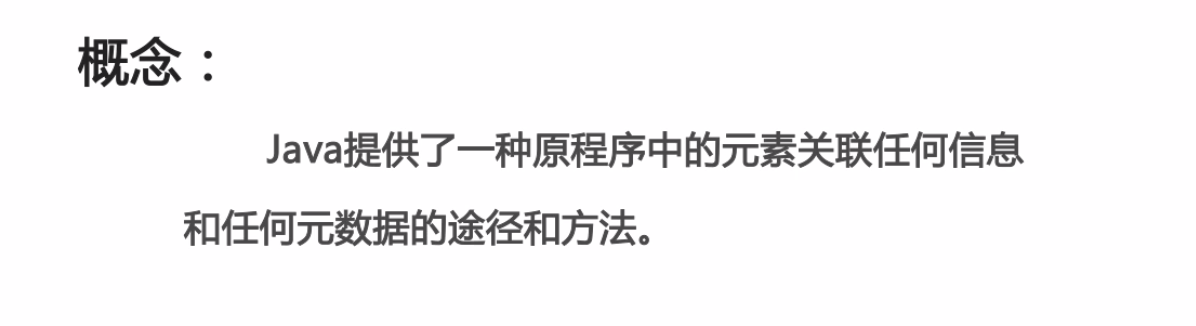
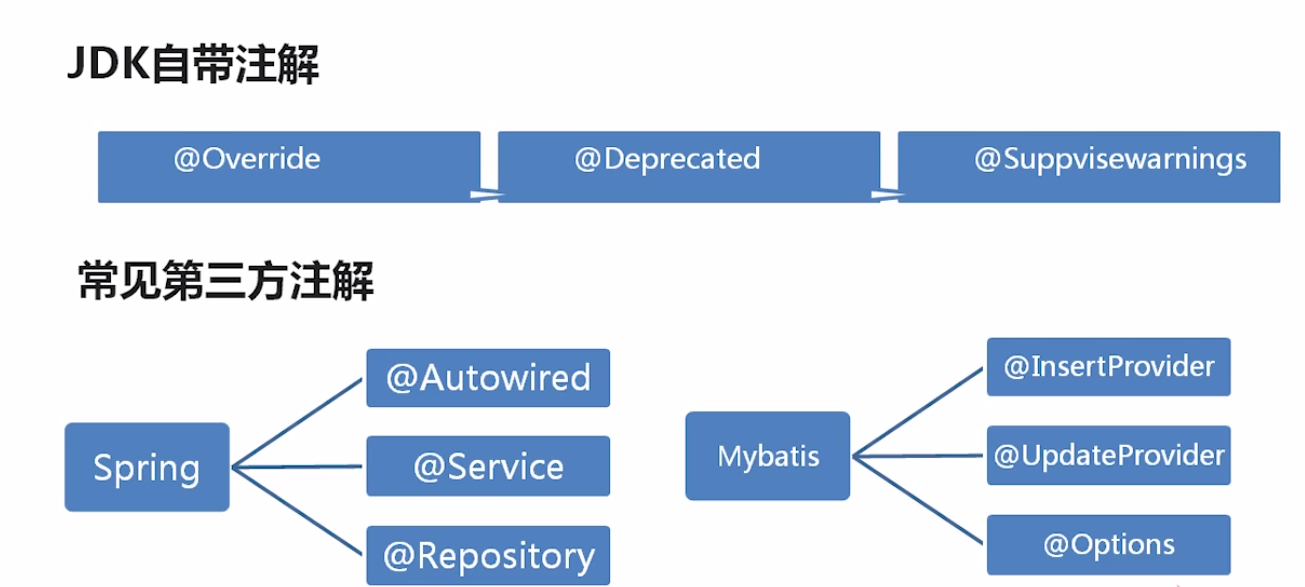
# 概念



# 分类



/\*\*

\* Override覆写注解

\*/

@Override

/\*\*

\* Deprecated方法过时注解

\*/

@Deprecated

忽略警告注解

@SuppressWarnings("deprecation")

