**J2EE三大框架介绍**

**Hibernate**是一个开放源代码的对象关系映射框架，它对JDBC进行了非常轻量级的对象封装，使得Java程序员可以随心所欲的使用对象[编程](https://baike.so.com/doc/3390106-3568658.html)思维来操纵[数据库](https://baike.so.com/doc/5447468-5685836.html)。

适用场合： Hibernate可以应用在任何使用JDBC的场合，既可以在Java的客户端程序使用，也可以在Servlet/JSP的Web应用中使用，最具革命意义的是，Hibernate可以在应用EJB的J2EE架构中取代CMP，完成[数据持久化](https://baike.so.com/doc/5030500-5256785.html)的重任。

分析：Hibernate的核心类和接口一共有6个，分别为: Configuration、SessionFactory 、Session、Transaction、Query和Criteria。

Hibernate的常用接口：

Configucation:负责加载Hibernate的配置及映射信息，启动Hibernate,根据连接到数据库的信息来创建SessionFactory对象。

SessionFactory:  初始化Hibernate，创建Session对象。他是线程安全的，通常采用单子模式创建此对象。

Session :负责保存·修改·删除·查询·加载对象，这个session不是线程安全的，使用Threadlocal来得到session（相当于创建了一个副本）。

Transaction:事务处理开启事物的方法，Transaction tx=  session.beginTransaction();  提交事务 tx.commit();

Query和Criteria：Query是执行数据库查询，Query接口包装了一个HQL语句，HQL语句是面向对象的，它引用类名和属性名，而不是表名和字段名。Criteria完全封装了基于字符串形式的查询语句，比Query接口更加面向对象，擅长动态查询。

Hibernate的工作原理：

1.通过Configuration().configure();读取并解析hibernate.cfg.xml配置文件  
2.由hibernate.cfg.xml中的<mapping resource="com/xx/User.hbm.xml"/>读取并解析映射信息  
3.通过config.buildSessionFactory();//创建SessionFactory  
4.sessionFactory.openSession();//打开Sesssion  
5.session.beginTransaction();//创建事务Transation  
6.persistent operate持久化操作  
7.session.getTransaction().commit();//提交事务  
8.关闭Session  
9.关闭SesstionFactory

Hibernate的查询方式：  
Sql、Criteria、object comptosition  
Hql：1、 属性查询 2、 参数查询、命名参数查询 3、 关联查询  
 4、 分页查询 5、 统计函数

如何优化Hibernate：  
1.使用双向一对多关联，不使用单向一对多  
2.灵活使用单向一对多关联  
3.不用一对一，用多对一取代  
4.配置对象缓存，不使用集合缓存  
5.一对多集合使用Bag,多对多集合使用Set  
6. 继承类使用显式多态  
7. 表字段要少，表关联不要怕多，有二级缓存撑腰

Hibernate查询数据的方式

         (1)导航对象图查询 (2)OID查询 (3)HQL (4)QBC (5)本地SQL

**Spring**是一个轻量级控制反转(IoC)和面向切面(AOP)的容器框架，是由于软件开发的复杂性而创建的。Spring使用的是基本的JavaBean来完成以前只可能由EJB完成的事情。然而，Spring的用途不仅仅限于服务器端的开发。从简单性、可测试性和松耦合性的角度而言，绝大部分Java应用都可以从Spring中受益。

适用场合：任何Java应用

分析：Spring之所以与Struts、Hibernate等单层框架不同，是因为Spring致力于提供一个以统一的、高效的方式构造整个应用，并且可以将单层框架以最佳的组合揉和在一起建立一个连贯的体系。可以说Spring是一个提供了更完善开发环境的一个框架，可以为POJO(Plain Old Java Object)对象提供企业级的服务。Spring 框架中的核心组件只有三个：Core、Context 和 Beans。它们构建起了整个 Spring 的骨骼架构。没有它们就不可能有 AOP、Web 等上层的特性功能。下面也将主要从这三个组件入手分析 Spring。

