1. **hibernate工作原理**

1.读取并解析配置

(通过Configuration().configure();读取并解析hibernate.cfg.xml[配置文件](http://baike.baidu.com/view/2117618.htm))  
2.读取并解析映射信息，创建Session Factory

(通过config.buildSessionFactory()得到sessionFactory。)

3.打开Session   
4.创建事务Transation   
5.持久化操作   
6.提交事务   
7.关闭Session  
8.关闭SesstionFactory

**Hibernate的运行过程如下**：

A:应用程序先调用Configration类，该类读取Hibernate的配置文件及映射文件中的信息，并用这些信息生成一个SessionFactory对象。

B:然后从SessionFactory对象生成一个Session对象，并用Session对象生成Transaction对象;可通过Session对象的get(),load(),save(),update(),delete()和saveOrUpdate()等方法对PO进行加载，保存，更新，删除等操作;在查询的情况下，可通过Session对象生成一个Query对象，然后利用Query对象执行查询操作;如果没有异常，Transaction对象将 提交这些操作结果到数据库中。

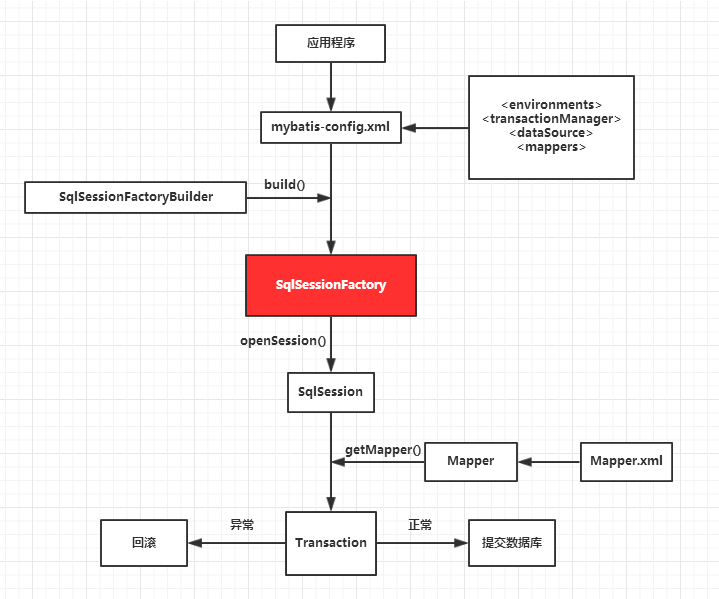
持久化对象可以是普通的Javabeans,惟一特殊的是它们与（仅一个）Session相关联。JavaBeans在Hibernate中存在三种状态：

1.临时状态(transient):当一个JavaBean对象在内存中孤立存在，不与数据库中的数据有任何关联关系时，那么这个JavaBeans对象就称为临时对象(TransientObject)。

2.持久化状态(persistent):当一个JavaBean对象与一个Session相关联时，就变成持久化对象(PersistentObject)

3.脱管状态(detached):在这个Session被关闭的同时，这个对象也会脱离持久化状态，就变成脱管状态(DetachedObject)，可以被应用程序的任何层自由使用，例如可以做与表示层打交道的数据舆对象(Data Transfer Object)。

1. **mybatis工作原理**



**工作原理解析**

mybatis应用程序通过SqlSessionFactoryBuilder从mybatis-config.xml配置文件（也可以用Java文件配置的方式，需要添加@Configuration）中构建出SqlSessionFactory（SqlSessionFactory是线程安全的）；

然后，SqlSessionFactory的实例直接开启一个SqlSession，再通过SqlSession实例获得Mapper对象并运行Mapper映射的SQL语句，完成对数据库的CRUD和事务提交，之后关闭SqlSession。

说明：SqlSession是单线程对象，因为它是非线程安全的，是持久化操作的独享对象，类似jdbc中的Connection，底层就封装了jdbc连接。

**详细流程如下：**

1、加载mybatis全局配置文件（数据源、mapper映射文件等），解析配置文件，MyBatis基于XML配置文件生成Configuration，和一个个MappedStatement（包括了参数映射配置、动态SQL语句、结果映射配置），其对应着<select | update | delete | insert>标签项。

