上一篇文章中介绍了一下什么是PO、BO、VO、DTO，以及在服务中的使用

服务之间定义不同的POJO是为了解耦，减少维护工作量，不过pojo之间的转换也是一个很大的工作量，工作量大在哪里呢？

1、分层并将pojo分类后，pojo的数量是原来的n倍，pojo属性调整，需修改get和set方法，将是很繁琐的重复工作

2、pojo间转换，两个不同pojo需要定义转换方法，pojo的转换要定义不大量同的方法，pojo属性有变动，会连锁修改一堆的方法

原来是为了减轻工作量，现在看来反而增加了不少工作，那怎么解决呢？

我选择的方案是dozer + lombok

用lombok来省略get和set方法

通过dozer来实现不同pojo的转换工作

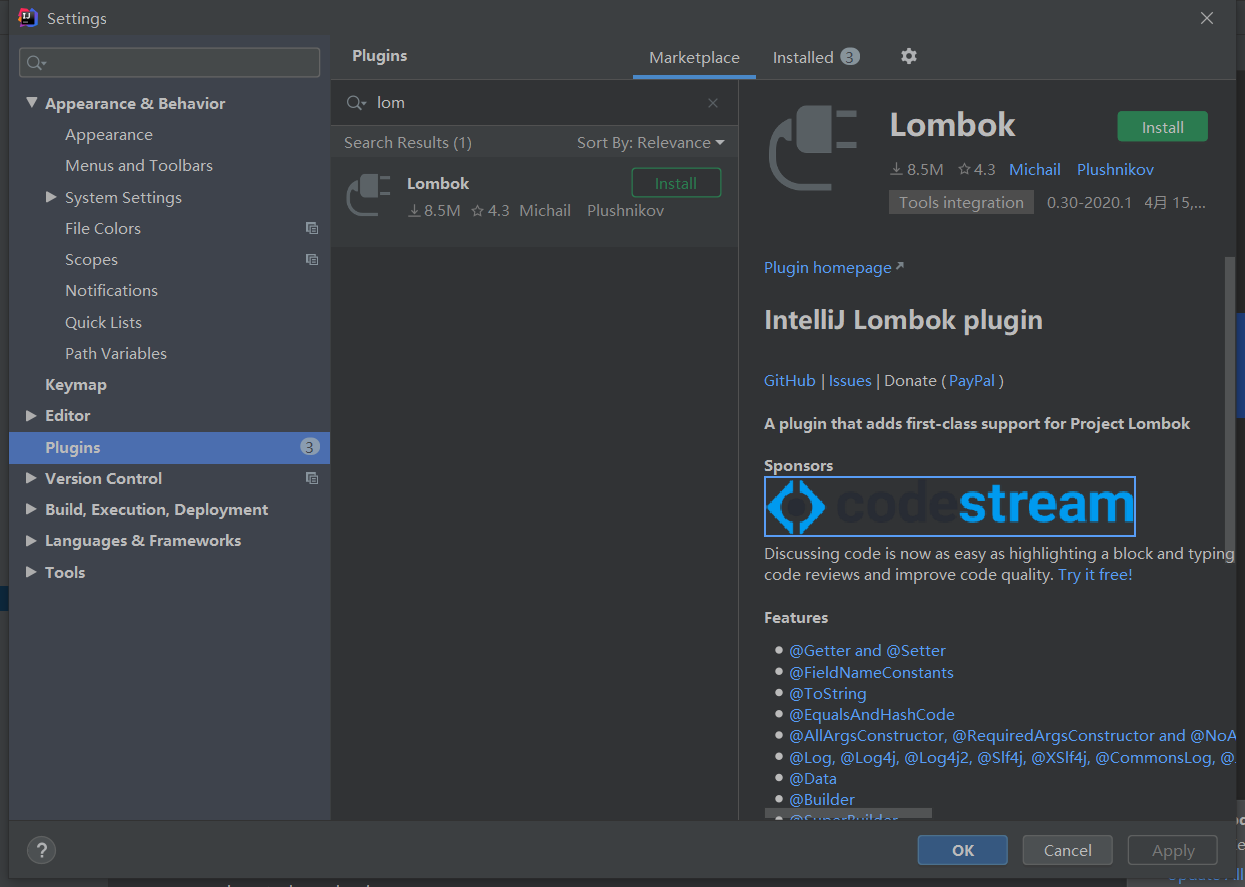
那具体怎么做呢？（接下来的内容，手动实践一下效果会比较好）

一、lombok的使用

首先创建一个maven项目，引入lombok，如下

<dependency>  
 <groupId>org.projectlombok</groupId>  
 <artifactId>lombok</artifactId>  
 <version>1.18.8</version>  
</dependency>

然后给idea安装lombok插件，如下图



现在让我们创建一个user类型，包含 id，name，age三个属性，通常的写法是这样的

**public** **class** User {

**private** **long** id;

**private** String name;

**private** **int** age;

**public** **long** getId() {

**return** id;

}

**public** **void** setId(**long** id) {

**this**.id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** **int** getAge() {

**return** age;

}

**public** **void** setAge(**int** age) {

**this**.age = age;

}

}

现在可以这么写了

@Data  
public class User {

**private** **long** id;

**private** String name;

**private** **int** age;

}

是不是很简单啊，感觉代码量少了很多。如果感觉@Data太简单暴力，也可以用@Getter和@Setter注解。

Lombok还有很多其他作用的注解，如

@ToString 可以重写String方法，通过of定义现实的字段

@**NoArgsConstructor 可以生成无参数构造器**

**@**AllArgsConstructor 生成带所有参数的构造器

@RequiredArgsConstructor 生成有参数的构造器，参数需要用**@NonNull注释一下**

等等，其他的还有很多用法，这里不做详细的描述了，咱们现在只需要通过@Data或者@Getter和@Setter解决修改get和set方法的问题就可以了

那么接下来让我们用dozer来解决类的转换问题

二、dozer的使用

还是要先引入maven

<dependency>  
 <groupId>net.sf.dozer</groupId>  
 <artifactId>dozer</artifactId>  
 <version>5.5.1</version>  
</dependency>

然后分别创建一个UserVo和UserDto，并对他们做转换

首先让两个类的属性名称一样，如下

@Data  
public class UserVo {  
 private String name;  
 private int id;  
}

@Data  
public class UserDto {  
 private String name;  
 private int id;  
}

接下来怎么转换呢，代码如下

Mapper dozerMapper = new DozerBeanMapper();  
  
UserVo userVo = new UserVo();  
userVo.setName("mg");  
userVo.setId(10);  
  
InUserDto userDto = dozerMapper.map(userVo,UserDto.class);  
System.*out*.println(userDto.getName());

输出结果为

Mg

UserVo转换为UserDto,这是属性完全相同的情况，不过通常属性名称都是不同的，那怎么办呢

修改UserVo 为

@Data  
public class UserVo {  
 private String sname;  
 private int sid;  
}

修改UserDto为

@Data  
public class UserDto {  
  
 @Mapping("sname")  
 private String name;  
 @Mapping("sid")  
 private int id;  
}

在执行下刚写的转换函数，试试是否能成功转换了

是不是很容易啊，这样既能分别定义的VO、PO、DTO，又能快速完成开发工作了

**最后需要注意下：**

**lombok这个插件虽然好用，不过外界他的褒贬不一，用和不用一直是热门话题，有不少大佬不提倡使用lombok，不过我还是挺喜欢用的，实在太方便了。**