一、Mybatis

1、简介

MyBatis 是一个基于 Java 的持久层框架，是对 jdbc 的封装，它让数据库底层操作变的透明。Mybatis 的操作都是围绕一个 sqlSessionFactory 实例展开的。Mybatis 通过配置文件关联到各实体类的 Mapper 文件，Mapper 文件中配置了每个类对数据库所需进行的 sql 语句映射。在每次与数据库交互时，通过 sqlSessionFactory 拿到一个sqlSession，再执行 sql 命令。在 Mybatis 中，sqlSessionFactory 可以看为一个数据库连接池，并使用的单例模式；sqlSession 就相当于一个数据库连接，它存活在某个具体的业务请求中。 创建映射器 映射器是 Mybatis 中最重要、最复杂的组件，它由一个接口和对应的 XML 文件(或注解组成)。

2、工作原理：

mybatis通过sqlSessionFactoryBuilder从mybatis-config.xml配置文件(如上图第一步)中构建出sqlSessionFactory(线程安全);然后调用sqlSessionFactory的openSession()方法直接开启一个sqlSession,在通过sqlSession实例获得Mapper.xml对象并运行Mapper.xml映射的sql语句,完成对数据库的CRUD和事务提交,每完成一次操作之后sqlSession都需要关闭.

3、工作流程图：

二、SpringMVC

1、MVC工作原理：

M:Model(完成业务逻辑 Service/dao/entity/)

V:View(完成界面渲染 jsp/html)

C:Controller(控制器->类似于CPU 接受请求->调用M->返回V)

1.1、MVC工作原理图：

2.1、SpringMVC工作原理

SpringMVC是一个MVC的开源框架，SpringMVC是Spring的一个后续产品，其实就是Spring在原有基础上，又提供了web应用的MVC模块，可以简单的把SpringMVC理解为是spring的一个模块（类似AOP，IOC这样的模块），网络上经常会说SpringMVC和Spring无缝集成，其实SpringMVC就是Spring的一个子模块，所以根本不需要同spring进行整合。

2.2、SpringMVC中的组件：

前端控制器（DispatcherServlet）：接收请求，响应结果，相当于电脑的CPU。

处理器映射器（HandlerMapping）：根据URL去查找处理器

处理器（Handler）：（需要程序员去写代码处理逻辑的）

处理器适配器（HandlerAdapter）：会把处理器包装成适配器，这样就可以支持多种类型的处理器，类比笔记本的适配器（适配器模式的应用）

视图解析器（ViewResovler）：进行视图解析，多返回的字符串，进行处理，可以解析成对应的页面

2.3、SpringMvc工作原理图

2.4、 SpringMvc工作流程

第一步:用户发起请求到前端控制器（DispatcherServlet）

第二步：前端控制器请求处理器映射器（HandlerMapping）去查找处理器（Handler）：通过xml配置或者注解进行查找

第三步：找到以后处理器映射器（HandlerMappping）像前端控制器返回执行链（HandlerExecutionChain）

第四步：前端控制器（DispatcherServlet）调用处理器适配器（HandlerAdapter）去执行处理器（Handler）

第五步：处理器适配器去执行Handler

第六步：Handler执行完给处理器适配器返回ModelAndView

第七步：处理器适配器向前端控制器返回ModelAndView

第八步：前端控制器请求视图解析器（ViewResolver）去进行视图解析

第九步：视图解析器像前端控制器返回View

第十步：前端控制器对视图进行渲染

第十一步：前端控制器向用户响应结果

2.5、 SpringMVC重点

在于它的流程和注解，包括控制器、视图解析器、视图等内容；

三、Spring

1、简介：

Spring是一种多层的J2EE应用程序框架，其核心就是提供一种新的机制管理业务对象及其依赖关系。它是一种容器框架，用于创建bean，维护bean之间的关系，它可以管理web层，持久层，业务层等，可以配置各个层的组件并且维护各个层的关系

2、Spring原理：

内部最核心的就是IOC了，动态注入，让一个对象的创建不用new了，可以自动的生产，这其实就是利用java里的反射，反射其实就是在运行时动态的去创建、调用对象，Spring就是在运行时，跟xml Spring的配置文件来动态的创建对象，和调用对象里的方法的 。

Spring还有一个核心就是AOP这个就是面向切面编程，可以为某一类对象 进行监督和控制（也就是 在调用这类对象的具体方法的前后去调用你指定的 模块）从而达到对一个模块扩充的功能。这些都是通过 配置类达到的。

Spring目的：就是让对象与对象（模块与模块）之间的关系没有通过代码来关联，都是通过配置类说明管理的（Spring根据这些配置 内部通过反射去动态的组装对象）

要记住：Spring是一个容器，凡是在容器里的对象才会有Spring所提供的这些服务和功能。

Spring里用的最经典的一个设计模式就是：模板方法模式。(这里我都不介绍了，是一个很常用的设计模式)， Spring里的配置是很多的，很难都记住，但是Spring里的精华也无非就是以上的两点，把以上两点跟理解了 也就基本上掌握了Spring.

3、Spring两大核心： AOP与IOC

作用：

为程序解耦和代码的简介提供了支持。

3.1、 IOC(Inversion of control): 控制反转

1、IoC：

概念：控制权由对象本身转向容器；由容器根据配置文件去创建实例并创建各个实例之间的依赖关系

核心：bean工厂；在Spring中，bean工厂创建的各个实例称作bean

3.2、AOP(Aspect-Oriented Programming): 面向方面编程

1、 代理的两种方式：

静态代理：

 针对每个具体类分别编写代理类；

 针对一个接口编写一个代理类；

动态代理：

针对一个方面编写一个InvocationHandler，然后借用JDK反射包中的Proxy类为各种接口动态生成相应的代理类

动态代理知识补充:

不用写代理类，虚拟机根据真实对象实现的接口产生一个类，通过类实例化一个动态代理，在实例化动态代理时将真实对象及装备注入到动态代理中，向客户端公开的是动态代理，当客户端调用动态代理方法时，动态代理根据类的反射得到真实对象的Method,调用装备的invoke方法，将动态代理、

Method、方法参数传与装备的invoke方法，invoke方法在唤起method方法前或后做一些处理。