法。
- 不可以使用其它源文件中自定义的类。

22.4.3 String类新增方法

方法声明	功能介绍
boolean isBlank()	判断字符串是否为空或只包含空白代码点
Optional map(Function<? super T,? extends U> mapper)	根据参数指定规则的结果来得到Optional类型的对象
T orElse(T other)	若该值存在就返回，否则返回other的数值。