# **Spring知识点概述**

## (1)什么是Spring?

Spring是一个从实际项目开发经验中抽取的，可高度重用的应用框架。

Spring是一个轻量级容器，它所带的包装器使许多不同的服务和框架更易于使用。轻量级容器接受任何JavaBean，而不是只接受特定类型的组件。

## (2)Spring框架各个模块介绍

Spring Core：核心容器提供 Spring 框架的基本功能。核心容器的主要组件是BeanFactory，它是工厂模式的实现。BeanFactory 使用控制反转 （IOC） 模式将应用程序的配置和 依赖性规范与实际的应用程序代码分开。

Context：Spring 上下文是一个配置文件，向 Spring 框架提供上下文信息。Spring 上下文包括企业服务，例如 JNDI、EJB、电子邮件、国际化、校验和调度功能。

Spring AOP：通过配置管理特性，Spring AOP 模块直接将面向方面的编程功能集成到了 Spring 框架中。所以，可以很容易地使 Spring 框架管理的任何对象支持 AOP。Spring AOP 模块为基于 Spring 的应用程序中的对象提供了事务管理服务。通过使用 Spring AOP，不用依赖 EJB 组件，就可以将声明性事务管理集成到应用程序中。

Spring DAO：JDBC DAO 抽象层提供了有意义的异常层次结构，可用该结构来管理异常处理和不同数据库供应商抛出的错误消息。异常层次结构简化了错误处理，并且极大地降低了需要编写的异常代码数量（例如打开和关闭连接）。Spring DAO 的面向 JDBC 的异常遵从通用的 DAO 异常层次结构。

Spring ORM：Spring 框架插入了若干个 ORM 框架，从而提供了 ORM 的对象关系工具，其中包括 JDO、Hibernate 和 iBatis SQL Map。所有这些都遵从 Spring 的通用事务和 DAO 异常层次结构。

Spring Web ：Web 上下文模块建立在应用程序上下文模块之上，为基于 Web 的应用程序提供了上下文。所以，Spring 框架支持与 Jakarta Struts 的集成。Web 模块还简化了处理多部分请求以及将请求参数绑定到域对象的工作。

Spring MVC ： MVC 框架是一个全功能的构建 Web 应用程序的 MVC 实现。通过策略接口，MVC 框架变成为高度可配置的，MVC 容纳了大量视图技术，其中包括 JSP、Velocity、Tiles、iText 和 POI。

## (3)Spring的实现

1，采用了IoC使代码对Spring的依赖减少 ,

2，Spring将管理的Bean作为POJO(Plain Old Java Object)进行控制

3，Spring的组件就是普通的Java Bean ,

4，单元测试可以不再依赖容器

5，组件的初始化和管理均由Spring完成 ,

6，配置文件中声明各种关系

# **SpringMVC知识点概述**

## 什么是SpringMVC?

SpringMVC全称是spring web mvc，是Spring框架的一部分,是一个MVC模式的WEB开发框架，使用了MVC架构模式的思想，将web层进行职责解耦。

SpringMVC是基于Spring功能之上添加的Web框架，想用SpringMVC必须先依赖Spring，springmvc仅给spring的表现层提供支持。

## MVC是什么?

MVC（Model-View-Controller，模型－视图－控制器）指把页面、后台的交付分成3层来组成，是一种解决页面代码和后台代码分离的设计思想！

MVC里面的M指的是Model（通常包含bean、dao(mapper)、service）；V指的是View，视图层，视图层主要的技术（JSP、HTML）；C指的是Controller，控制层。控制层不负责具体数据、逻辑的处理和运算，它只负责将Model层的结果返回给对应的视图层去展示。

在JavaWeb阶段， Controller层指的就是Servlet； View层指的就是JSP或者HTML; Model层指的就是bean、dao、service。

在J2EE阶段，Controller层指的就是SpringMVC、Structs1\2； View层不变还是主流的页面展示技术; Model层包括bean、mybatis、service。

## SpringMVC的主要组件

1.前端控制器 DispatcherServlet（不需要程序员开发）

作用：接收请求，处理结果，相当于转发器，有了DispatcherServlet 就减少了其它组件之间的耦合度。

2.处理器映射器 HandleMapping (不需要程序员开发)

作用：根据url找到 handler

3.处理器适配器HandlerAdapter（适配器模式）

注意：在编写Handler的时候要按照HandlerAdapter要求的规则去编写，这样适配器HandlerAdapter才可以 正确的去执行Handler。

4.处理器Handler（需要程序员开发）

作用：处理请求的具体过程。

5.视图解析器 ViewResolver（不需要程序员开发）

进行视图的解析，根据视图逻辑名解析成真正的视图（view）

6.视图View（需要程序员开发jsp）

# **MyBatis知识点概述**

## 什么是MyBatis?

MyBatis是一个优秀的持久层框架，它对jdbc的操作数据库的过程进行封装，使开发者只需要关注SQL本身，而不需要花费精力去处理例如注册驱动、创建connection、创建statement、手动设置参数、结果集检索等jdbc繁杂的过程代码。

MyBatis 是支持普通 SQL 查询，存储过程和高级映射的优秀持久层框架。 MyBatis 消除了几乎所有的 JDBC 代码和参数的手工设置以及对结果集的检索。 MyBatis 可以使用简单的XML 或注解用于配置和原始映射，将接口和 Java 的 POJO（ Plain Old Java Objects，普通的Java 对象）映射成数据库中的记录。

## MyBatis的执行流程

1. MyBatis配置文件，包括MyBatis全局配置文件和MyBatis映射文件，其中全局配置文件配置了数据源、事务等信息；映射文件配置了SQL执行相关的信息。

2.MyBatis通过读取配置文件信息，构造出SqlSessionFactory，即会话工厂。

3.通过SqlSessionFactory，可以创建SqlSession即会话。Mybatis是通过SqlSession来操作数据库的。

4.SqlSession本身不能直接操作数据库，它是通过底层的Executor执行器来操作数据库的。Executor接口有两个实现类，一个是普通执行器，一个是缓存执行器（默认）。

5.Executor执行器要处理的SQL信息是封装到一个底层对象MappedStatement中。该对象包括：SQL语句、输入参数映射信息、输出结果集映射信息。其中输入参数和输出结果的映射类型包括Java的简单类型、HashMap集合对象、POJO对象类型。

## MyBatis的功能架构

1. API接口层：提供给外部使用的接口API，开发人员通过这些本地API来操纵数据库。接口层一接收到调用请求就会调用数据处理层来完成具体的数据处理。
2. 数据处理层：负责具体的SQL查找、SQL解析、SQL执行和执行结果映射处理等。它主要的目的是根据调用的请求完成一次数据库操作。

3.基础支撑层：负责最基础的功能支撑，包括连接管理、事务管理、配置加载和缓存处理，这些都是共用的东西，将他们抽取出来作为最基础的组件。为上层的数据处理层提供最基础的支撑。