自定义注解

先来了解元注解

元注解

1、元注解是用于定义注解的注解

2、取值

@Target：描述注解能够作用的位置

属性ElementType的主要取值：（还有其他范围）

- TYPE：类、接口（包括注解类型）或枚举声明
- METHOD：可以作用于方法上
- FIELD：可以作用于成员变量上

```
@Target({ElementType.METHOD})
public @interface Ety {
        String value();
        String name();
}
```

@Documented：

表示拥有该注解的元素可通过javadoc此类的工具进行文档化。

```
@Documented
public @interface Ety {
    String value();
    String name();
}
```

@Inherited：描述注解是否能被子类继承

```
@Inherited
public @interface Ety {
    String value();
    String name();
}
```

@Retention：表示该注解类型的注解保留的时长。

当注解类型声明中没有@Retention元注解，则默认保留策略为RetentionPolicy.CLASS。

关于保留策略(RetentionPolicy)是枚举类型，共定义3种保留方式：

- **SOURCE　仅存在Java源文件，经过编译器后便丢弃相应的注解**
- **CLASS　　存在Java源文件，以及经编译器后生成的Class字节码文件，但在运行时VM不再保留注释**
- **RUNTIME 存在源文件、编译生成的Class字节码文件，以及保留在运行时VM中，可通过反射性地读取注解**

```
@Retention(RetentionPolicy.RUNTIME)
public @interface Ety {
    @Deprecated
    String value();
    @Deprecated
    String name();
}
```

自定义注解

一、基本介绍

1、格式：

元注解
public @interface 注解名称{
    属性列表;
}

注解本质上就是一个接口，该接口默认继承Annotation接口 创建MyAnno注解先编译然后再使用反编译，如下

2、返回值类型

- **.基本数据类型**
- **String类型**
- **枚举**
- **注解　　不要带`@`**
- **以上类型的数组**

注意：属性名称（即方法名称）最好定义成属性名的形式，因为使用的时候是`方法名

```
public @interface Anno {
        int age();//注意：不能带参数age(int x)不行
        String name();
        Anno2 anno();//注解
        Animal dog();//枚举  DOG,CAT;
        int[] arrs();
}
```

3、使用细节

1）可以设置默认值

```
public @interface MyAnno {
    int age() default 18;
    String name() default "dog";
}
```

2）单一参数：如果自定义注解 只有一个String value()方法

```
public @interface MyAnno {
    String value();
}
```

可以不需要指定value="hello"

```
//@MyAnno(value="hello")
@MyAnno("hello")// 不需要指定value="hello"
```

```
public class ItemDao {
    }
```

二、实例

自定义注解的语法要求：

```
@Target({ElementType.METHOD,ElementType.TYPE})
@Retention(RetentionPolicy.RUNTIME)
@Inherited
@Documented
public @interface Description {
    String desc();
    String author();
    int age() default 18;
}
```