Spring 框架是一个分层架构，由 7 个定义良好的模块组成。Spring 模块构建在核心容器之上，核心容器定义了创建、配置和管理 bean 的方式，如图 1 所示。组成 Spring 框架的每个模块（或组件）都可以单独存在，或者与其他一个或多个模块联合实现。每个模块的功能如下：

核心容器：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。

Spring 上下文：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 模块：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 框架：Spring为构建Web应用提供了一个功能全面的MVC框架。虽然Spring可以很容易地与其它MVC框架集成，例如Struts，但Spring的MVC框架使用IoC对控制逻辑和业务对象提供了完全的分离。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。

Spring 框架的功能可以用在任何 J2EE 服务器中，大多数功能也适用于不受管理的环境。Spring 的核心要点是：支持不绑定到特定 J2EE 服务的可重用业务和数据访问对象。毫无疑问，这样的对象可以在不同 J2EE 环境 （Web 或 EJB）、独立应用程序、测试环境之间重用。

Spring的工作机制：

1.Spring mvc请所有的请求都提交给DispatcherServlet,它会委托应用系统的其他模块负责负责对请求进行真正的处理工作。  
 2.DispatcherServlet查询一个或多个HandlerMapping,找到处理请求的Controller.  
 3.DispatcherServlet请请求提交到目标Controller  
 4.Controller进行业务逻辑处理后，会返回一个ModelAndView  
 5.Dispathcher查询一个或多个ViewResolver视图解析器,找到ModelAndView对象指定的视图对象  
 6.视图对象负责渲染返回给客户端

**Struts2** 是Struts的下一代产品，是在 struts 1和WebWork的技术基础上进行了合并的全新的Struts 2框架。其全新的Struts 2的[体系结构](https://baike.so.com/doc/6477885-6691588.html)与Struts 1的体系结构差别巨大。Struts 2以WebWork为核心，采用拦截器的机制来处理用户的请求，这样的设计也使得业务逻辑控制器能够与[Servlet](https://baike.so.com/doc/2354665-2490021.html)API完全脱离开，所以Struts 2可以理解为WebWork的更新产品。虽然从Struts 1到Struts 2有着太大的变化，但是相对于WebWork，Struts 2的变化很小。

Struts2框架改善的地方：核心基于Filter，流程可以动态扩展，多例创建对象，支持多种视图展现技术（JSP，Freemarker，Volicity）

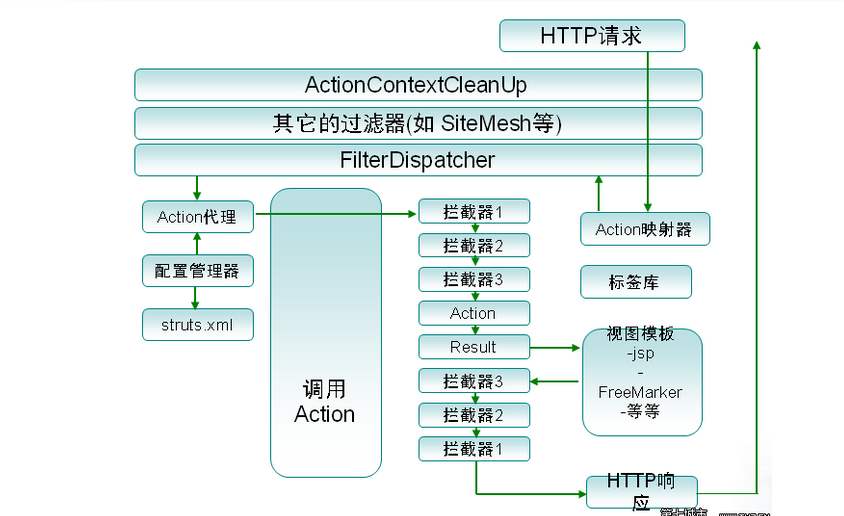
Struts2的体系结构：当Web容器收到 请求(HttpServletRequest)它将请求传递给一个标准的的过滤链包括(ActionContextCleanUp)过滤器，然后经过Other filters(SiteMesh ,etc)，接下来需要调用FilterDispatcher核心控制器，然后它调用ActionMapper确定请求哪个Action，ActionMapper返回一个收集Action详细信息的ActionMaping对象。接下来FilterDispatcher将控制权委派给ActionProxy,ActionProxy调用配置管理器(ConfigurationManager) 从配置文件中读取配置信息(struts.xml)，然后创建ActionInvocation对象，ActionInvocation在调用Action之前会依次的调用所用配置拦截器(Interceptor N) 一旦执行结果返回结果字符串ActionInvocation负责查找结果字符串对应的(Result)然后执行这个Result Result会调用一些模版(JSP)来呈现页面，之后拦截器(Interceptor N)会在被执行(顺序和Action执行之前相反)最后响应(HttpServletResponse)被返回在web.xml中配置的那些过滤器和(核心控制器)(FilterDispatcher)。

