### 总结：

Struts、Spring、Hibernate三者都是开源框架，框架越来越大，功能也越来越多，有一些没有遇到过的坑肯定还有很多，遇到了再解决了。官网上有很多说明，但是有些东西不熟悉找起来很麻烦，如果遇到难办的坑，就到官网上找答案吧，其实官网文档一般都有写到（虽然可能有点难找），再不行就需要去阅读源代码了。而且现在SSH框架慢慢用得少了，SSM可能更多，因为SSM更加轻量级。

### 注意事项：

如果觉得网页有问题就清理一下浏览器的缓存数据和tomcat的缓存数据（右击tomcat然后clean就可以了）。

tomcat下html文件加载js和css等，用相对路径的话需要加上Project名，否则路径不对找不到，使用chrome的开发者工具即可看出。

### 环境：

Windows10+JDK1.8+Tomcat8+Eclipse Oxygen JavaEE+MySQL57

### Struts框架（拦截器和标签）

* Struts下载：<http://struts.apache.org> Struts官网的下载不难找，下一个all和min-lib即可。
* 使用Eclipse Oxygen JavaEE创建Dynamic Web项目，需要勾选web.xml，配置tomcat8和jdk8，注意eclipse版本、tomcat版本、jdk版本可能会有兼容问题，需要对应版本。
* 教程：<http://www.yiibai.com/struts_2/>

入门例子：<http://www.cnblogs.com/qulianqing/p/6627746.html>

* 导入min-lib下的jar文件到WebContent/WEB\_INF/lib下：

commons-fileupload-1.3.3.jar

commons-io-2.5.jar

commons-lang3-3.6.jar

freemarker-2.3.23.jar

javassist-3.20.0-GA.jar

log4j-api-2.8.2.jar

ognl-3.1.15.jar

struts2-core-2.5.13.jar

* 配置web.xml加入Struts的<filter>和<filter-mapping>配置解析的类路径和匹配模式，这里解析为struts的， web.xml中的filter和filter-mapping中的org.apache.struts2.dispatcher.filter包下含4个类，一般用一个StrutsPrepareAndExecuteFilter即可。如果出现直接用浏览器浏览html文件，但是eclipse中用tomcat执行却又乱码可以在web.xml中<mime-mapping>来配置打开文件的方式以及编码。
* 创建struts.xml，需要在src文件夹下，如果struts.xml文件配置出现错误，那么解析时出错，就会包404 not found错误。
* struts.xml中devMode为true时对应为开发模式devMode，而为false是为产品模式proMode，devMode会提供日志和debug等信息，并且开发模式下有：

1、每次请求都会重新加载资源文件，所以当对properties文件进行修改后，下一次请求是就能够反映相应的变化。（struts.i18n.reload = true也可以达到相同的效果）

2、每次请求的时候也会重新加载xml配置文件、验证文件等，这样可以方便调试或者修改相应的配置文件，而不用重新部署重启web服务器。(struts.configuration.xml.reload = true 也可以达到相同的效果)

* struts.xml中package的namespace可以指定访问路径中加一个namespace，且name不能重名，但是此时action中的html和jsp等文件的路径也相较于项目路径多了一层namspace，extends一般要继承自struts-default。
* 创建Action：

方式1：普通的java类中定义public String execute()方法返回"success",然后在struts.xml中声明action，其class需要包名加类名，execute为默认的method，当然method也可以指定到其他方法上，name是路径访问时的名字。

方式2：java类实现Action接口，并且重写execute，然后在struts.xml中配置，其实和方式1没啥区别，只能能够直接返回SUCCESS而不用自己写“success”了而已，可指定method。

方式3：java类继承自ActionSupport，然后重写execute，然后在struts.xml中配置，和方式2没啥区别，也可指定method。

方式4：使用注解的方式，不用再struts.xml中配置。需要导入struts2-convention-plugin-2.5.13.jar以及asm-5.2.jar、asm-commons-5.2.jar、asm-tree-5.2.jar包，java类继承ActionSupport，然后在方法前添加注解如@Action(value="test41", results={@Result(name="success", location="/index.html")})即可访问，但是注解方式和配置方式只能使用其中一种，而且注解的优先，配置的显示The origin server did not find a current representation for the target resource or is not willing to disclose that one exists.注意：导入了注解的包之后，默认的execute方法对应的action name就会显示没有设置result success，但是类中其余方法action还是正常的。

