Java实现自己的Annotation

作者：wanglei0622
原文链接：https://ld246.com/article/1461740868194
来源网站：链滴

<h2>Annotation释义</h2>
Annotation：注解，顾名思义就是一个说明解释，在项目中的作用就是取代繁琐的配置文件，接在代码上标注。
标注可以加在包，类，字段，方法，方法的参数以及局部变量上。
<h2>实现方式</h2>
Java中提供了四种元注解，专门负责注解其他的注解，分别如下

@Retention元注解，表示需要在什么级别保存该注释信息（生命周期）。可选的RetentionPoicy数包括：
RetentionPolicy.SOURCE: 停留在java源文件，编译器被丢掉
RetentionPolicy CLASS：停留在class文件中，但会被VM丢弃（默认）
RetentionPolicy.RUNTIME：内存中字节码，VM将在运行时也保留注解，因此可以通过反射机制读取注解的信息

@Target元注解，默认值为任何元素，表示该注解用于什么地方。可用的ElementType参包括
ElementType.CONSTRUCTOR: 构造器声明
ElementType.FIELD: 成员变量、对、属性（包括enum实例）
ElementType.LOCAL_VARIABLE: 局部变量声明
ElementTye.METHOD: 方法声明
ElementType.PACKAGE: 包声明
ElementType.PARAMETER: 数声明
ElementType.TYPE: 类、接口（包括注解类型)或enum声明

Documented将注解包含在JavaDoc中

@Inheried允子类继承父类中的注解
<h2>代码示例</h2>
<pre class="brush: java">/**
 * 注解类型@interface,必须把一个类型定义为@interface，而不能用class或interface关键字。
 * 所有的注解类都隐式继承于java.lang.annotation.Annotation，注解不允许显式继承于其他的接口

 * @author wanglei
 */
@Target(ElementType.METHOD)
@Retention(RetentionPolicy.RUNTIME)
@Documented
@Inherited
public @interface MyAnnotation {
    //为注解添加属性
      String path();
      String type() default "get"; //为属性提供默认值
      int[] array() default {1, 2, 3};
}

/**

- 注解测试类
- @author wanglei
-

*/

public class AnnotationTest {

@MyAnnotation(path="test/add",type="post")

public void test(){

}
//通过反射获得注解属性
public static void main(String[] args) {
    try {

```
        if(AnnotationTest.class.getMethod("test", null).isAnnotationPresent(MyAnnotation.class))

            MyAnnotation annotation=(MyAnnotation)AnnotationTest.class.getMethod("test", nul
).getAnnotation(MyAnnotation.class);
            System.out.println(annotation);
            System.out.println(annotation.path());
            System.out.println(annotation.type());
        }
    } catch (NoSuchMethodException e) {
        e.printStackTrace();
    } catch (SecurityException e) {
        e.printStackTrace();
    }
}

}</pre>

<p> </p>
```

原文链接：Java实现自己的Annotation