## 按运行机制分：

源码注解

注解只在源码中存在，编译成class文件后就不存在了

编译注解

注解在源文件和.class文件中都存在。如Override等

运行注解

运行时还存在或能起作用，影响程序运行逻辑。

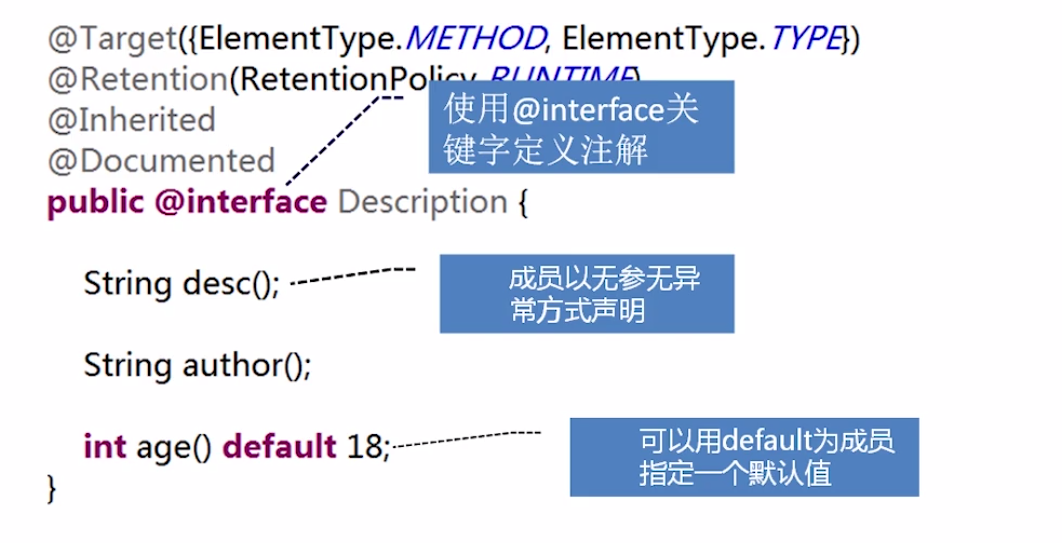
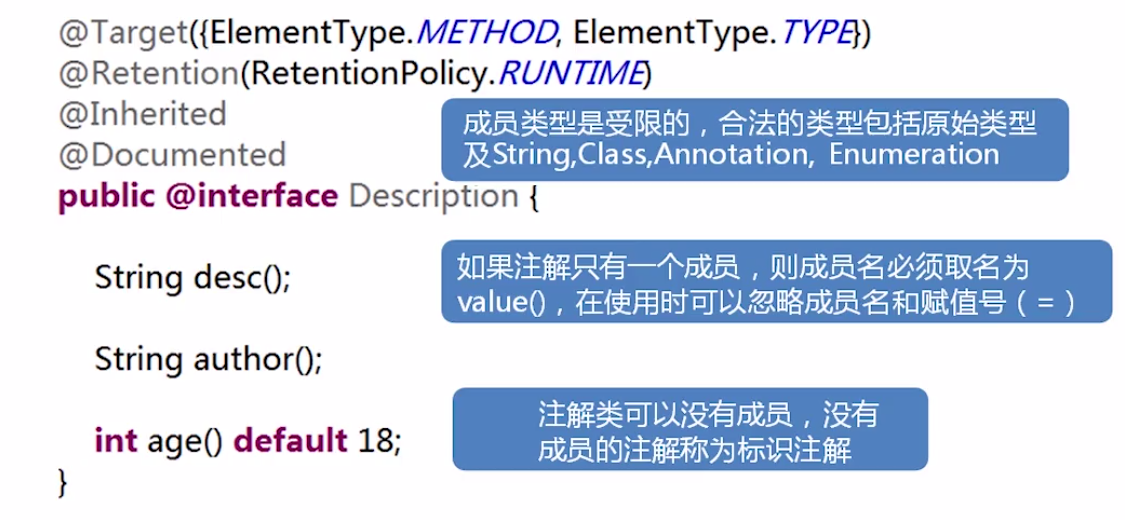
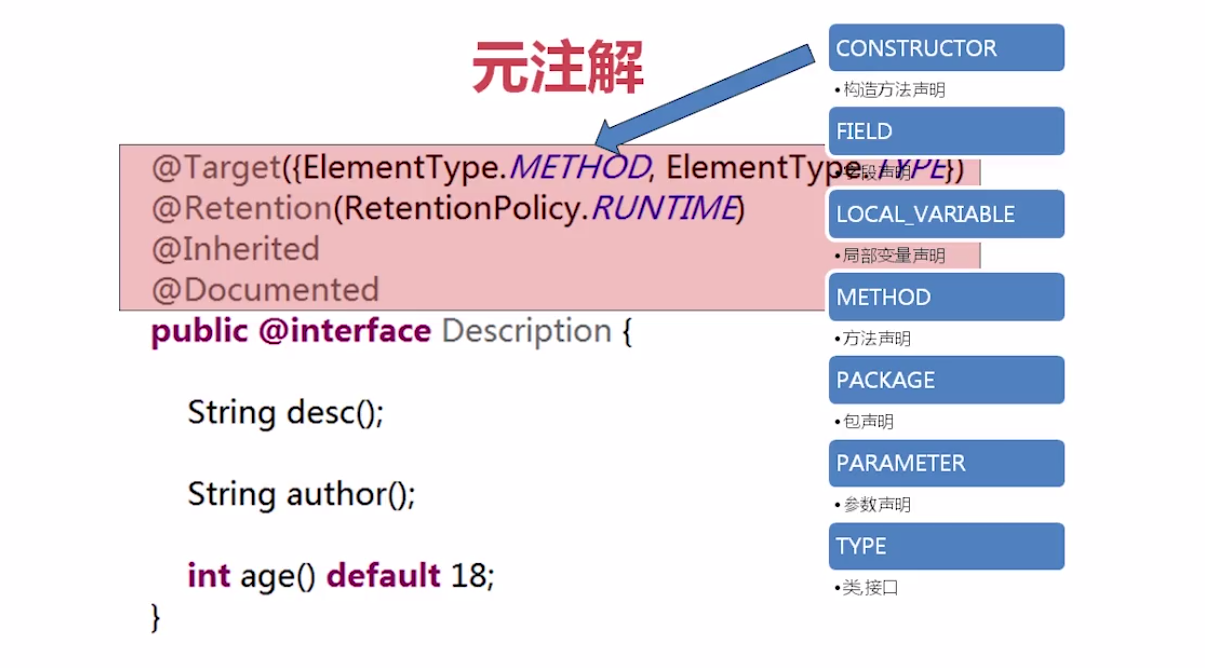
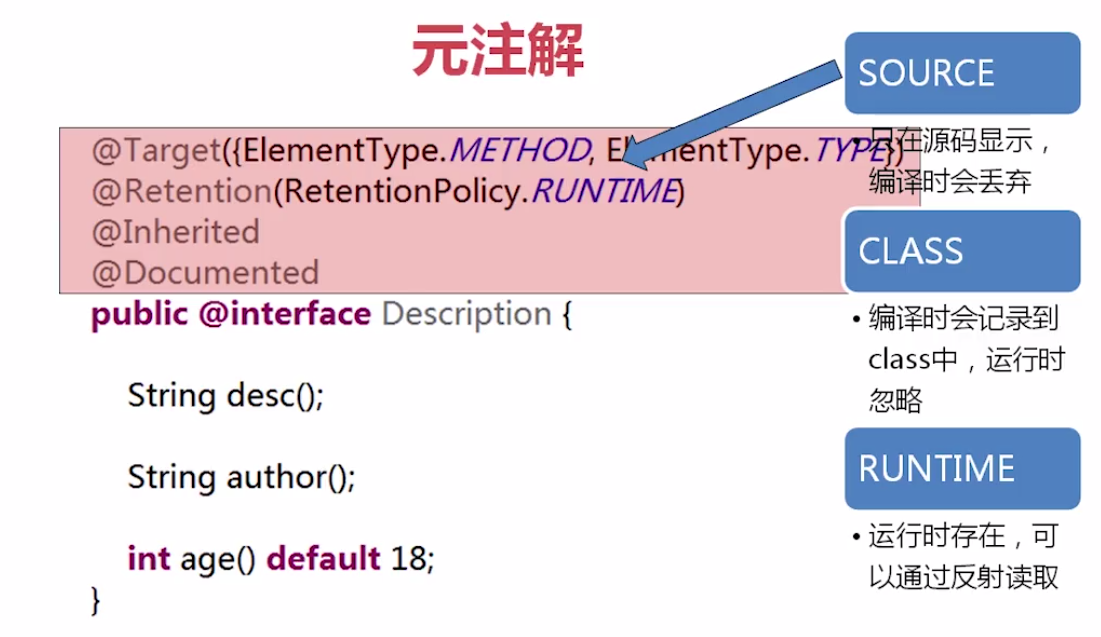
## 按注解来源分：

