## 框架认识

1.什么是框架？  
框架就是一些类和接口的集合，通过这些类和接口协调来完成一系列的程序实现。有了框架，我们就可以集中精力进行业务逻辑的开发而不用去关心它的技术实现以及一些辅助的业务逻辑。  
说白了，框架就是封装好方便程序员操作的类，使项目的开发更简单，维护起来也更容易。  
框架的本体，实际上是一些jar包和配置文件。  
2.为什么要使用框架？  
框架的优越性在于：  
(1)集中精力完成系统的业务逻辑设计；  
(2)使程序更容易扩展和维护，稳定性更强；  
(3)处理系统很多细节问题，比如安全性问题。

二、框架的使用  
(1)导入相关的jar包  
(2)配置项目配置文件（比如web.xml），让项目用框架处理各种行为  
(3)添加框架配置文件（比如struts-config.xml），设定框架如何处理各种行为（比如接收到某个请求时，调用哪个类处理）  
(4)编写业务逻辑（继承、实现框架中的类、接口），完成具体处理操作。

## mybatis

### 核心

**运行示例**

public class MybatisTest {  
 public static void main(String[] args) throws Exception {  
 String resource = "mybatis-config.xml";  
 InputStream inputStream = Resources.getResourceAsStream(resource);  
 //核心步骤，加载所有相关配置,会将所有信息封装到configuration类中  
 SqlSessionFactory sqlSessionFactory = new SqlSessionFactoryBuilder().build(inputStream);  
 //sqlSession用于使用MyBatis的主要Java接口  
 SqlSession sqlSession=sqlSessionFactory.openSession();  
 try {  
 //生成动态代理mapper，实际执行的是sqlsession的方法  
 BlogMapper blogMapper = sqlSession.getMapper(BlogMapper.class);  
 Integer id=101;  
 String name="tom";  
 //具体实现由sqlsession实现，从configuration获取sql，再获取数据库连接connection执行sql  
 Blog blog = blogMapper.selectBlogById(id,name);  
 System.out.println(blog.getId());  
 } finally {  
 sqlSession.close();  
 }  
 }  
}

**最核心的类**

Configuration:启动mybatis后所有配置信息的实例，应用开启后就一个实例，当xml改变后若想生效则要重新加载configuration实例

SeqSession：增删改查等操作的接口，不是线程安全的，spring中每次请求都会产生新的实例，所以作用域应该同request。

mybatis官方文档<https://mybatis.org/mybatis-3/zh/index.html>

mybatis-spring官方文档<http://mybatis.org/spring/zh/index.html>

mybatis源码<https://github.com/gitqiyun/mybatis-3-src>

### 常见面试题

**1、#{}和${}的区别是什么？**

#{}是预编译处理，${}是字符串替换。

Mybatis在处理#{}时，会将sql中的#{}替换为?号，调用PreparedStatement的set方法来赋值，会做特殊处理由第三发驱动实现，有效的防止SQL注入，提高系统安全性。

Mybatis在处理${}时，就是把${}替换成变量的值，select \* from user where name = ${name}，会变成select \* from user where name =tom直接替换没有单引号查询会报错

一般用于传入数据库对象，例如传入表名。（具体实现查看源码是如何拼接sql语句的）

**2、为什么说Mybatis是半自动ORM映射工具？它与全自动的区别在哪里？**

（1）Hibernate属于全自动ORM映射工具，使用Hibernate查询关联对象或者关联集合对象时，可以根据对象关系模型直接获取，所以它是全自动的。而Mybatis在查询关联对象或关联集合对象时，需要手动编写sql来完成，所以，称之为半自动ORM映射工具。

（2）Mybatis直接编写原生态sql，可以严格控制sql执行性能，灵活度高，非常适合对关系数据模型要求不高的软件开发，因为这类软件需求变化频繁，一但需求变化要求迅速输出成果。但是灵活的前提是mybatis无法做到数据库无关性，如果需要实现支持多种数据库的软件，则需要自定义多套sql映射文件，工作量大。

**3、Mybaits的优缺点**

Mybaits的优点：

（1）基于SQL语句编程，相当灵活，不会对应用程序或者数据库的现有设计造成任何影响，SQL写在XML里，解除sql与程序代码的耦合，便于统一管理；提供XML标签，支持编写动态SQL语句，并可重用。