Struts的设计模式：

MVC模式: web应用程序启动时就会加载并初始化ActionServler。用户提交表单时，一个配置好的ActionForm对象被创建，并被填入表单相应的数据，ActionServler根据Struts-config.xml文件配置好的设置决定是否需要表单验证，如果需要就调用ActionForm的 Validate（）验证后选择将请求发送到哪个Action，如果Action不存在，ActionServlet会先创建这个对象，然后调用 Action的execute（）方法。Execute（）从ActionForm对象中获取数据，完成业务逻辑，返回一个ActionForward对象，ActionServlet再把客户请求转发给ActionForward对象指定的jsp组件，ActionForward对象指定的jsp生成动态的网页，返回给客户。

Struts2的工作原理：

Struts 2框架本身大致可以分为3个部分：核心控制器FilterDispatcher、业务控制器Action和用户实现的企业业务逻辑组件。核心控制器FilterDispatcher是Struts 2框架的基础，包含了框架内部的控制流程和处理机制。业务控制器Action和业务逻辑组件是需要用户来自己实现的。用户在开发Action和业务逻辑组件的同时，还需要编写相关的配置文件，供核心控制器FilterDispatcher来使用。 Struts 2的工作流程相对于Struts 1要简单，与WebWork框架基本相同，所以说Struts 2是WebWork的升级版本。基本简要流程如下：1、客户端浏览器发出HTTP请求。2、根据web.xml配置，该请求被FilterDispatcher接收。3、根据struts.xml配置，找到需要调用的Action类和方法， 并通过IoC方式，将值注入给Aciton。4、Action调用业务逻辑组件处理业务逻辑，这一步包含表单验证。5、Action执行完毕，根据struts.xml中的配置找到对应的返回结果result，并跳转到相应页面。6、返回HTTP响应到客户端浏览器。



**MVC框架**

**数据 — Model 页面展示 —View  处理具体业务的场所 — Control**

**MVC**全名是Model View Controller，是模型(model)-视图(view)-控制器(controller)的缩写，一种软件设计典范，用一种业务逻辑、数据、界面显示分离的方法组织代码，将业务逻辑聚集到一个部件里面，在改进和个性化定制界面及用户交互的同时，不需要重新编写业务逻辑。MVC被独特的发展起来用于映射传统的输入、处理和输出功能在一个逻辑的图形化用户界面的结构中。MVC是一个框架模式，它强制性的使应用程序的输入、处理和[输出](https://baike.so.com/doc/6782547-6999006.html)分开。使用MVC应用程序被分成三个核心部件:模型、[视图](https://baike.so.com/doc/2478032-2619020.html)、控制器。它们各自处理自己的任务。最典型的MVC就是JSP + [servlet](https://baike.so.com/doc/2354665-2490021.html) + [javabean](https://baike.so.com/doc/2778481-2932635.html)的模式。

