# 注解

## 简述：

* 注解它不是**注释，**注释是程序员写的，给程序员看的。
* 注解**给程序看**，用于描述程序如何运行及在什么阶段来运行。
* 注解在实际开发中，**最大的功能是用于替换配置文件**。
* 注解是jdk1.5的新特性
* **可以通过反射来让注解具有功能。**
* 注解的格式： @xxxx
* **注解它的本质就是一个接口，这个接口需要继承 Annotation接口**

## 三个基本的注解

@Override：检查子类确实是覆盖/重写了父类的方法。

@Deprecated：说明已经过时了。

@SuppressWarnings({ "unused", "deprecation" })：抑制程序中的警告。unused警告的类型。{}数组。all抑制所有警告

|  |
| --- |
|  |

## 如何声明注解

Step1:声明一个注解 **@interface 注解名｛｝**

Step2:声明注解中的成员(注解的属性类型可以有哪些?)

1.基本类型

2.String

3.枚举类型

4.注解类型

5.Class类型

6.以上类型的一维数组类型

Step3:注解使用

|  |
| --- |
| 声明注解 |
|  |
| 使用注解 |
|  |

## 模拟Junit中的@Test

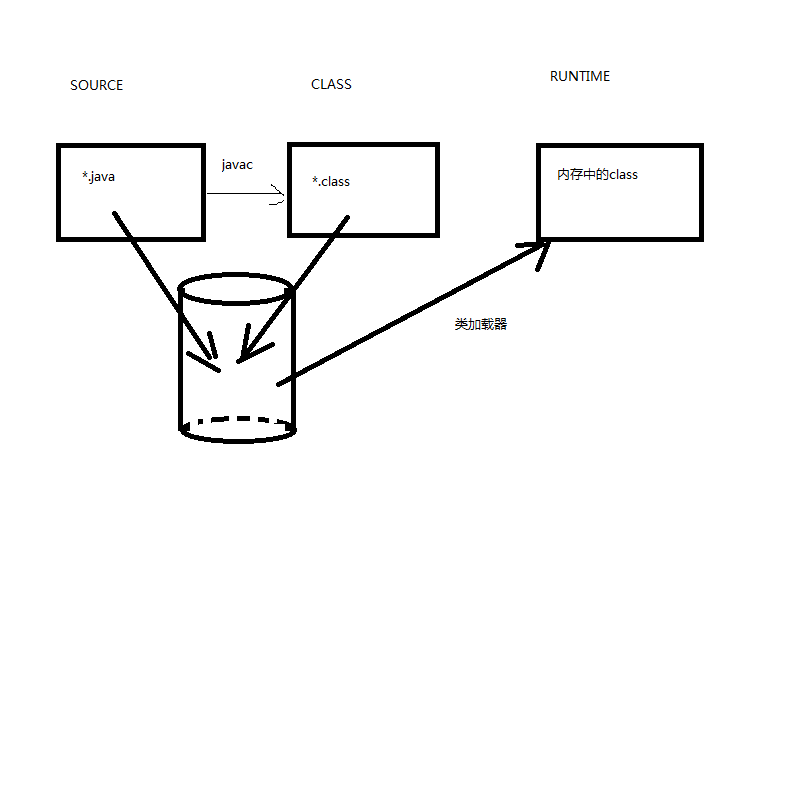
|  |
| --- |
| Step1:写个MyTest.java注解 |
|  |
| Step2:写个测试类Demo01.java |
|  |
| Step3:写个TestRunner |
|  |

## 元注解

什么是元注解：

只能用在注解上的注解叫做元注解。（即：用于修饰注解的注解）

a、自定义的注解的存活范围（生命周期）：默认是CLASS。



* @Retention：作用。改变自定义的注解的存活范围。

RetentionPolicy:

SOURCE

CLASS

RUNTIME

* @Target:作用，指定该注解能用在什么地方。

ElementType：

TYPE：

METHOD：

FIELD：

ANNOTATION\_TYPE

@Documented：作用，使用了@MyTest的注解的类，如果@MyTest注解上面有@Documented注解，那么使用了@MyTest的注解的类的API文档中会出现@MyTest的身影。

@Inherited：作用，说明该注解可以被继承下去。

## Servlet3.0中的2个常用注解

### WebFilter

|  |
| --- |
|  |

### WebServlet

|  |
| --- |
|  |