

## 目录

Ssh 快速入门 .....	3
1. 了解原生 JAVA .....	3
1.1 hello world 程序 .....	3
1.2 Java Servlet 版 hello world .....	4
2. 理解几个基本概念 .....	5
2.1 反射 .....	5
2.2 单例（单态） .....	6
2.3 Java 的接口例子 .....	6
3. Struts 简介 .....	9
3.1 MVC .....	9
3.2 Struts 最简单例子 .....	10
3.3 通过 actionForm 对象传值 .....	17
4. Hibernate 简介 .....	22
4.1 什么是 ORM .....	22
4.2 什么是 Hibernate .....	23
4.3 ORM 的实现原理 .....	23
4.4 hibernate 的工作流程范例 .....	24
5. Spring .....	24
5.1 工厂模式 .....	24
5.2 Spring 的 IOC 原理 .....	24
5.3 Spring 的设值注入范例 .....	24
5.4 Spring 的构造注入 .....	29
6. 综合例子（一） .....	29
6.1 ide 中添加 mysql 数据库驱动创建连接 .....	29
6.2 新建工程 .....	32
6.3 导入 spring、struts 和 hibernate 包 .....	34
6.4 通过数据库生成 bean 类 .....	44
6.5 配置 struts，实现 Hello World .....	47
6.6 Spring 注入配置 .....	53
6.7 HQL 读取表数据 .....	67
7. 综合例子（二） .....	80
7.1 增加 actionForm .....	80
7.2 增删改查完整例子 .....	88
7.3 最终代码大全 .....	90
8. 连接数据库乱码问题解决 .....	114
8.1 设置 myeclipse 字符集 .....	114
8.2 修改 myeclipse 的视图和配置文件编码 .....	115
8.3 设置 mysql 的字符集编码 .....	115
8.4 修改数据库连接路径 .....	115
8.5 添加 struts 字符集过滤器 .....	118



# Ssh 快速入门

([soft456@gmail.com](mailto:soft456@gmail.com) 2016-11-27)

Ssh 是指 Spring + Struts + Hibernate

本教程所用版本为: Spring3.1.1 + Struts1.3.8 + Hibernate3.3.2

IDE 为 MyEclipse2014

## 1. 了解原生 JAVA

### 1.1 hello world 程序

任何文本编辑器编写 Hello.java 文件

----- 内容 -----

```
public class Hello{  
    public static void main(String[] args){  
        System.out.println("Hello Java!");  
    }  
}
```

#### 1.1.1 代码分析

```
public class Hello
```

定义了一个类，类名是 Hello; 类的类型是 public （公有公开）

注意:Java 中主类名应该和要保存的 Java 文件名相同,也就是说,这里定义类名是“Hello”,则文件应该保存为“Hello.java”。

```
public static void main(String[] args)
```

Java 中的程序入口方法,这是约定俗成,同 C/C++中的 main()作用一样,就是所有的程序都从 main()方法开始执行。

Java 应用程序有且只有一个 main 方法。

#### 1.1.2 编译 Hello.java 文件

Win7 下运行 cmd——》进入命令行状态——》进入 Hello.java 文件所在目录:

```
管理员: C:\Windows\system32\cmd.exe

E:\javaHello>dir
驱动器 E 中的卷没有标签。
卷的序列号是 5489-6FEF

E:\javaHello 的目录

2016/11/22  10:59    <DIR>          .
2016/11/22  10:59    <DIR>          ..
2016/11/22  10:53                106 Hello.java
               1 个文件             106 字节
               2 个目录 199,968,624,640 可用字节

E:\javaHello>
```

```
管理员: C:\Windows\system32\cmd.exe

E:\javaHello>dir
驱动器 E 中的卷没有标签。
卷的序列号是 5489-6FEF

E:\javaHello 的目录

2016/11/22  10:59    <DIR>          .
2016/11/22  10:59    <DIR>          ..
2016/11/22  10:53                106 Hello.java
               1 个文件             106 字节
               2 个目录 199,968,624,640 可用字节

E:\javaHello>javac Hello.java

E:\javaHello>
```

### 1.1.3 执行编译后的 class 文件

```
E:\javaHello>java Hello
Hello Java!
```

## 1.2 Java Servlet 版 hello world

Servlet 是 java 的 web 原生解决方案，框架 ssh、springMVC 均是基于此的再封装。

## 1.21 建 servlet 工程

new ---> Web project ---> 设置工程名

Project name : servlet (工程名称)

Target runtime: Apache Tomcat v7.0 (刚设置的)

其他项默认 --> 完成

## 1.22 应用发布

新建的工程名 (servlet) 上右键 --> MyEclipse --> Add and Remove

Project Deployment --> Add --> Server 栏选 “Tomcat 7.x” --> Finish --> ok

## 1.23 运行

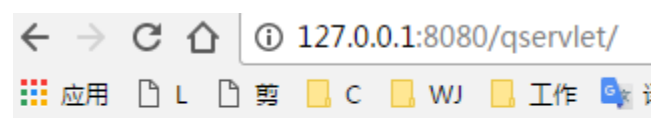
IDE 下方 Servers 标签 --> Tomcat 7.x --> servlet --> 运行

## 1.24 校验

浏览器输入 url:

http://127.0.0.1:8080/servlet

结果如下:



This is my JSP page.

