# 注解（Annotation）

## 一、概述

注解（Annotation）相当于一种标记，注解本身是不会做任何事，但是根据注解的类型，编译器会对注解进行处理，例如运行时类型的注解（runtime）会被写入编译文件中，并被加载到虚拟机。

JDK1.5之后，JDK内部提供了@Override(重写)、@Deprecated(废弃)等几个注解，同时为了利用好注解，为开发者提供自定义注解方法。

## 二、几个基本概念

1. 注解本质是注解接口类，隐式的扩展自java.lang.annotation.Annotation接口。
2. 使用在注解上的注解被称为元注解，jdk提供了四种元注解在自定义注解时使用：@Documented,@Retention,@Target,@Inherited。

## 三、自定义注解实现及用法

### 基本语法

modifers @interface AnnotationName{

type elementName();

type elementName() default value;

}

Modifers：上文所说，自定义注解本质是注解接口类，所以modifers只是public。

Type：注解元素类型必须在编写时确定，所以不能是泛型指代也不能是Object，支持的类型可以是八种基本类型，String，Class，enum，注解类型，上述类型组成的数组。

Default：在实际应用中一位内一个注解元素会导致许多不便，所以直接规范中一个注解元素永远不能为null，所以就需要使用defautl赋予一个默认值。

### 简单实例及说明

@Documented

@Retention(RetentionPolicy.***RUNTIME***)

@Target({ElementType.***METHOD***,ElementType.***FIELD***,ElementType.***TYPE***,ElementType.***PARAMETER***})

@Inherited

**public** **@interface** MyAnnotation {

String value() **default** "a";

}

--------------------------------------------

#### 2.1 四个元注解说明

@Documented是可选元注解，使用时javadoc生成文档中该注解生成到文档中，例附文图1

@Retention必须标注，指定一条注解保留多长时间。参数值有：

RetentionPolicy.RUNTIME(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.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)

其中SOURCE和CLASS说明，两者都不会被加载到JVM中，但如果在使用两者无影响的情况下是推荐使用CLASS的

@Target~~必须标注~~ 一条没有Target限制的注解可以应用于任何项上，限制注解可以使用在什么地方，参数值有：

ANNOTATION\_TYPE：注解类型声明；

PACKAGE 包；

TYPE 类/接口/注解/enum；

METHOD 方法

CONSTRUCTOR 构造器；

FIELD 成员域；

PARAMETER 方法/构造器参数；

LOCAL\_PARAMETER 局部变量

TYPE\_PARAMETER 类型参数

TYPE\_USE 类型用法

@Inherited 可选，只能应用于可对类注解的注解，表示继承了拥有该注解的超类时，子类自动拥有该注解

#### 特殊情况说明

注解元素的名称被命名为value时，使用注解时不用严格按照赋值规范 @MyAnnotation(value=”c”) 可以直接省略为 @MyAnnotation("c")，如附文图2 ；其他情况，如果这么赋值会直接无法通过编译。

