简述

因为平常开发少见，相信有不少的人员会认为注解的地位不高。其实同 classs 和 interface 一样，注解也属于一种类型。它的概念

注解的定义

注解通过 `@interface` 关键字进行定义。

```
public @interface TestAnnotation {
}
```

它的形式跟接口很类似，不过前面多了一个 @ 符号

元注解

元注解是什么意思呢？元注解是可以注解到注解上的注解，或者说元注解是一种基本注解，但是它能够应用到其它的注解上

元标签有 @Retention、@Documented、@Target、@Inherited、@Repeatable 5 种

@Retention

Retention 的英文意为保留期的意思。当 @Retention 应用到一个注解上的时候，它解释说明了这个注解的的存活时间。
它的取值如下：
- RetentionPolicy.SOURCE 注解只在源码阶段保留，在编译器进行编译时它将被丢弃忽视。
- RetentionPolicy.CLASS 注解只被保留到编译进行的时候，它并不会被加载到 JVM 中。
- RetentionPolicy.RUNTIME 注解可以保留到程序运行的时候，它会被加载进入到 JVM 中，所以在程序运行时可以获取到它

@Documented

顾名思义，这个元注解肯定是和文档有关。它的作用是能够将注解中的元素包含到 Javadoc 中去

@Target

Target 是目标的意思，@Target 指定了注解运用的地方。
你可以这样理解，当一个注解被 @Target 注解时，这个注解就被限定了运用的场景。
类比到标签，原本标签是你想张贴到哪个地方就到哪个地方，但是因为 @Target 的存在，它张贴的地方就非常具体了，比如参数上等等。@Target 有下面的取值

- ElementType.ANNOTATION_TYPE 可以给一个注解进行注解
- ElementType.CONSTRUCTOR 可以给构造方法进行注解
- ElementType.FIELD 可以给属性进行注解
- ElementType.LOCAL_VARIABLE 可以给局部变量进行注解
- ElementType.METHOD 可以给方法进行注解
- ElementType.PACKAGE 可以给一个包进行注解
- ElementType.PARAMETER 可以给一个方法内的参数进行注解
- ElementType.TYPE 可以给一个类型进行注解，比如类、接口、枚举

@Inherited

Inherited 是继承的意思，但是它并不是说注解本身可以继承，而是说如果一个超类被 @Inherited 注解过的注解进行注解的何注解应用的话，那么这个子类就继承了超类的注解

@Repeatable

Repeatable 自然是可重复的意思。@Repeatable 是 Java 1.8 才加进来的，所以算是一个新的特性。什么样的注解会多次应取多个。举个例子，一个人他既是程序员又是产品经理,同时他还是个画家。

```
1
2  @interface Persons {
3      Person[]  value();
4  }
5
6  @Repeatable(Persons.class)
7  @interface Person{
8      String role default "";
9  }
10
11 @Person(role="artist")
12 @Person(role="coder")
13 @Person(role="PM")
14 public class SuperMan{
15 }
```

注意上面的代码，@Repeatable 注解了 Person。而 @Repeatable 后面括号中的类相当于一个容器注解