## 2. 理解几个基本概念

### 2.1 反射

简单理解，事先不知道需要调用的类名和方法名，根据传入的变量，动态的初始化类实例，并执行动态的方法。

## 2.2 单例（单态）

同进程里，类之实例化一次，每次调用前看是否在缓存里存在，如果存在，就返回，不存在就先实例化再返回。

参见 php 例子：

```
/**
 * 获取队列实例
 *
 * @return E_Queue_Httpsqs || E_Queue_Redis
 */
private static function _getQueueInc() {
    $regName = "yafext_async_queue_httpsqs";

    if (Yaf_Registry::has($regName)) {
        return Yaf_Registry::get($regName);
    }

    //没有则实例化
    $config = Extconfig::$asyncQueue;
    $adapter = $config['adapter'];
    $class = 'E_Queue_' . ucfirst($adapter);
    $queue = new $class($config);
    Yaf_Registry::set($regName, $queue);
    return $queue;
}
```

## 2.3 Java 的接口例子

用关键字 interface 定义接口类，接口类中只申明方法名、参数和返回值类型  
接口实现类，用 implements 指定接口定义类，需要实现接口定义类申明的所有接口。

### 2.3.1 新建一个一般 java 工程

菜单 → New → Java Project → 取工程名 api，其他选项默认 → Finish

### 2.3.2 接口定义和实现

接口定义类——IPerson.java

----- 内容 -----

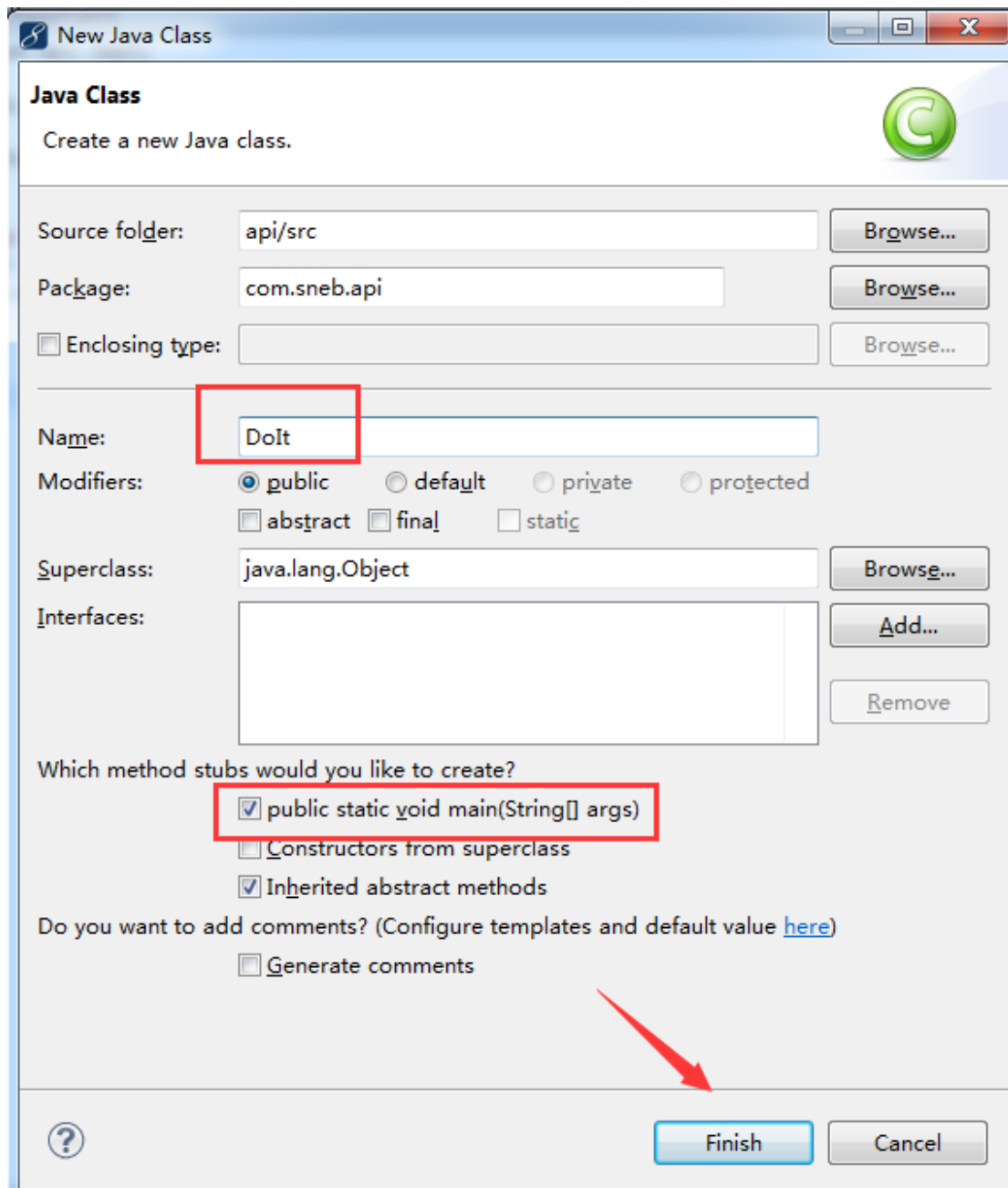
```
package com.sneb.api;
public interface IPerson {
    public void sayHello();
}
----- 内容结束 -----
```

