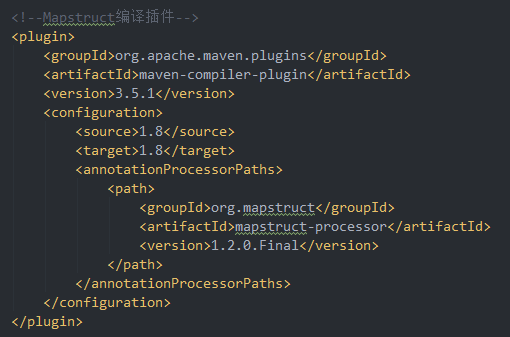
# MapStruct

## 开发环境搭建

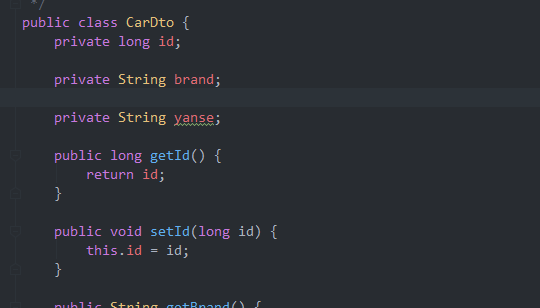
1. 引入依赖包，插件配置

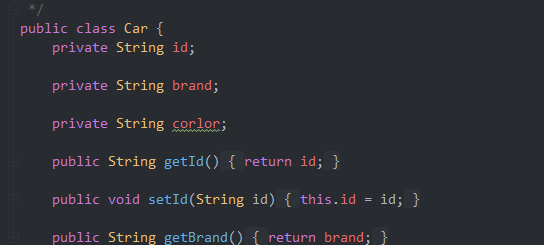




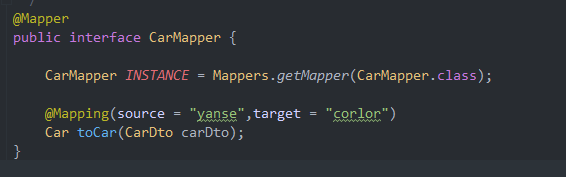
## 开发步骤

* 1. 编写实体类





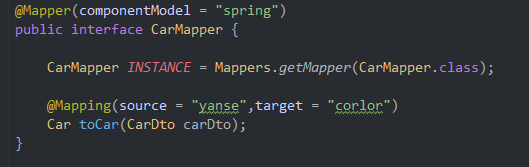
* 1. Mapper类实现转换



* 1. 编译文件

# 具体注解

1. 注解@Mapper
   1. 属性componentModel



用于指定自动生成的接口实现类的组件类型，有4个值。

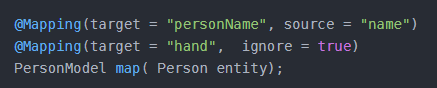
1. default: 这是默认的情况，mapstruct不使用任何组件类型, 可以通过Mappers.getMapper(Class)方式获取自动生成的实例对象。
2. cdi: the generated mapper is an application-scoped CDI bean and can be retrieved via @Inject
3. spring: 生成的实现类上面会自动添加一个@Component注解，可以通过Spring的 @Autowired方式进行注入
4. jsr330: 生成的实现类上会添加@javax.inject.Named 和@Singleton注解，可以通过 @Inject注解获取。
5. 注解@Mapping
   1. @Mapping和@Mappings

@Mapping单个转换属性时使用，如果有多个属性值需要转换，则用@Mappings包裹在外面。

* 1. 属性source和属性target

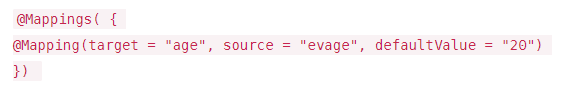
1. Source指被转换的属性名称，target指目标的属性名称，用于两个实体类名称不相同的时候。
   1. 属性ignore
2. 当目标实体类的属性多于源实体类的时候，ignore表示忽略目标实体类的某一个属性。

结合target使用。



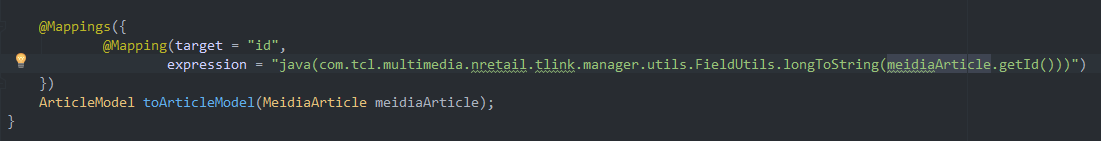
* 1. 属性defaultValue

1. 在转换过程中，可能因为空值或其他原因使得映射结果不正确，此时可以指定一个默认值，防止程序出错。



* 1. 属性expressions

1. 当目标属性值和源属性值存在逻辑关系的时候，可以构造一些简单的表达式来处理这种关系。一般结合target使用。



1. 必须用java包裹，写出类名的全称。