来自JDK

来自第三方

自己定义的

元注解（注解的注解）





# 自定义注解

**import** java.lang.annotation.Documented;

**import** java.lang.annotation.ElementType;

**import** java.lang.annotation.Inherited;

**import** java.lang.annotation.Retention;

**import** java.lang.annotation.RetentionPolicy;

**import** java.lang.annotation.Target;

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

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

@Inherited

@Documented

**public** **@interface** Description {

String desc();

String author();

**int** age() **default** 18;

}

# 自定义注解的使用

@Description(author = "duan", desc = "any",age = 24)

**public** **void** getAge()

{

# 解析注解

通过反射获取类、函数或成员上的运行时注解信息，从而实现动态控制程序运行的逻辑。

**import** java.lang.reflect.Method;

**import** java.lang.annotation.Annotation;

**public** **class** **ParseAnn** {

**public** **static** **void** **main**(**String**[] args) {

//使用类加载器加载类

**try** {

**Class** **c** = **Class**.*forName*("test.Child");

//找到类上面的注解

**boolean** **isExist** = c.isAnnotationPresent(***Description***.**class**);

**if** (isExist)

{

//拿到注解实例

***Description*** **des** = (***Description***)c.getAnnotation(***Description***.**class**);

**System**.***out***.println(des.value());

}

//找到方法上的注解

**Method**[] **me** = c.getMethods();

**for**(**Method** **m**: me)

{

**if** (m.isAnnotationPresent(***Description***.**class**))

{

***Description*** **des** = (***Description***)m.getAnnotation(***Description***.**class**);

**System**.***out***.println(des.value());

}

}

//另一种解析方法上注解的方法

**for** (**Method** **m** : me)

{

Annotation[] **an** = m.getAnnotations();

**for** (Annotation **a** : an)

**System**.***out***.println(a.toString());

}

} **catch** (**ClassNotFoundException** **e**) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}