接口实现类——Chinese.java

```
----- 内容 -----
package com.sneb.api.impl;
import com.sneb.api.IPerson;
public class Chinese implements IPerson{
    @Override
    public void sayHello() {
        // TODO Auto-generated method stub
        System.out.println("你好! ");
    }
}
```

### 2.3.3 接口调用

新建一个带 main 方法 class



DoIt.java 完成代码如下:

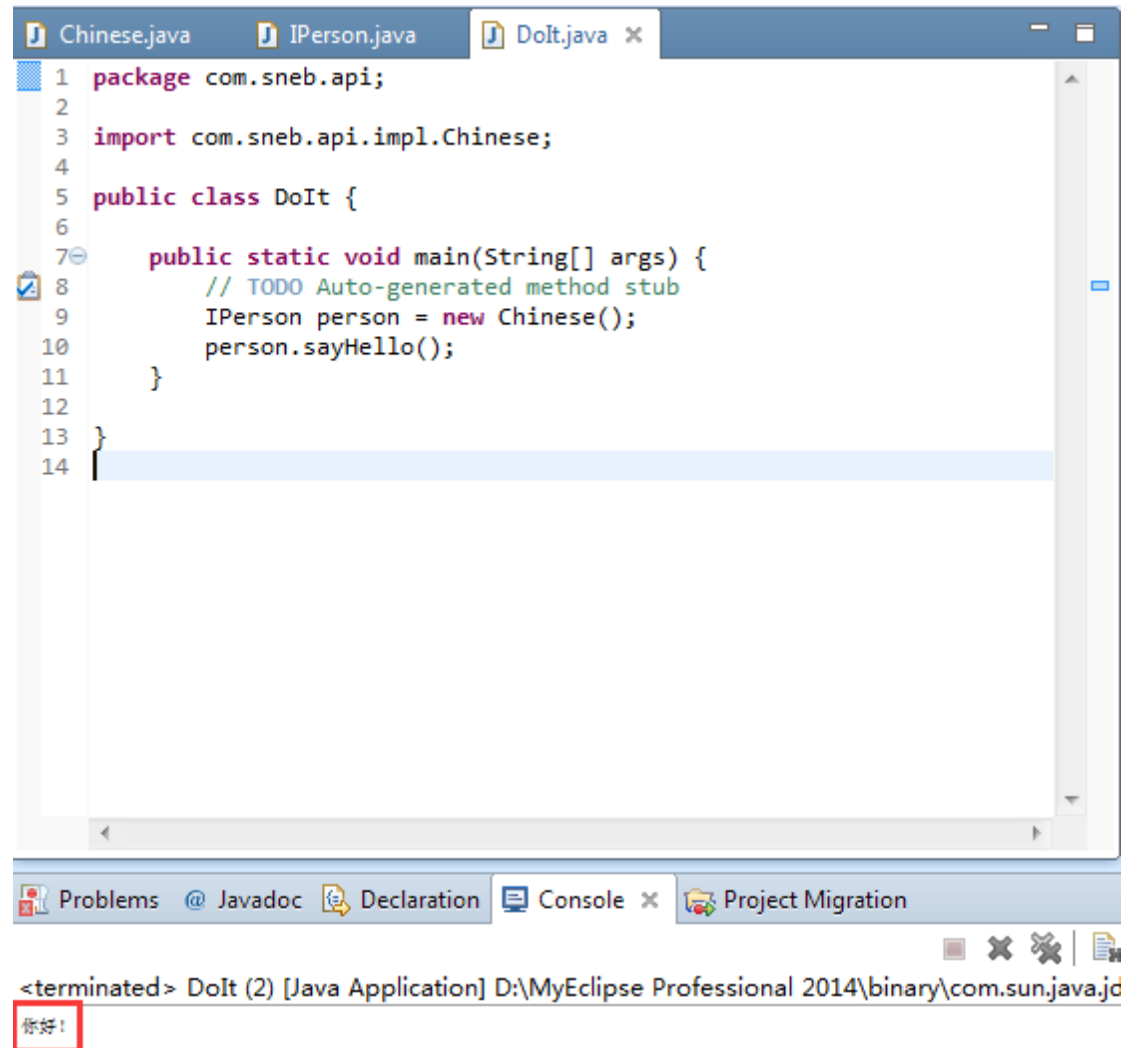
```
----- 内容 -----  
package com.sneb.api;  
import com.sneb.api.impl.Chinese;  
public class DoIt {  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        IPerson person = new Chinese();  
        person.sayHello();  
    }  
}
```