* Action返回json：

配置struts.xml方式：需要导入struts2-json-plugin-2.5.13.jar包，当然具体版本可有不同只要兼容就好，在struts.xml中配置action需要在extends="json-default"的package下，默认会返回所有get方法的属性，不论有没有这个属性。

注解方式：同样需要导入struts2-json-plugin-2.5.13.jar包，类前需要添加注解@ParentPackage("json-default")，方法前添加注解如：@Action(value="actionName",results= {@Result(type="json",params= {"key1","value1","key2","value2"})})，其中value为类中get方法后的value返回值，如果没有，则不返回该key和value。

* Action请求格式设置为json：@InterceptorRef("json")
* **Struts理解：**核心功能都是使用拦截实现“值栈”的概念，OGNL表达式（对象图形导航语言）和Struts2标签来解决应用程序数据，很多注解和约定，使这个框架更加易于使用。值栈：存储机制，类似于域对象，是值栈，可以存值和取值，在action里面把数据放到值栈里面，在页面中获取到值栈的数据。
* **Struts框架特性：**POJO形式和POJO动作；Tag支持；AJAX支持；容易整合（Spring、Tiles、SiteMesh等）；模板支持；插件支持；概要分析（集成调试）；标签修改容易（可用FreeMarker等）；提升更少的配置（默认配置）；

### Spring框架（反转控制IOC也称依赖注入DI，AOP）

* 概念区分：Spring 是一个“引擎” ，Spring MVC 是基于 Spring 的一个 MVC 框架，Spring Boot 是基于 Spring4 的条件注册的一套快速开发整合包。Spring 最初利用“工厂模式”（ DI ）和“代理模式”（ AOP ）解耦应用组件。大家觉得挺好用，于是按照这种模式搞了一个 MVC 框架（一些用 Spring 解耦的组件），用开发 web 应用（ SpringMVC ）。然后有发现每次开发都要搞很多依赖，写很多样板代码很麻烦，于是搞了一些懒人整合包（ starter ），这套就是 Spring Boot。Spring Boot extends SpringMVC extends Spring。
* 下载：spring.io中的Spring FrameWork便是spring，官网有说怎么下载：<http://repo.spring.io/release/org/springframework/spring/> 下载-dist.zip即可。
* 教程：<http://www.yiibai.com/spring/>
* 需要的包：

commons-logging-1.2.jar（下载：http://commons.apache.org/proper/commons-logging/index.html）

spring-aop-4.3.9.RELEASE.jar

spring-aspects-4.3.9.RELEASE.jar

spring-beans-4.3.9.RELEASE.jar

spring-context-4.3.9.RELEASE.jar

spring-context-support-4.3.9.RELEASE.jar

spring-core-4.3.9.RELEASE.jar

spring-expression-4.3.9.RELEASE.jar

spring-instrument-4.3.9.RELEASE.jar

spring-instrument-tomcat-4.3.9.RELEASE.jar

spring-jdbc-4.3.9.RELEASE.jar

spring-jms-4.3.9.RELEASE.jar

spring-orm-4.3.9.RELEASE.jar

spring-oxm-4.3.9.RELEASE.jar

spring-test-4.3.9.RELEASE.jar

spring-tx-4.3.9.RELEASE.jar

spring-web-4.3.9.RELEASE.jar

spring-webmvc-4.3.9.RELEASE.jar

spring-webmvc-portlet-4.3.9.RELEASE.jar

spring-websocket-4.3.9.RELEASE.jar

* Spring的控制反转实际上就是Class用配置成bean的方式来管理，否则每次我们都得自己写清楚，改动时很麻烦且容易出错还耗时。
* 声明为bean的Class可以有两种方式，一种是在applicationContext.xml中配置：<bean id="helloWorld" class="com.pifutan.spring.HelloWorld"></bean>，另一种是使用组件自动扫描。
* 组件自动扫描：使用@Component需要配置applicationContext.xml：

