注解：annotation（标识，标签），从Java 5开始支持注解。

注解是贴在程序元素上面。

程序元素：类、方法、字段、方法参数、接口、构造函数、枚举。

在反射的时候，动态可以获取字节码，方法，字段等等程序元素，获取了这些程序元素，那么就能获取程序元素上贴的注解。这些注解会参与程序运行提供比较的相关信息和数据。

注解是一个特殊的接口，所有的注解都继承自java.lang.annotation这个接口。

完整的注解（从编写到最终运行）需要三方面的参与。

1. 定义一个注解
2. 被贴的程序元素（类、方法、字段、构造器等）
3. 第三方程序的支持（赋予我注解的特殊功能）



#### 1、JDK中内置的注解

@Override 限定覆写父类方法

@Deprecated 标记已过时的成员

#### 2、JDK中的元注解

注解：贴在程序元素上面的标签。

元注解：注解的注解（贴在注解上面的注解）

元注解主要用于限定当前的注解能够贴在哪儿？能够保留在哪个阶段（程序执行三个阶段源代码阶段，字节码阶段，JVM中）

@Retention：表示注解可以保存在哪一个时期。

@Documented

@Retention(RetentionPolicy.*RUNTIME*)  
@Target(ElementType.*ANNOTATION\_TYPE*)  
public @interface Retention {  
 RetentionPolicy value();  
}

保存的时期的值，封装在RetentionPolicy枚举类中

|  |  |
| --- | --- |
| 枚举常量摘要 | |
| RetentionPolicy.CLASS | 注解保留到字节码阶段，运行时候失效了 |
| RetentionPolicy.RUNTIME | 注解保留到运行阶段，运行时候使用反射获取做相应的程序处理（一般开发者自定注解都保留运行阶段） |
| RetentionPolicy.SOURCE | 注解在源代码阶段有效，编译字节码就失效了 |

@Target 表示注解可以贴在哪些位置（类，方法，构造器上等）

位置的常量封装在ElementType枚举类中：

|  |  |
| --- | --- |
| 枚举常量摘要 | |
| ElementType.ANNOTATION\_TYPE | 修饰Annotation |
| ElementType.CONSTRUCTOR | 修饰构造方法 |
| ElementType.FIELD | 修饰字段（属性），包括枚举常量 |
| ElementType.LOCAL\_VARIABLE | 修饰局部变量 |
| ElementType.METHOD | 修饰方法 |
| ElementType.PACKAGE | 修饰包（极少使用） |
| ElementType.PARAMETER | 修饰参数 |
| ElementType.TYPE | 修饰类，接口，枚举 |

#### 3、自定义注解（@Interface）

1. 定义一个注解
2. 被贴的程序元素
3. 第三方程序参与赋予注解功能（反射程序）

*/\*\**

*\** ***@author*** *chance  
 \*  
 \* Target表示注解可以贴在哪些程序元素上面  
 \*  
 \* Retention表示注解可以保留到哪个阶段，一般自定义保留到RUNTIME运行时  
 \*/*@Target(ElementType.*ANNOTATION\_TYPE*)  
@Retention(RetentionPolicy.*RUNTIME*)  
public @interface Table {  
  
 */\*\*  
 \* 注解属性  
 \* 语法  
 \* 数据类型  
 \* 如果属性名称是value，在使用注解的时候，可以省略value  
 \*   
 \*/* String name();  
}