### 2.3.4 运行检验

菜单 → Run → Run As → Java Application

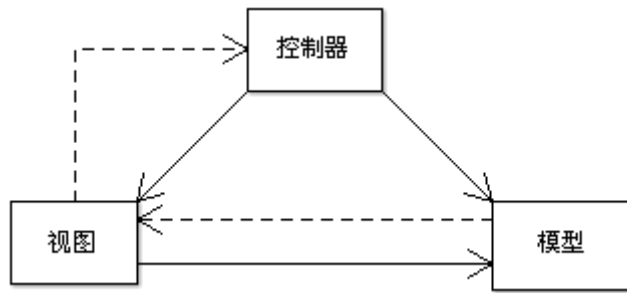
效果截图：



## 3. Struts 简介

### 3.1 MVC

MVC 模式（Model - view - controller）是软件工程中的一种软件架构模式，把软件系统分为三个基本部分：模型（Model）、视图（View）和控制器（Controller）。



控制器（Controller）- 负责转发请求，对请求进行处理。

视图（View）- 界面设计人员进行图形界面设计。

模型（Model）- 程序员编写程序应有的功能（实现业务逻辑）、数据库专家进行数据管理和数据库设计(可以实现具体的功能)。

在 J2EE 应用程序中

视图(View) ——可能由 Java Server Page(JSP)担任。

控制器（Controller） ——J2EE 应用中，Controller 可能是一个 servlet。

也可用其他框架来撰写，常见的有 Struts2、Spring Framework……等等。

模型（Model） —— 由一个实体 Bean 来实现。

## 3.2 Struts 最简单例子

### 3.2.1 新建工程，并导入 struts1 类库

New → Web Project

New Web Project

### Create a JavaEE Web Project

Specify project name and other details

Project name: **myStruts**

Project location

☒ Use default location

Location: E:\java\_work\myStruts [Browse...](#)

Project configuration

Java EE version: JavaEE 6 - Web 3.0

Java version: 1.6

JSTL Version: 1.2.1

☐ Add maven support [Learn more about Maven4MyEclipse...](#)

Target runtime

**Apache Tomcat v7.0** [Add New Runtime...](#)

EAR membership

☐ Add project to an EAR

EAR project name: myStrutsEAR [New Project...](#)

Working sets

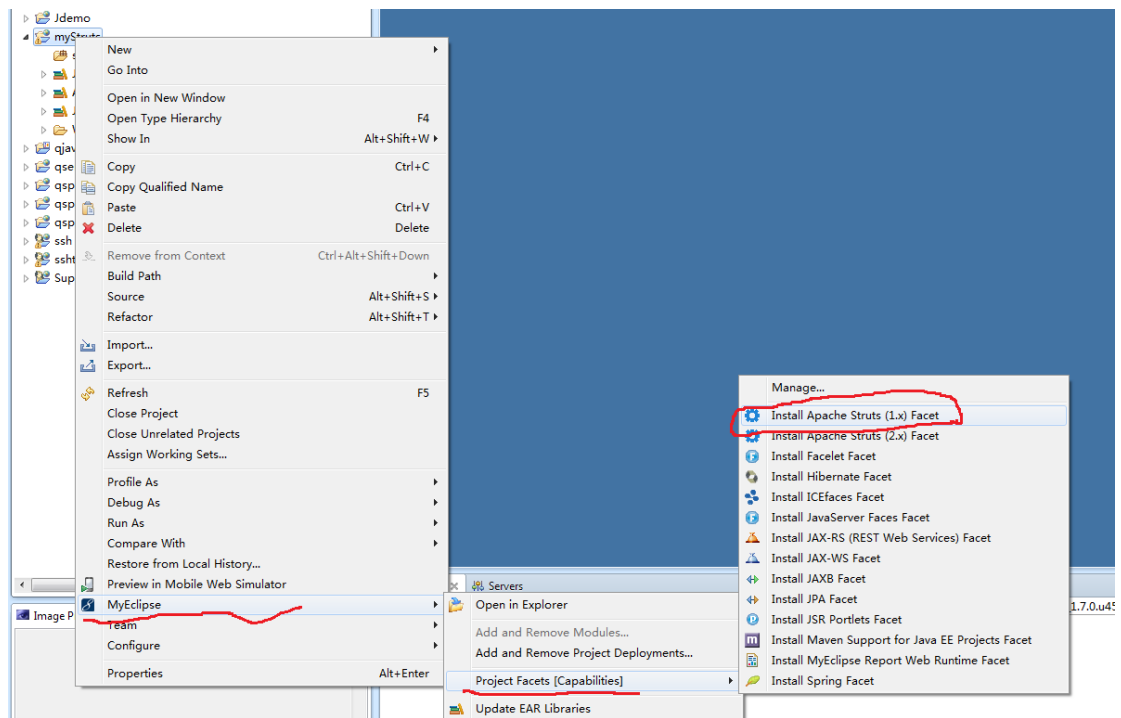
☐ Add project to working sets

Working sets: [Select...](#)