MVC 是一种使用 MVC(Model View Controller 模型-视图-控制器)设计创建 Web 应用程序的模式:Model(模型)表示应用程序核心(比如数据库记录列表)。View(视图)显示数据(数据库记录)。Controller(控制器)处理输入(写入数据库记录)。MVC 模式同时提供了对 HTML、CSS 和 JavaScript 的完全控制。

MVC的设计思想

Model(模型)是应用程序中用于处理应用程序数据逻辑的部分。

通常模型对象负责在数据库中存取数据。

模型表示企业数据和业务规则。在MVC的三个部件中，模型拥有最多的处理任务。例如它可能用像EJBs和ColdFusion Components这样的构件对象来处理数据库，被模型返回的数据是中立的，就是说模型与数据格式无关，这样一个模型能为多个视图提供数据，由于应用于模型的代码只需写一次就可以被多个视图重用，所以减少了代码的重复性。

View(视图)是应用程序中处理数据显示的部分。

通常视图是依据模型数据创建的。

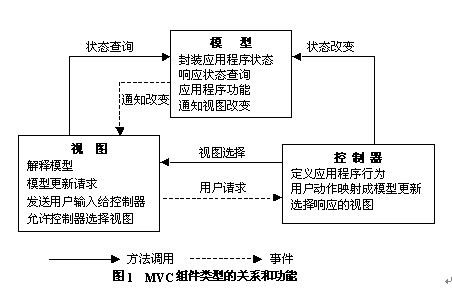
视图是用户看到并与之交互的界面。对老式的Web应用程序来说，视图就是由HTML元素组成的界面，在新式的Web应用程序中，HTML依旧在视图中扮演着重要的角色，但一些新的技术已层出不穷，它们包括[Adobe Flash](https://baike.so.com/doc/5364122-5599712.html)和像[XHTML](https://baike.so.com/doc/1038070-1097948.html)，XML/[XSL](https://baike.so.com/doc/6727969-6942245.html),[WML](https://baike.so.com/doc/5508541-5744287.html)等一些标识语言和[Web services](https://baike.so.com/doc/5507166-5742912.html). MVC设计模式对于视图的处理仅限于视图上数据的采集和处理，以及用户的请求，而不包括在视图上的业务流程的处理。业务流程的处理交予模型(Model)处理。比如一个订单的视图只接受来自模型的数据并显示给用户，以及将用户界面的输入数据和请求传递给控制和模型。

MVC好处是它能为应用程序处理很多不同的[视图](https://baike.so.com/doc/2478032-2619020.html)。在视图中其实没有真正的处理发生，不管这些数据是联机存储的还是一个雇员列表，作为视图来讲，它只是作为一种输出数据并允许用户操纵的方式。

Controller(控制器)是应用程序中处理用户交互的部分。

通常控制器负责从视图读取数据，控制用户输入，并向模型发送数据。

控制器接受用户的输入并调用模型和视图去完成用户的需求，所以当单击Web页面中的超链接和发送[HTML表单](https://baike.so.com/doc/3664673-3851694.html)时，控制器本身不输出任何东西和做任何处理。它只是接收请求并决定调用哪个模型构件去处理请求，然后再确定用哪个视图来显示返回的数据。



MVC好处是它能为应用程序处理很多不同的[视图](https://baike.so.com/doc/2478032-2619020.html)。在视图中其实没有真正的处理发生，不管这些数据是联机存储的还是一个雇员列表，作为视图来讲，它只是作为一种输出数据并允许用户操纵的方式。

总结：模型、视图与控制器的分离，使得一个模型可以具有多个显示视图。如果用户通过某个视图的控制器改变了模型的数据，所有其它依赖于这些数据的视图都应反映到这些变化。因此，无论何时发生了何种数据变化，控制器都会将变化通知所有的视图，导致显示的更新。这实际上是一种模型的变化-传播机制。模型、视图、控制器三者之间的关系和各自的主要功能，