<!-- 启用自动组件扫描 -->

<context:component-scan base-package="com.pifutan.bean" />

实际上有四种方式：

@Component – 指示自动扫描组件。

@Repository – 表示在持久层DAO组件。

@Service – 表示在业务层服务组件。

@Controller – 表示在表示层控制器组件。

但是这四种方式实际上是一样的，且都是开启component-scan就可以了，只是为了方便阅读弄出了四种而已。

* 过滤器组件自动扫描，即不使用如@Component就可以用了。

可以使用包含和不包含两种过滤。

<context:component-scan base-package="com.pifutan" >

<context:include-filter type="regex"

expression="com.pifutan.dao.\*DAO.\*" />

<context:include-filter type="regex"

expression="com.pifutan.service.\*Service.\*" />

<context:exclude-filter type="regex"

expression="com.pifutan.service.\*Service.\*" />

</context:component-scan>

* List，Set，Map，Properties和Date注入，都有相应支持和使用。
* 在Spring中，bean作用域用于确定哪种类型的 bean 实例应该从Spring容器中如何返回给调用者。bean支持的5种范围域scope：

单例singleton - 每个Spring IoC 容器返回同一个bean实例

原型prototype - 当每次请求时返回一个新的bean实例

请求request - 返回每个HTTP请求的一个Bean实例

会话session - 返回每个HTTP会话的一个bean实例

全局会话global session - 返回全局HTTP会话的一个bean实例

在大多数情况下，可能只处理了 Spring 的核心作用域 - 单例和原型，默认作用域是单例。

注：意味着只有在一个基于web的Spring ApplicationContext情形下有效！

* 如果需要同一个类的两个单例，则可以声明不同的两个beanid，使用@Qualifier("personA")自动装配。
* 声明IUser接口然后User和Admin都实现IUser接口，并且User和Admin都在xml中配置bean或者在类@Component(“beanid”)，使用的时候声明IUser user则会匹配User，声明IUser admin则会匹配Admin，如果找不到beanid且接口只有一个实现时能匹配，如果有多个实现则会报错多个匹配。如果是使用过滤器自动扫描组件，则类名和变量名对应即可（大小写无区别）。

当然IUser需要加@Autowired自动装配，[使用@Autowired需要在applicationContext.xml](mailto:使用@Autowired需要在applicationContext.xml)中添加<context:annotation-config />

* 配置bean使用new方式实例化是无法填充内部的类变量的。需要：

ApplicationContext context = **new** ClassPathXmlApplicationContext( "applicationContext.xml");(User) context2.getBean("user");

* Spring EL表达式，就是可以定义按一定条件和形式注入值。用到的时候再详细学吧。
* @Required以及@Mandatory 定制 @Required-style 注解，相当于@Required注解。
* Spring提供了JDBC支持，提供了JdbcTemplate和JdbcDaoSupport以及SimpleJdbcTemplate。
* Spring AOP：Advice（指示之前或方法执行后采取的行动），Pointcut（指明哪些方法应该拦截，通过方法的名称或正则表达式模式），Advisor（分组"通知"和”切入点“成为一个单元，并把它传递到代理工厂对象）。
* Spring AOP+ AspectJ：注意Spring AOP 中没有 AspectJ 支持

常见AspectJ的注解：

@Before – 方法执行前运行

@After – 运行在方法返回结果后

@AfterReturning – 运行在方法返回一个结果后，在拦截器返回结果。

@AfterThrowing – 运行方法在抛出异常后，

@Around – 围绕方法执行运行，结合以上这三个通知。

### Hibernate框架（ORM(对象关系映射)以及事务）

* Hibernate框架简化了java应用程序与数据库交互的开发。 Hibernate是一个开源，轻量级的ORM(对象关系映射)工具。

ORM工具简化了数据创建，数据处理和数据访问。它是将对象映射到数据库中存储的数据(表)的编程技术。

优点：开源和轻量级，快速性能，数据库独立查询，自动创建表，简化复杂连接，提供查询统计和数据库状态。

Hibernate架构包括许多对象持久对象，会话工厂，事务工厂，连接工厂，会话，事务等。Hibernate架构中有4层Java应用层，hibernate框架层，反手api层和数据库层。