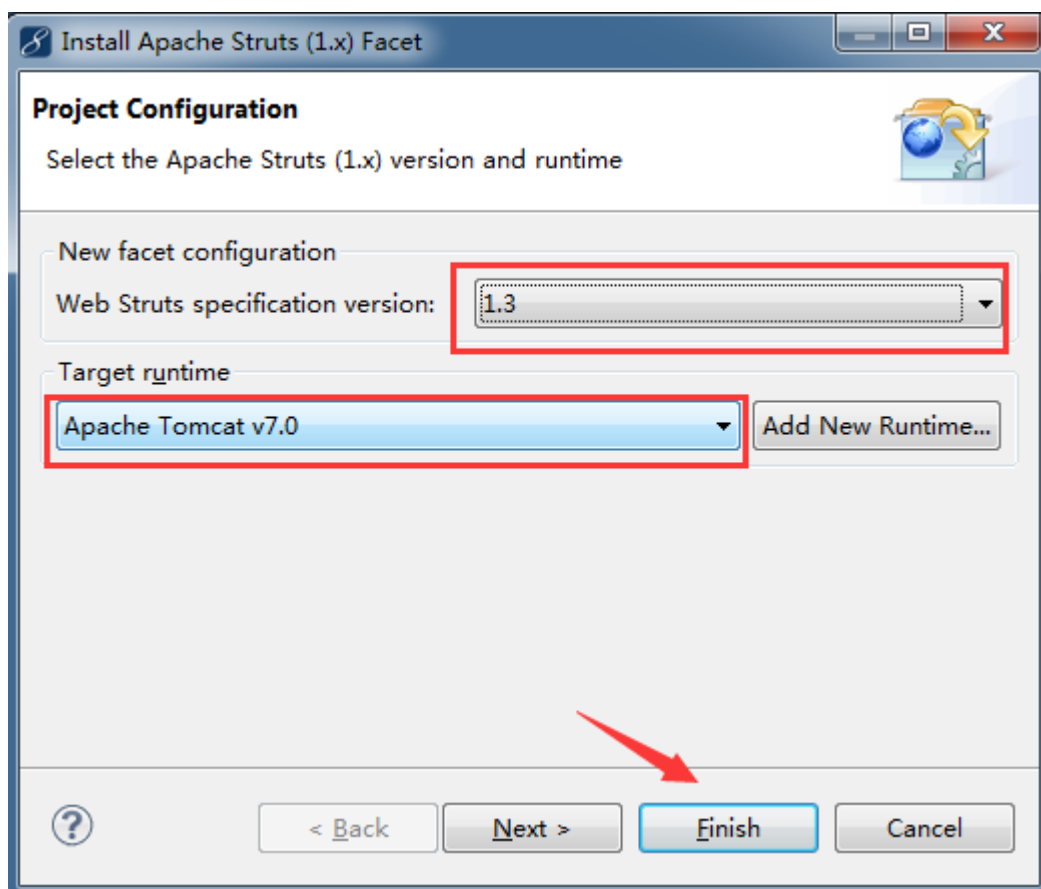
[?](#) [< Back](#) [Next >](#) [Finish](#) [Cancel](#)

导入 struts1.3 类库:

