**lombok**

lombok 提供了简单的注解的形式来帮助我们简化消除一些必须有但显得很臃肿的 java 代码。通过使用对应的注解，可以在编译源码的时候生成对应的方法，所以不会影响任何运行效率。

常用的 lombok 注解：

@Data   ：注解在类上；包含了@ToString，@EqualsAndHashCode，@Getter / @Setter和@RequiredArgsConstructor的功能，提供类所有属性的 getter 和 setter 方法，此外还提供了equals、canEqual、hashCode、toString 方法

@Setter：注解在属性上；为属性提供 setter 方法

@Getter：注解在属性上；为属性提供 getter 方法

@ToString：注解在类上；生成toString()方法，默认情况下，它会按顺序（以逗号分隔）打印你的类名称以及每个字段。可以这样设置不包含哪些字段@ToString(exclude = "id") / @ToString(exclude = {"id","name"})

如果继承的有父类的话，可以设置callSuper 让其调用父类的toString()方法，例如：@ToString(callSuper = true)

@EqualsAndHashCode：注解在类上；生成hashCode()和equals()方法，默认情况下，它将使用所有非静态，非transient字段。但可以通过在可选的exclude参数中来排除更多字段。或者，通过在parameter参数中命名它们来准确指定希望使用哪些字段。

@NonNull：  注解在属性上；标识属性是不能为空，为空则抛出异常。

@Slf4j ：注解在类上；根据用户实际使用的日志框架生成log日志对象。

@Log4j ：注解在类上；为类提供一个 属性名为log 的 log4j 日志对象

@NoArgsConstructor：注解在类上；为类提供一个无参的构造方法。当类中有final字段没有被初始化时，编译器会报错，此时可用@NoArgsConstructor(force = true)，然后就会为没有初始化的final字段设置默认值 0 / false / null。对于具有约束的字段（例如@NonNull字段），不会生成检查或分配，因此请注意，正确初始化这些字段之前，这些约束无效。

@AllArgsConstructor：注解在类上；为类提供一个全参的构造方法

默认生成的方法是public的，如果要修改方法修饰符可以设置AccessLevel的值，例如：@Getter(access = AccessLevel.PROTECTED)

@RequiredArgsConstructor：注解在类上；会生成构造方法（可能带参数也可能不带参数），如果带参数，这参数只能是以final修饰的未经初始化的字段，或者是以@NonNull注解的未经初始化的字段@RequiredArgsConstructor(staticName = "of")会生成一个of()的静态方法，并把构造方法设置为私有的。