## 框架介绍

pom.xml ：配置meven第三方框架包。

controller：控制层。

mapper：数据层，注意：接口名必须和mapper下namespace一致，方法名必须和mapper下的id一致。

service：业务逻辑层。

用到的注解：

@Service：加入到业务层容器中。

@Autowired：当Spring发现。

@Autowired注解时，将自动在代码上下文中找到和其匹配（默认是类型匹配）的Bean，并自动注入到相应的地方去。

@Autowired注解时，属性会自动创建对象的。

@Resource 和 @Autowired用法相似，但是@Autowired是Spring的，@Resource是J2EE的。

@Controller：自动加入控制层。

## mybatis

<!--

注释：command+／，在英文的情况下

学习概要：

java db code

mybatis：

对原生态的jdbc程序（单独使用jdbc开发）问题总结

原理：

增删改查

mybatis开发dao两种方法：

1.原始的dao开发方法（程序需要编写dao接口和dao实现类）

2.mybatis的mapper接口（相当于dao接口）代理开发方法

mybatis配置文件：SqlMapConfig.xml

mabatis的核心：

mybatis 输出映射：

mubatis 输入映射：

mybaits的动态sql：

高级映射（一对一，一对多，多对多）

mybitas延迟加载，

mybitas查询缓存（一级缓存，二级缓存）

mybitas和Spring进行整合

mybitas逆向工程

### resultType和resultMap

resultType 非链表查询的结果，指定的是单条的java对象，查询出来如果是多条则就是list<object>结合。

resultMap 连标查询的结果,查询出来的和对象属性不匹配，比如有对象不包括的数据。

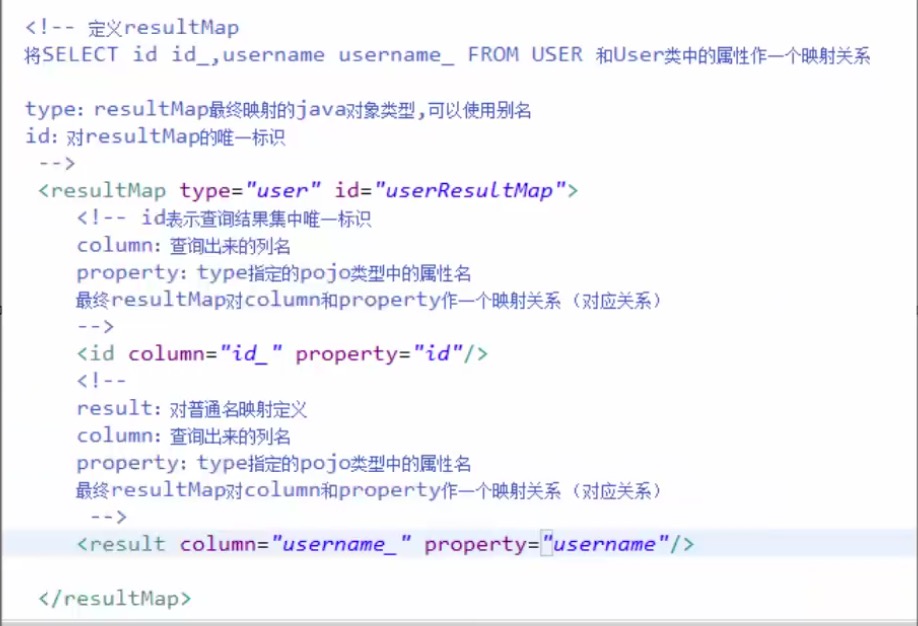
可以说resultType返回对象结果。resultMap返回map的key、value。

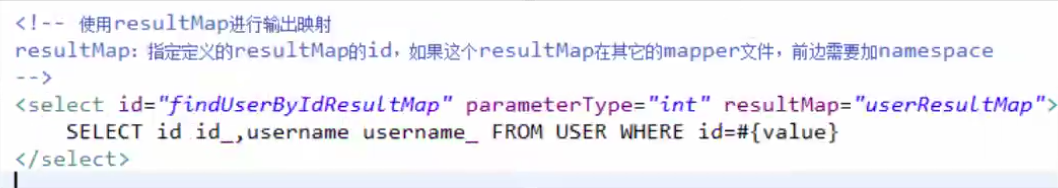
<resultMap type="user" id="userResultMap"> :