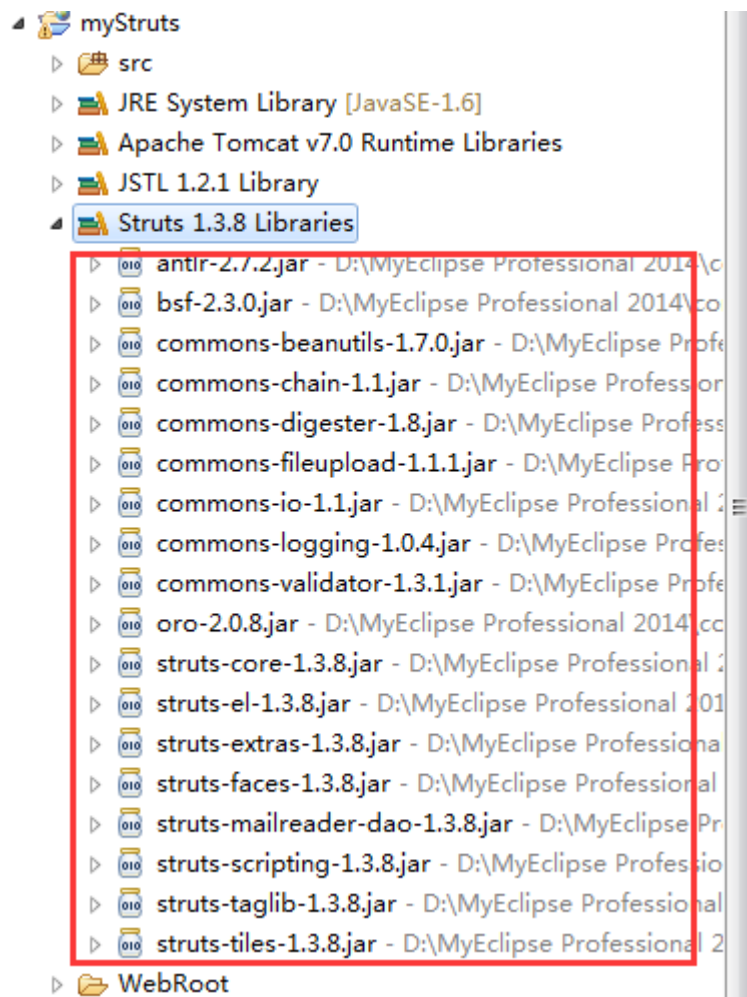
myStruts 工程名上右键 → MyEclipse → Project Facets [Capabilities] → Install Apache Struts(1.x)Facet



