14.注解

讲师： 李刚

本章要点

- 注解的概念和作用。
- 5个基本注解
- JDK的元注解
- 自定义注解及其用法
- 开发APT工具

5个基本的Annotation

- @Override
- @Deprecated：Java 9改进了该注解，该注解支持since、forRemoval两个属性。
- @SuppressWarnings
- @SafeVarargs：Java 9增强了该注解，允许该注解修饰私有实例方法
- @FunctionalInterface

JDK的元注解

- 使用@Retention
- 使用@Target
- 使用@Documented
- 使用@Inherited

使用自定义注解

- 使用@interface定义注解
- 使用注解修饰程序中的类、方法、变量、接口等定义，通常我们会把注解放在所有修饰符之前。
- 定义带成员变量的注解。
- 为注解的成员变量指定初始值。

提取注解信息

- Annotation接口来代表程序元素前面的注释，该接口是所有Annotation类型的父接口。
- AnnotatedElement接口代表程序中可以接受注释的程序元素。
- 调用AnnotatedElement对象的如下三个方法来访问Annotation信息：
  - getAnnotation(Class<T> annotationClass)：返回该程序元素上存在的、指定类型的注释，如果该类型的注释不存在，则返回null。
  - Annotation[] getAnnotations()：返回该程序元素上存在的所有注释。
  - boolean isAnnotationPresent(Class<? extends Annotation> annotationClass)：判断该程序元素上是否包含指定类型的注释，存在则返回true，否则返回false。

Java 8新增的重复注解

- 在Java 8以前，同一个程序元素前最多只能使用一个相同类型的注解；如果需要在同一个元素前使用多个相同类型的注解，则必须使用注解“容器”。
- 为了将该注解改造成重复注解，需要使用@Repeatable修饰该注解，使用@Repeatable时必须为value成员变量指定值。

Java 8新增的类型注解

- Java 8为ElementType枚举增加了TYPE_PARAMETER、TYPE_USE两个枚举值，这样就允许定义枚举时使用@Target(ElementType.TYPE_USE)修饰，这种注解被称为类型注解（Type Annotation），类型注解可用在任何用到类型的地方。
- 从Java 8开始，类型注解可以在任何用到类型的地方使用。

APT简介

- APT（annotation processing tool）是一种处理注释的工具，它对源代码文件进行检测找出其中的类型注解后，对类型注解进行额外的处理。
- 类型注解处理器在处理类型注解时可以根据源文件中的类型注解生成额外的源文件和其它的文件（文件具体内容由类型注解处理器的编写者决定），APT还会编译生成的源代码文件和原来的源文件，将它们一起生成class文件。

开发用户自定义APT

- 为了使用系统的APT工具来读取源文件中的类型注解，程序员必须自定义一个类型注解处理器，编写Annotation处理器需要使用JDK lib目录中的tools.jar里的如下4个包：
  - com.sun.mirror.apt：和APT交互的接口。
  - com.sun.mirror.declaration：包含各种封装类成员、类方法、类声明的接口。
  - com.sun.mirror.type：包含各种封装源代码中程序元素的接口。
  - com.sun.mirror.util：提供了用于处理类型和声明的一些工具。