（2）与JDBC相比，减少了50%以上的代码量，消除了JDBC大量冗余的代码，不需要手动开关连接；

（3）很好的与各种数据库兼容（因为MyBatis使用JDBC来连接数据库，所以只要JDBC支持的数据库MyBatis都支持）。

（4）能够与Spring很好的集成；

（5）提供映射标签，支持对象与数据库的ORM字段关系映射；提供对象关系映射标签，支持对象关系组件维护。

MyBatis框架的缺点：

（1）SQL语句的编写工作量较大，尤其当字段多、关联表多时，对开发人员编写SQL语句的功底有一定要求。

（2）SQL语句依赖于数据库，导致数据库移植性差，不能随意更换数据库。

MyBatis框架适用场合：

（1）MyBatis专注于SQL本身，是一个足够灵活的DAO层解决方案。

（2）对性能的要求很高，或者需求变化较多的项目，如互联网项目，MyBatis将是不错的选择。

**4、Xml映射文件中，除了常见的select|insert|updae|delete标签之外，还有哪些标签？**

答：还有很多其他的标签，<resultMap>、<parameterMap>、<sql>、<include>、<selectKey>，加上动态sql的9个标签，trim|where|set|foreach|if|choose|when|otherwise|bind等，通过<include>标签引入sql片段，<selectKey>为不支持自增的主键生成策略标签。

**5、最佳实践中，通常一个Xml映射文件，都会写一个Dao接口与之对应，请问，这个Dao接口的工作原理是什么？Dao接口里的方法，参数不同时，方法能重载吗？**

Dao接口，就是人们常说的Mapper接口，接口的全限名，就是映射文件中的namespace的值，接口的方法名，就是映射文件中MappedStatement的id值，接口方法内的参数，就是传递给sql的参数。Mapper接口是没有实现类的，当调用接口方法时，接口全限名+方法名拼接字符串作为key值，可唯一定位一个MappedStatement

com.mybatis3.mappers.StudentDao.findStudentById，可以唯一找到namespace为com.mybatis3.mappers.StudentDao下面id = findStudentById的MappedStatement。在Mybatis中，每一个<select>、<insert>、<update>、<delete>标签，都会被解析为一个MappedStatement对象。

Dao接口里的方法，是不能重载的，因为是全限名+方法名的保存和寻找策略。

Dao接口的工作原理是JDK动态代理，Mybatis运行时会使用JDK动态代理为Dao接口生成代理proxy对象，代理对象proxy会拦截接口方法，转而执行MappedStatement所代表的sql，然后将sql执行结果返回。

**6、Mybatis是如何进行分页的？分页插件的原理是什么？**

Mybatis使用RowBounds对象进行分页，它是针对ResultSet结果集执行的内存分页，而非物理分页，可以在sql内直接书写带有物理分页的参数来完成物理分页功能，也可以使用分页插件来完成物理分页。分页插件的基本原理是使用Mybatis提供的插件接口，实现自定义插件，在插件的拦截方法内拦截待执行的sql，然后重写sql，根据dialect方言，添加对应的物理分页语句和物理分页参数。

举例：select \* from student，拦截sql后重写为：select t.\* from （select \* from student）t limit 0，10

**7、Mybatis执行批量插入，能返回数据库主键列表吗？**

能，JDBC都能，Mybatis当然也能。

1、对于支持生成自增主键的数据库：增加 useGenerateKeys和keyProperty ，<insert>标签属性。

2、不支持生成自增主键的数据库：使用<selectKey>。

**8、Mybatis动态sql是做什么的？都有哪些动态sql？能简述一下动态sql的执行原理不？**

答：Mybatis动态sql可以让我们在Xml映射文件内，以标签的形式编写动态sql，完成逻辑判断和动态拼接sql的功能，Mybatis提供了9种动态sql标签trim|where|set|foreach|if|choose|when|otherwise|bind。

其执行原理为，使用OGNL从sql参数对象中计算表达式的值，根据表达式的值动态拼接sql，以此来完成动态sql的功能。

**9、Mybatis是否支持延迟加载？如果支持，它的实现原理是什么？**

Mybatis仅支持association关联对象和collection关联集合对象的延迟加载，association指的就是一对一，collection指的就是一对多查询。在Mybatis配置文件中，可以配置是否启用延迟加载lazyLoadingEnabled=true|false。

它的原理是，使用CGLIB创建目标对象的代理对象，当调用目标方法时，进入拦截器方法，比如调用a.getB().getName()，拦截器invoke()方法发现a.getB()是null值，那么就会单独发送事先保存好的查询关联B对象的sql，把B查询上来，然后调用a.setB(b)，于是a的对象b属性就有值了，接着完成a.getB().getName()方法的调用。这就是延迟加载的基本原理。当然了，不光是Mybatis，几乎所有的包括Hibernate，支持延迟加载的原理都是一样的。