接下来如下图：

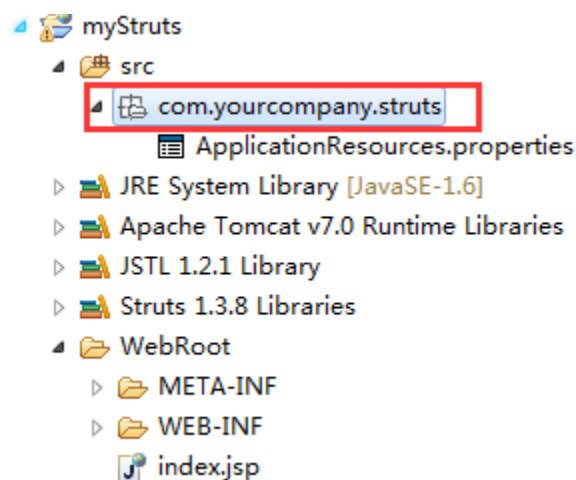


这些操作将帮我们导入 struts 类库到当前工程，可以看到导入的 jar 包如下：



### 3.2.2 修改包名

Java 类的存放路径一般按.分割，如下：



请将 yourcompany 修改成自己公司的简称如：sneb

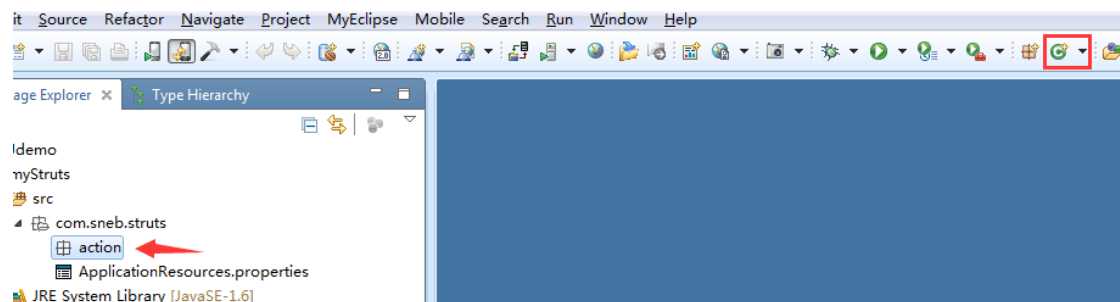
修改方法： 目录名右键 → Refactor → Rename

### 3.2.3 创建控制器类 HelloAction

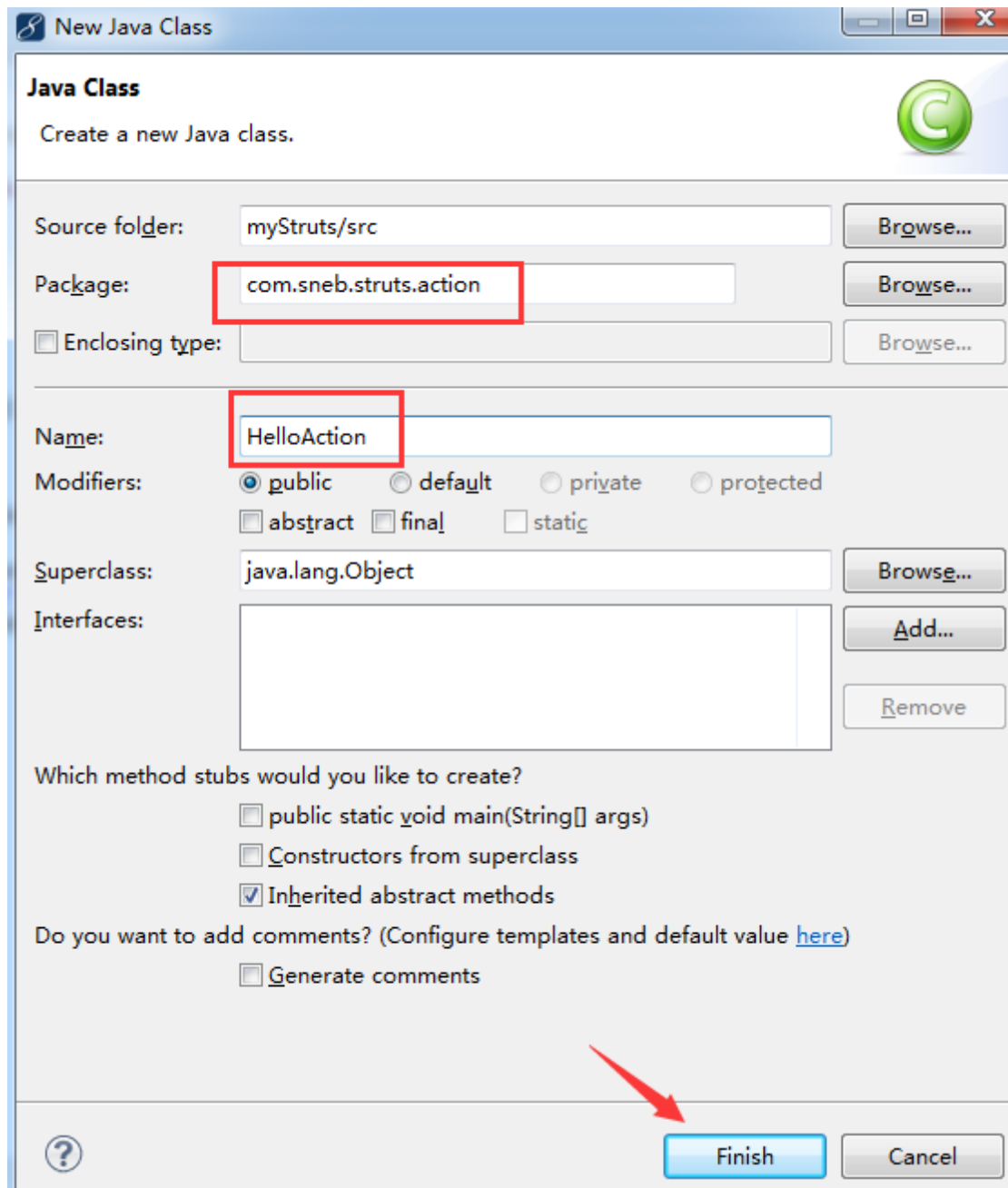
包名下创建新目录“action”用来存放控制器类,操作如下:

com.sneb.struts 上右键 → new → Package

新建控制器的类,可菜单里创建, 也可以直接工具栏快捷方式创建:



点击后如下图:



输入代码如下:

----- 内容 -----

```
package com.sneb.struts.action;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
```

```

public class HelloAction extends Action{

    public ActionForward execute(ActionMapping mapping, ActionForm
    form,HttpServletRequest request, HttpServletResponse response){

        request.setAttribute("msg", "Hello world!");
        return mapping.findForward("show");
    }

}

```

说明： msg 为传值到视图层的变量名； show 是 struts-config.xml 里配置声明的对应 jsp 页面的变量名；

### 3.2.4 创建 jsp 页面显示

直接修改 index.jsp 页面显示 action 中的数据

核心代码如下：

`<%=request.getAttribute("msg") %>` 或者 `${msg}`

截图如下：

```

1  <%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>
2  <%
3  String path = request.getContextPath();
4  String basePath = request.getScheme()+ "://" +request.getServerName()+ ":" +request.getServerPort()+path+"/";
5  %>
6
7  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
8  <html>
9  <head>
10     <base href="<%=basePath%>">
11
12     <title>My JSP 'index.jsp' starting page</title>
13     <meta http-equiv="pragma" content="no-cache">
14     <meta http-equiv="cache-control" content="no-cache">
15     <meta http-equiv="expires" content="0">
16     <meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
17     <meta http-equiv="description" content="This is my page">
18     <!--
19     <link rel="stylesheet" type="text/css" href="styles.css">
20     -->
21 </head>
22
23 <body>
24     <%=request.getAttribute("msg") %>
25 </body>
26 </html>
27

```

### 3.2.5 struts 配置修改

修改 struts 的核心配置文件 struts-config.xml

详细配置介绍见：<http://blog.csdn.net/tang9140/article/details/50559726>

核心配置代码如下：



```

<action-mappings>
    <action path="/hello" scope="request"
        type="com.sneb.struts.action.HelloAction">
        <forward name="show" path="/index.jsp"></forward>
    </action>
</action-mappings>

