https://blog.csdn.net/asdgbc/article/details/70196749

@Retention注解：

SOURCE   
被编译器忽略

CLASS   
注解将会被保留在Class文件中，但在运行时并不会被VM保留。这是默认行为，所有没有用Retention注解的注解，都会采用这种策略。

RUNTIME   
保留至运行时。所以我们可以通过反射去获取注解信息。

[Annotation实战【自定义AbstractProcessor】](https://www.cnblogs.com/avenwu/p/4173899.html)

<https://www.cnblogs.com/avenwu/p/4173899.html>

javax.annotation.processing.AbstractProcessor：

javax.lang.model.element.TypeElement:

代表一个程序元素，例如包、类或者方法。每一个元素代表一个静态的、语言层的构件（并且不是一个虚拟机的运行时构件）。

元素的比较应该使用equals方法。不能保证一个特定的元素总能被同样的对象代表。

要实现基于元素对象的类的操作，要么使用一个visitor，要么使用getKind()方法的返回值。在此建模层中使用instanceof决定一个对象的有效类不是可靠的语法，因为一个实现可能选择让一个对象实现多个元素子接口。

javax.lang.model.element.Element:

TypeElement 代表一个类或接口程序元素，它提供了访问与其类型和成员相关的信息的方法。注意，枚举类型是一类class，而注解类型是一类interface。

当TypeElement代表一个类或接口元素，DeclaredType代表类或接口类型，后者成为前者的（use）使用（或调用）。两者的区别在泛型上是最明显的，对于泛型，单独的元素可以定义整个类型集。例如元素java.util.Set对应参数化类型java.util.Set、java.util.Set(以及许多其他类型)和根类型java.util.Set.

接口中每一个返回元素列表的的方法，都将按照程序信息的底层源码的自然顺序返回列表。例如，如果信息的底层源码是java源代码，这些元素则按源代码的顺序返回。