**10、Mybatis的Xml映射文件中，不同的Xml映射文件，id是否可以重复？**

答：不同的Xml映射文件，如果配置了namespace，那么id可以重复；如果没有配置namespace，那么id不能重复；毕竟namespace不是必须的，只是最佳实践而已。

原因就是namespace+id是作为Map<String, MappedStatement>的key使用的，如果没有namespace，就剩下id，那么，id重复会导致数据互相覆盖。有了namespace，自然id就可以重复，namespace不同，namespace+id自然也就不同。

**11、Mybatis映射文件中，如果A标签通过include引用了B标签的内容，请问，B标签能否定义在A标签的后面，还是说必须定义在A标签的前面？**

答：虽然Mybatis解析Xml映射文件是按照顺序解析的，但是，被引用的B标签依然可以定义在任何地方，Mybatis都可以正确识别。

原理是，Mybatis解析A标签，发现A标签引用了B标签，但是B标签尚未解析到，尚不存在，此时，Mybatis会将A标签标记为未解析状态，然后继续解析余下的标签，包含B标签，待所有标签解析完毕，Mybatis会重新解析那些被标记为未解析的标签，此时再解析A标签时，B标签已经存在，A标签也就可以正常解析完成了。

**12.当实体类中的属性名和表中的字段名不一样 ，怎么办 ？**

第1种： 通过在查询的sql语句中定义字段名的别名，让字段名的别名和实体类的属性名一致。

第2种： 通过<resultMap>来映射字段名和实体类属性名的一一对应的关系。

**13、Mybatis都有哪些Executor执行器？它们之间的区别是什么？**

答：Mybatis有三种基本的Executor执行器，SimpleExecutor、ReuseExecutor、BatchExecutor。

SimpleExecutor：每执行一次update或select，就开启一个Statement对象，用完立刻关闭Statement对象。

ReuseExecutor：执行update或select，以sql作为key查找Statement对象，存在就使用，不存在就创建，用完后，不关闭Statement对象，而是放置于Map<String, Statement>内，供下一次使用。简言之，就是重复使用Statement对象。

BatchExecutor：执行update（没有select，JDBC批处理不支持select），将所有sql都添加到批处理中（addBatch()），等待统一执行（executeBatch()），它缓存了多个Statement对象，每个Statement对象都是addBatch()完毕后，等待逐一执行executeBatch()批处理。与JDBC批处理相同。

作用范围：Executor的这些特点，都严格限制在SqlSession生命周期范围内。

**14、Mybatis的一级、二级缓存**

1）一级缓存: 基于 PerpetualCache 的 HashMap 本地缓存，其存储作用域为 Session，当 Session flush 或 close 之后，该 Session 中的所有 Cache 就将清空，默认打开一级缓存。

2）二级缓存与一级缓存其机制相同，默认也是采用 PerpetualCache，HashMap 存储，不同在于其存储作用域为 Mapper(Namespace)，并且可自定义存储源，如 Ehcache。默认不打开二级缓存，要开启二级缓存，使用二级缓存属性类需要实现Serializable序列化接口(可用来保存对象的状态),可在它的映射文件中配置<cache/> ；

3）对于缓存数据更新机制，当某一个作用域(一级缓存 Session/二级缓存Namespaces)的进行了C/U/D 操作后，默认该作用域下所有 select 中的缓存将被 clear。

**15、使用MyBatis的mapper接口调用时有哪些要求？**

①  Mapper接口方法名和mapper.xml中定义的每个sql的id相同；  
②  Mapper接口方法的输入参数类型和mapper.xml中定义的每个sql 的parameterType的类型相同；  
③  Mapper接口方法的输出参数类型和mapper.xml中定义的每个sql的resultType的类型相同；  
④  Mapper.xml文件中的namespace即是mapper接口的类路径。

**16、简述Mybatis的插件运行原理，以及如何编写一个插件。**

答：Mybatis仅可以编写针对ParameterHandler、ResultSetHandler、StatementHandler、Executor这4种接口的插件，Mybatis使用JDK的动态代理，为需要拦截的接口生成代理对象以实现接口方法拦截功能，每当执行这4种接口对象的方法时，就会进入拦截方法，具体就是InvocationHandler的invoke()方法，当然，只会拦截那些你指定需要拦截的方法。

编写插件：实现Mybatis的Interceptor接口并复写intercept()方法，然后在给插件编写注解，指定要拦截哪一个接口的哪些方法即可，记住，别忘了在配置文件中配置你编写的插件。

## spring

### ioc

### di

### aop

## springmvc

## springboot

## springcloud

## activitymq

## activity