2、SqlSessionFactoryBuilder通过Configuration对象生成SqlSessionFactory，用来开启SqlSession。

3、SqlSession对象完成和数据库的交互：

a、用户程序调用mybatis接口层api（即Mapper接口中的方法）

b、SqlSession通过调用api的Statement ID找到对应的MappedStatement对象

c、通过Executor（负责动态SQL的生成和查询缓存的维护）将MappedStatement对象进行解析，sql参数转化、动态sql拼接，生成jdbc Statement对象

d、JDBC执行sql。

e、借助MappedStatement中的结果映射关系，将返回结果转化成HashMap、JavaBean等存储结构并返回。

1. **Struts2工作原理**

**其实用最简单的七个步骤**：

（1）客户端在浏览器中输入一个url地址；

（2）这个url请求通过http协议发送给tomcat；

（3）Tomcat根据url找到对应项目里面的web.xml文件；

（4）在web.xml里面会发现有struts2的配置；

（5）然后会找到struts2对应的struts.xml配置文件；

（6）根据url解析struts.xml配置文件就会找到对应的class；

（7）调用完class返回一个结果result，根据struts.xml返回到对应的jsp。

**复杂的描述：**

1.用户发起请求（.action结尾的请求表示请求一个Action）。

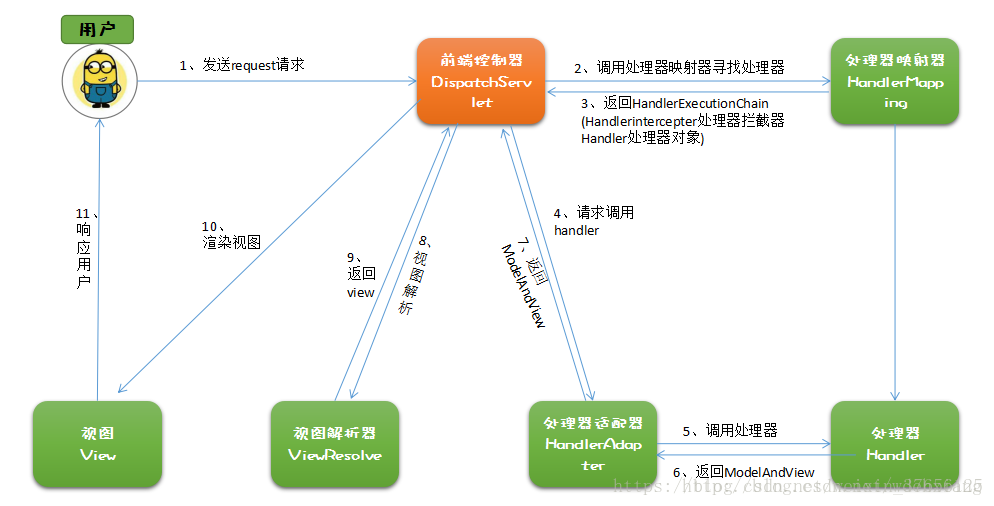
2.Struts2的核心过滤器（StrutsPrepareAndExecuteFilter）接收用户发起的请求，然后判断这个请求交给Action处理还是交给Web组件处理。如果请求的Action或者Web组件不存在，那么 404错误。在整个处理过程中需要一个辅助对象：Action映射器（ActionMapper）。

3.第二步判断的结果如果是交给Action处理，并且存在对应的Action，那么根据struts.xml中对此Action的配置信息，首先执行拦截此Action的所有拦截器，然后再执行请求的Action对象。在这个处理过程中需要辅助对象：Action代理（ActionProxy）、配置管理器（ConfigurationManager）、Action调度（ActionInvocation）。

4.Action执行完毕后返回一个结果（此结果用字符’’’’’’’’’’’’’’’’串表示），这个结果经过拦截Action的所有拦截器之后，返回给核心过滤器（FilterDispatcher），核心过滤器根据此结果从配置文件中找到真正的路径，然后将请求转发给对应的视图，由视图向客户端作出响应。

1. **SpringMVC工作原理**
2. 首先，我们先来认识一下SpringMVC的主要组件  
   前端控制器（DisatcherServlet）:接收请求，响应结果，返回可以是json,String等数据类型，也可以是页面（Model）。  
   处理器映射器（HandlerMapping）:根据URL去查找处理器，一般通过xml配置或者注解进行查找。  
   处理器（Handler）：就是我们常说的controller控制器啦，由程序员编写。  
   处理器适配器（HandlerAdapter）:可以将处理器包装成适配器，这样就可以支持多种类型的处理器。  
   视图解析器（ViewResovler）:进行视图解析，返回view对象（常见的有JSP,FreeMark等）。
3. 工作原理

这里写图片描述



下面是文字步骤说明：  
1、用户发送请求到前端控制器（DispatcherServlet）。  
2、前端控制器请求处理器映射器（HandlerMapping）去查找处理器（Handler）。  
3、找到以后处理器映射器（HandlerMappering）向前端控制器返回执行链（HandlerExecutionChain）。  
4、前端控制器（DispatcherServlet）调用处理器适配器（HandlerAdapter）去执行处理器（Handler）。  
5、处理器适配器去执行Handler。  
6、处理器执行完给处理器适配器返回ModelAndView。  
7、处理器适配器向前端控制器返回ModelAndView。  
8、前端控制器请求视图解析器（ViewResolver）去进行视图解析。  
9、视图解析器向前端控制器返回View。  
10、前端控制器对视图进行渲染。  
11、前端控制器向用户响应结果。