注解系列知识总结（一）

栏目:Java基础 作者:admin 日期:2016-07-22 评论:1 点击:5,399 次

第一节：注解的作用

Annotation(注解)是JDK5.0及以后版本引入的。它的作用是修饰编程元素。什么是编程元素呢？例如：包、类、构造方法、方法、成员变量等。

第二节：注解的作者

注解是JDK 5.0推出来的，这部分JDK源码的作者是：Joshua Bloch，这是一个技术大牛。Collections Framework皆是他一手打造，还获得了当年的Jolt大奖。另外，他还写了一本非常出名的书：《Effective Java》，相信大家不会陌生吧。如果没有读过这本书，那到这里下载下来看看吧：http://swiftlet.net/archives/495。关于更多Joshua Bloch的故事，后期金丝燕网将会单独撰文介绍，这里不再赘述了。

第三节：注解的语法与定义形式

（1）以@interface关键字定义
（2）注解包含成员，成员以无参数的方法的形式被声明。其方法名和返回值定义了该成员的名字和类型。
（3）成员赋值是通过@Annotation(name=value)的形式。
（4）注解需要标明注解的生命周期，注解的修饰目标等信息，这些信息是通过元注解实现。
上面的语法不容易理解，下面通过例子来说明一下，这个例子就是Target注解的源码，

```
1 @Retention(value = RetentionPolicy.RUNTIME)
2 @Target(value = { ElementType.ANNOTATION_TYPE } )
3 public @interface Target
4 {
5     ElementType[] value();
6 }
```

源码分析如下：
第一：元注解@Retention，成员value的值为RetentionPolicy.RUNTIME。
第二：元注解@Target，成员value是个数组，用{}形式赋值，值为ElementType.ANNOTATION_TYPE
第三：成员名称为value，类型为ElementType[]
另外，需要注意一下，如果成员名称是value，在赋值过程中可以简写。如果成员类型为数组，但是只赋值一个元素，则也可以简写。如上面的简写形式为：
@Retention(RetentionPolicy.RUNTIME)
@Target(ElementType.ANNOTATION_TYPE)

第四节：注解的分类

注解分为两类，一类是元注解，另外一类是普通注解。
所谓元注解就是修饰注解的注解。
拿到一个注解，如何知道它是否是元注解呢？需要看它的元注解（无论是元注解还是普通注解都是有元注解的），如果看到这样的元注解：@Target(ElementType.ANNOTATION_TYPE)，那么此时这个注解一定是元注解。

第五节：注解的生命周期

注解的定义语法中已经说到了：注解需要标明注解的生命周期，这些信息是通过元注解实现。而这个元注解是：

```
1 public @interface Retention
2 {
3     RetentionPolicy value();
4 }
```

Retention注解的值是enum类型的RetentionPolicy。包括如下几种情况：
（1）SOURCE: 注解只保留在源文件，当Java文件编译成class文件的时候，注解被遗弃。Annotations are to be discarded by the compiler.
（2）CLASS: 注解被保留到class文件，jvm加载class文件时候被遗弃。这是默认的生命周期。
Annotations are to be recorded in the class file by the compiler
but need not be retained by the VM at run time. This is the default
behavior.
（3）RUNTIME: 注解不仅被保存到class文件中，jvm加载class文件之后，仍然存在，保存到class对象中，可以通过反射来获取。
Annotations are to be recorded in the class file by the compiler and
retained by the VM at run time, so they may be read reflectively.

第六节：注解的修饰目标

最新文章 精品系列

在你的眼里，JDK中那个API是最...
《Java高手的境界》公开课公告
关于写工具的两点原则
华为年轻化，39岁被嫌太老，被赶走！
jsonstar简介
工作心法
学而不思则罔，思而不学则殆
第三节：源码阅读的方法
第二节：正确的读书方法
第一节：夯实基础

注解的定义语法中已经说到了：注解需要标明注解的修饰目标，这些信息是通过元注解实现。而这个元注解是：

```
public @interface Target
{
    ElementType[] value();
}
```

这个注解的值是enum类型ElementType。包括以下几种情况：

（1）TYPE:指的是在类，接口（包括注解）或者enum上使用的注解。

（2）FIELD:指的在field属性，也包括enum常量使用的注解。

（3）METHOD:指的是在方法声明上使用的注解。

（4）PARAMETER:指的是在参数上使用的注解，

（5）CONSTRUCTOR: 指的是在构造器使用的注解。

（6）LOCAL_VARIABLE:指的是在局部变量上使用的注解。

（7）ANNOTATION_TYPE:指的是在注解上使用的元注解

（8）PACKAGE:指的是在包上使用的注解。

第七节：注解的底层实现

定义一个注解：

```
@Retention(RetentionPolicy.RUNTIME)
@Target(ElementType.TYPE)
public @interface Cache
{
    String value() default "cache";
}
```

分析其字节码，如下图所示：

C:\Users\Administrator\workspace\cache\bin>javap -verbose Cache

```
Classfile /C:/Users/Administrator/workspace/cache/bin/Cache.class
  Last modified 2016-7-23; size 429 bytes
  MD5 checksum 771f9c4d63ce2ded5d3b093deebd2c69
  Compiled from "Cache.java"
public interface Cache extends java.lang.annotation.Annotation
  minor version: 0
  major version: 52
  flags: ACC_PUBLIC, ACC_INTERFACE, ACC_ABSTRACT, ACC_ANNOTATION
Constant pool:
   #1 = Class              #2             // Cache
   #2 = Utf8               Cache
   #3 = Class              #4             // java/lang/Object
   #4 = Utf8               java/lang/Object
   #5 = Class              #6             // java/lang/annotation/Annotation
   #6 = Utf8               java/lang/annotation/Annotation
   #7 = Utf8               value
   #8 = Utf8               ()Ljava/lang/String;
   #9 = Utf8               AnnotationDefault
  #10 = Utf8               cache
  #11 = Utf8               SourceFile
  #12 = Utf8               Cache.java
  #13 = Utf8               RuntimeVisibleAnnotations
  #14 = Utf8               Ljava/lang/annotation/Retention;
  #15 = Utf8               Ljava/lang/annotation/RetentionPolicy;
  #16 = Utf8               RUNTIME
  #17 = Utf8               Ljava/lang/annotation/Target;
  #18 = Utf8               Ljava/lang/annotation/ElementType;
  #19 = Utf8               TYPE
{
  public abstract java.lang.String value();
    descriptor: ()Ljava/lang/String;
    flags: ACC_PUBLIC, ACC_ABSTRACT
    AnnotationDefault:
    default_value: s#10
}
SourceFile: "Cache.java"
RuntimeVisibleAnnotations:
  0: #14(#7=e#15.#16)
  1: #17(#7=[e#18.#19])
```

分析上面的字节码，我们可以得出：

第一：

public interface Cache extends java.lang.annotation.Annotation，说明Cache注解是继承自Annotation，仍然是interface。

第二：

public abstract java.lang.String value()，说明value方法是abstract类型。

上一篇：2016年7月16日20点网络公开课《面向接口编程思想》内容
下一篇：《通向名企之路》

注解系列知识总结（一）：目前有1 条留言

mno: 沙发

不错

2016-09-29 下午4:58 [回复]

发表评论

昵称 *

邮箱 *

网址

提交[Ctrl+Enter] 重写