type:resultMap最终映射的java对象类型，可以使用别名。

id： 对resultMap的唯一标识。

如图：





### parameterType

传入的参数名称

#{name} : 其中的name表示接收输入的参数，如果输入的是简单类型，#{}中的参数名可以任意，可以是value或者其他名称。

${}：表示拼接sql串，将接收到的参数的内容，不加任何的修饰，拼接在sql中。使用会引起sql注入。

${value}：接收输入参数的内容，如果传入的是简单类型，${}中只能是value。

### selectOne：

表示查询出一条数据记录进行映射，如果使用selectOne可以实现，自然也可以使用selectList来实现，

但是selectList只有一条记录

### selectList：

表示查询出一个列表数据，惊醒映射，如果使用selectList查询多条记录，不能使用selectOne。

### 自增主键返回：

可以参考登录sql插入注册时的代码。

SELECT LAST\_INSERT\_ID():得到刚执行插入记录的主键值，只使用与自增。

### keyProperty：

将查询到主建值设置到parameterType指定的对象的哪个属性。

### order：

SELECT LAST\_INSERT\_ID() AS id 执行顺序，相对于insert语句来说它的执行顺序，在插入之后

### useGeneratedKeys

取值范围true|false 默认值是：false。

含义：设置是否使用JDBC的getGenereatedKeys方法获取主键并赋值到keyProperty设置的领域模型属性中。

selectKey返回最近插入的id。请对比接口一看便知，会自动将id返回出来。

### 非自增主键返回（使用uuid）:

执行思路：

先通过uuid查询到主键，将主键输入到sql语句中。

执行uudi语句顺序相当于insert语句之前。

使用mysql的uuid()生成主键，

执行过程：

首先通过uuid()得到主键，将主键值设置到id主键位置，再insert执行，从user对象中，取出id属性值。

代码：

<insert id="insertUser" useGeneratedKeys="true" keyProperty="id">

<selectKey resultType="java.lang.Integer" keyProperty="id" order="BEFORE">

SELECT uuid()

</selectKey>

INSERT INTO `user`

(

`id`,`name`,pwd,age,sex,idCard,staffNum,phone,department,

post,headImagePatch,telephone,mail,bossName,bossid

)

VALUES

(

#{id},#{name},#{pwd},#{age},#{sex},#{idCard},#{staffNum},#{phone},#{department},

#{post},#{headImagePatch},#{telephone},#{mail},#{bossName},#{bossid}

);

</insert>

### 小结：

#{}和${}都可以接收pojo对象值，通过OGNL对象导航图来读取对象中的属性值，通过属性.属性.属性...的方式获取属性值。

#### mybatis 和 hibernate 的本质和应用场景。

hibernate：入门门槛高。他是一个标准的ORM（对象关系映射）框架。不需要程序员写sql，自动生成，sql优化，

特有的函数优化，表关联优化，修改比较困难。

应用场景：使用需求变化不多的，固定的中小型项目，比如：erp，orm，oa。

mybatis：专注sql本身，需要自己编写sql，管理起来方便，不完全的ORM框架，自己写sql，也有输入输出映射。

应用场景：适用于需求变化较多的。比如：互联网项目。

企业进行技术选型，低成本，高回报作为技术选型的原则，根据项目组的技术力量进行选择。

#### Mybatis开发Dao

#### SqlMapConfig.xml配置

SqlMapConfig.xml（工程中的spring.xml文件）的配置内容和顺序如下（顺序不能乱）：

Properties（属性）

Settings（全局参数设置）：在运行时，可以调整运行的参数，比如：开启二级缓存，开启延迟加载。。。

typeAliases（类型别名-重点）：分为默认别名和自定义别名。

单个别名的定义：<typeAliases type="路径" alias="别名名称"></typeAliases>

引用别名：直接拿着别名名称使用。

批量别名的定义： <typeAliases> <package name="model" /> </typeAliases>会自动扫描包中的po类，自动的创建定义别名，

别名就是类名，首字母大写或者小写。

typeHandlers（类型处理器）：在mybitas中通过typeHandlers完成jdbc类型和java类型转换。

通常情况下，mybatis提供的类型处理器满足日常需要，不需要自定义。

objectFactory（对象工厂）

plugins（插件）

environments（环境信息集合）

environment（单个环境信息）

transactionManager（事物）

dataSource（数据源）

mappers（映射器配置-重点）

configLocation、mapperLocations

-->