## 四、附文

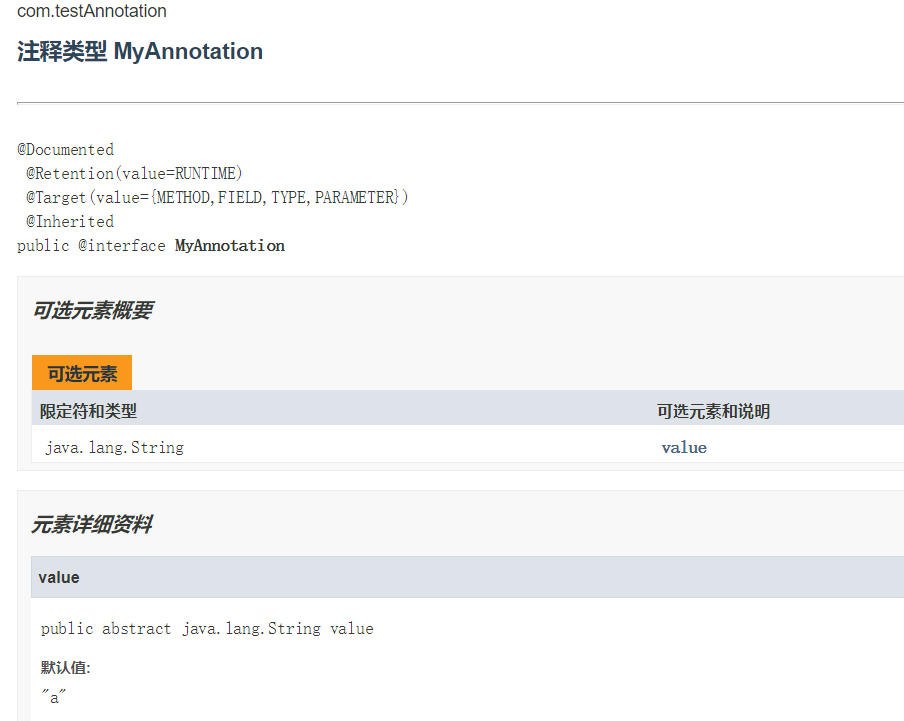


图 1

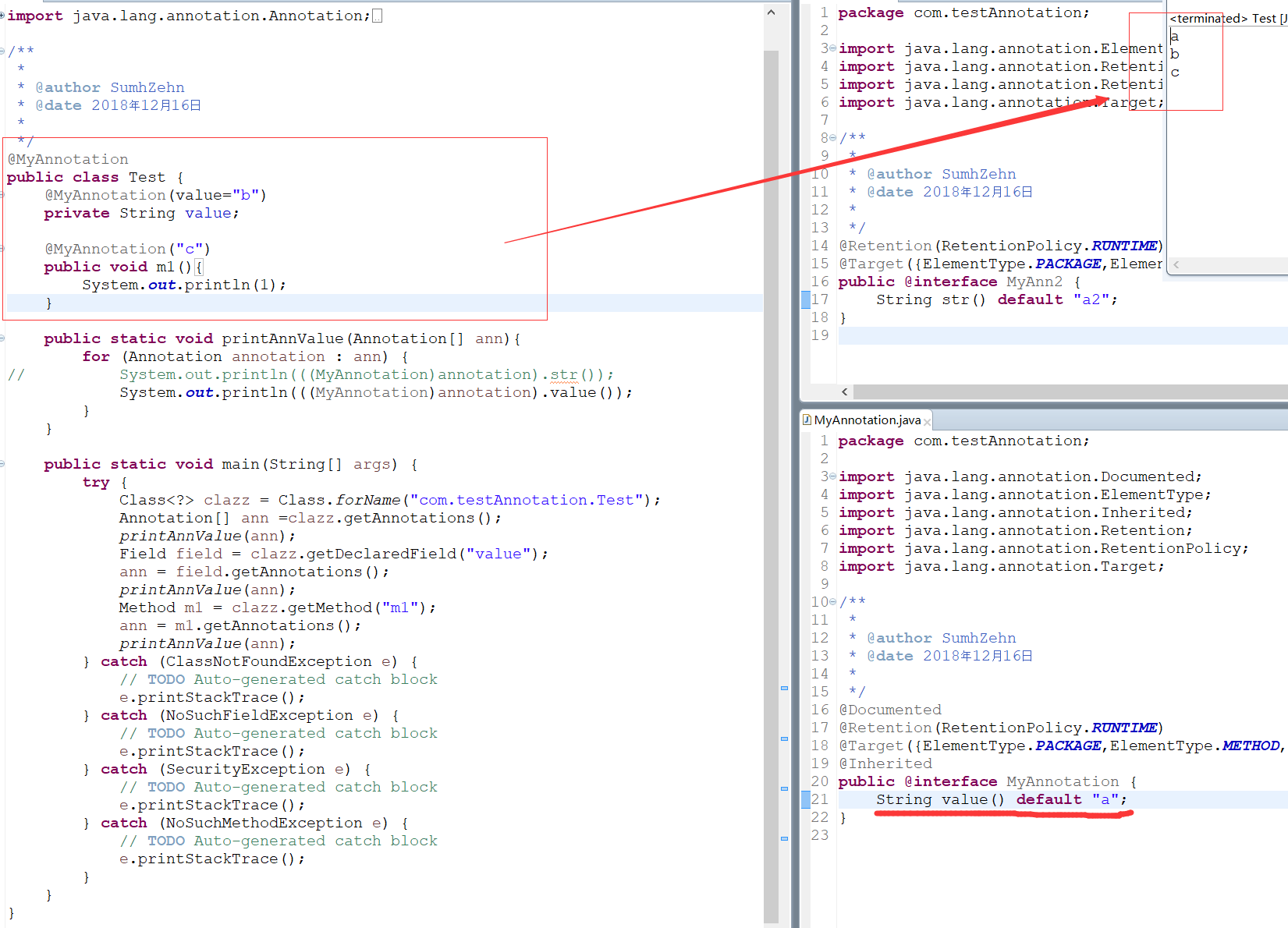


图 2