* 下载：<http://hibernate.org/orm/releases/> MySQLConnector下载：<https://dev.mysql.com/downloads/connector/j/>
* 教程：<http://www.yiibai.com/hibernate/>
* 需要的包：

mysql-connector-java-5.1.44-bin.jar

antlr-2.7.7.jar

classmate-1.3.0.jar

dom4j-1.6.1.jar

hibernate-commons-annotations-5.0.1.Final.jar

hibernate-core-5.2.11.Final.jar

hibernate-jpa-2.1-api-1.0.0.Final.jar

jandex-2.0.3.Final.jar

javassist-3.20.0-GA.jar

jboss-logging-3.3.0.Final.jar

jboss-transaction-api\_1.2\_spec-1.0.1.Final.jar

* 首先在MySQL中创建相应的表，在代码中创建相应的实体类，可以用table.hbm.xml或者用注解的方式来映射数据库中的表，使用javax.persistence.下的@Entity@Table(name="tablename")@Id。
* Hibernate生成器类，包括assigned，increment，sequence，hilo，native，identity，seqhilo，uuid，guid，select，foreign，sequence-identity。
* 使用Log4j日志，使用log4j.xml或log4j.properties，下载地址：<https://logging.apache.org/log4j/>
* 集合映射，使用xml。
* HQL：

查询，它是一个面向对象的Hibernate Query表示。 Query的对象可以通过Session接口调用createQuery()方法。

查询接口提供了很多方法。下面给出了一些最常用的方法：

public int executeUpdate() 用于执行更新或删除查询。

public List list() 将关系的结果作为列表返回。

public Query setFirstResult(int rowno) 指定从哪里检索记录的行号。

public Query setMaxResult(int rowno) 指定从关系(表)中检索记录的行号。

public Query setParameter(int position, Object value) 它将该值设置为JDBC样式查询参数。

public Query setParameter(String name, Object value) 它将该值设置为命名查询参数。

获取所有记录示例：

Query query=session.createQuery("from Emp");//here persistent class name is Emp

List list=query.list();

获取分页记录示例：

Query query=session.createQuery("from Emp");

query.setFirstResult(5);

query.setMaxResult(10);

List list=query.list();//will return the records from 5 to 10th number

更新查询示例：

Transaction tx=session.beginTransaction();

Query q=session.createQuery("update User set name=:n where id=:i");

q.setParameter("n","Udit Kumar");

q.setParameter("i",111);

int status=q.executeUpdate();

System.out.println(status);

tx.commit();

删除示例：

Query query=session.createQuery("delete from Emp where id=100");

//specifying class name (Emp) not tablename

query.executeUpdate();

聚合函数：avg()，min()，max()

Query q=session.createQuery("select sum(salary) from Emp");

List<Integer> list=q.list();

System.out.println(list.get(0));

* HCQL：Hibernate标准(Criteria)查询语言(HCQL)用于根据具体条件获取记录。

session接口的createCriteria()方法的语法：

常用的Criteria接口方法如下：

public Criteria add(Criterion c) 用于添加限制(条件)。

public Criteria addOrder(Order o) 指定排序顺序。

public Criteria setFirstResult(int firstResult) 指定要检索的第一个记录数。

public Criteria setMaxResult(int totalResult) 指定要检索的记录总数。

public List list() 返回包含对象的列表。

public Criteria setProjection(Projection projection) 指定投影。

Restrictions类：

public static SimpleExpression lt(String propertyName,Object value) 将给定属性的约束设置为小于约束。

public static SimpleExpression le(String propertyName,Object value) 设置给定属性的小于或等于约束。

public static SimpleExpression gt(String propertyName,Object value) 设置给定属性的大于约束。

public static SimpleExpression ge(String propertyName,Object value) 设置给定属性的大于或等于约束。

public static SimpleExpression ne(String propertyName,Object value) 对给定的属性设置不相于约束。

public static SimpleExpression eq(String propertyName,Object value) 设置约束与给定属性相等。

public static Criterion between(String propertyName, Object low, Object high) 设置约束之间范围。

public static SimpleExpression like(String propertyName, Object value) 将类似的约束设置为给定的属性。

Order类：Restrictions 类方法

public static Order asc(String propertyName) 适用于给定属性的基础上，按升序排列。

public static Order desc(String propertyName) 适用于给定属性的基础上，按降序排列。

* Hibernate缓存：一级缓存和二级缓存。Session对象保存第一级缓存数据，第一级缓存默认情况下启用。 第一级缓存数据将不可用于整个应用程序。这是因为应用程序可以使用很多会话(Session)对象。SessionFactory对象保存二级缓存数据。 存储在第二级缓存中的数据将可用于整个应用程序。 但是我们需要明确地启用它，这是第二级缓存默认情况下不启用。二级缓存实现由不同的程序开发者(商)提供。

### Struts + Spring + Hibernate框架

* 搭建Struts框架，导入struts包配置web.xml和struts.xml，将spring和hibernate包导入，另外再导入struts2-spring-plugin-2.5.13.jar包，配置applicationContext.xml，不需要配置hibernate.cfg.xml，在web.xml中配置contextConfigLocation为applicationContext.xml的路径且配置Spring监听器，applicationContext.xml需要配置dataSource、sessionFactory（packagesToScan的property配置扫描的hibernate的表对应的Entity类）、transactionManager、transactionInterceptor（配置一下如save方法权限等）、hibernateTemplate（推荐使用）。
* jar包：

antlr-2.7.7.jar

asm-5.2.jar

asm-commons-5.2.jar

asm-tree-5.2.jar

c3p0-0.9.5.2.jar

classmate-1.3.0.jar

commons-fileupload-1.3.3.jar

commons-io-2.5.jar

commons-lang3-3.6.jar

commons-logging-1.2.jar

dom4j-1.6.1.jar

freemarker-2.3.23.jar

hibernate-c3p0-5.2.11.Final.jar

hibernate-commons-annotations-5.0.1.Final.jar

hibernate-core-5.2.11.Final.jar

hibernate-jpa-2.1-api-1.0.0.Final.jar

jandex-2.0.3.Final.jar

javassist-3.20.0-GA.jar

jboss-logging-3.3.0.Final.jar

jboss-transaction-api\_1.2\_spec-1.0.1.Final.jar

log4j-api-2.8.2.jar

log4j-core-2.9.1.jar

mchange-commons-java-0.2.11.jar

mysql-connector-java-5.1.44-bin.jar

ognl-3.1.15.jar

slf4j-api-1.7.25.jar

spring-aop-4.3.9.RELEASE.jar

spring-aspects-4.3.9.RELEASE.jar

spring-beans-4.3.9.RELEASE.jar

spring-context-4.3.9.RELEASE.jar

spring-context-support-4.3.9.RELEASE.jar

spring-core-4.3.9.RELEASE.jar

spring-expression-4.3.9.RELEASE.jar

spring-instrument-4.3.9.RELEASE.jar

spring-instrument-tomcat-4.3.9.RELEASE.jar

spring-jdbc-4.3.9.RELEASE.jar

spring-jms-4.3.9.RELEASE.jar

spring-orm-4.3.9.RELEASE.jar

spring-oxm-4.3.9.RELEASE.jar

spring-test-4.3.9.RELEASE.jar

spring-tx-4.3.9.RELEASE.jar

spring-web-4.3.9.RELEASE.jar

spring-webmvc-4.3.9.RELEASE.jar

spring-webmvc-portlet-4.3.9.RELEASE.jar

spring-websocket-4.3.9.RELEASE.jar

struts2-convention-plugin-2.5.13.jar

struts2-core-2.5.13.jar

struts2-json-plugin-2.5.13.jar

struts2-spring-plugin-2.5.13.jar

* 写了一个BaseAction，其余Action集成BaseAction，返回json时想要返回父类的属性可以添加一个参数ignoreHierarchy=false，注解方式为：@Result(type="json",params= {"ignoreHierarchy","false"})，xml方式：<param name=”ignoreHierarchy”>false</param>。

类似的属性还有（值有多个时用逗号隔开）：（可参照官网：<http://struts.apache.org/docs/json-plugin.html> ）

excludeProperties, 不包含指定类容的集合

includeProperties, 包含指定类容的集合

excludeNullProperties, 是否转换输出null值

root, 只返回指定的对象

wrapPrefix, 在json结果前添加符号

wrapSuffix, 在json结果后添加符号

ignoreHierarchy, 是否忽略继承关系

enableGZIP, 是否支持gzip压缩输出

noCache, 是否缓存

statusCode, 设置响应代码号

errorCode, 错误代码

callbackParameter, json跨域访问回调设置jsonp

contentType, 输出类型

encoding, 编码