```

截图:

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE struts-config PUBLIC "-//Apache Software Foundation//DTD Struts Configuration 1.3//EN" "http://struts.apache.org/dtds/struts-config_1_3.dtd">
3
4 <struts-config>
5     <form-beans />
6     <global-exceptions />
7     <global-forwards />
8
9     <action-mappings>
10        <action path="/hello" scope="request"
11            type="com.sneb.struts.action.HelloAction">
12            <forward name="show" path="/index.jsp"></forward>
13        </action>
14    </action-mappings>
15
16    <message-resources parameter="com.yourcompany.struts.ApplicationResources" />
17 </struts-config>

```

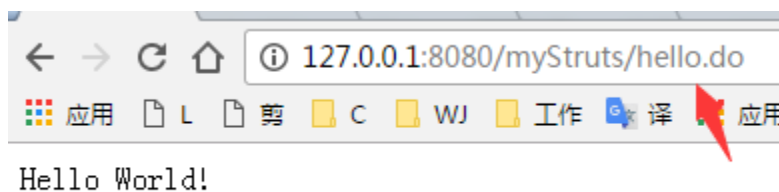
### 3.2.6 发布应用

myStuts 工程上右键 --> MyEclipse --> Add and Remove Project  
Deployment --> Add --> Server 栏选 “Tomcat 7.x” --> Finish --> ok

### 3.2.7 检验

启动应用后，浏览器地址栏输入：

<http://127.0.0.1:8080/myStruts/hello.do>



## 3.3 通过 actionForm 对象传值

### 3.3.1 新建 actionForm 对象

actionForm.java 代码如下：

```

----- 内容 -----
package com.sneb.struts.form;
import org.apache.struts.action.ActionForm;
public class HelloForm extends ActionForm{
    private String msg;

    public String getMsg() {
        return msg;
    }

    public void setMsg(String msg) {
        this.msg = msg;
    }
}

```

### 3.3.2 修改 HelloAction 控制器

HelloAction.java

```

----- 内容 -----
package com.sneb.struts.action;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;
import com.sneb.struts.form.HelloForm;

public class HelloAction extends Action {
    public ActionForward execute(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response) {

        HelloForm hform = (HelloForm)form;

        hform.setMsg("Hello World!!!");

        // request.setAttribute("msg", "Hello!");

        return mapping.findForward("show");
    }
}
----- 内容结束 -----

```

截图如下：

```
1 package com.sneb.struts.action;
2
3 import javax.servlet.http.HttpServletRequest;
4 import javax.servlet.http.HttpServletResponse;
5
6 import org.apache.struts.action.Action;
7 import org.apache.struts.action.ActionForm;
8 import org.apache.struts.action.ActionForward;
9 import org.apache.struts.action.ActionMapping;
10
11 import com.sneb.struts.form.HelloForm;
12
13 public class HelloAction extends Action {
14
15
16
17     public ActionForward execute(ActionMapping mapping, ActionForm form,
18         HttpServletRequest request, HttpServletResponse response) {
19
20         HelloForm hform = (HelloForm)form;
21         hform.setMsg("Hello World!!!");
22         // request.setAttribute("msg", "Hello!");
23
24         return mapping.findForward("show");
25     }
26 }
27
28
29 }
```

### 3.3.3 Struts-config.xml 配置修改

----- 内容 -----

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE struts-config PUBLIC "-//Apache Software Foundation//DTD Struts
Configuration 1.3//EN" "http://struts.apache.org/dtds/struts-
config_1_3.dtd">

<struts-config>

    <form-beans>
        <form-bean name="helloForm" type="com.sneb.struts.form.HelloForm"
        />
    </form-beans>

    <global-exceptions />
    <global-forwards />

    <action-mappings>
```

```

        <action name="helloForm" path="/hello" scope="request"
            type="com.sneb.struts.action.HelloAction">
            <forward name="show" path="/index.jsp"
redirect="false"></forward>
        </action>
    </action-mappings>

    <message-resources
parameter="com.yourcompany.struts.ApplicationResources" />
</struts-config>

```

----- 内容结束 -----

截图如下：

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE struts-config PUBLIC "-//Apache Software Foundation//DTD Struts Configuration 1.3//EN"
3
4 <struts-config>
5
6
7 <form-beans>
8   <form-bean name="helloForm" type="com.sneb.struts.form.HelloForm" />
9 </form-beans>
10
11   <global-exceptions />
12   <global-forwards />
13
14 <action-mappings>
15
16
17   <action name="helloForm" path="/hello" scope="request"
18     type="com.sneb.struts.action.HelloAction">
19     <forward name="show" path="/index.jsp" redirect="false"></forward>
20   </action>
21
22 </action-mappings>
23
24
25   <message-resources parameter="com.yourcompany.struts.ApplicationResources" />
26 </struts-config>
27

```

### 3.3.4 视图 index.jsp 修改

----- 内容 -----

```

<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>
<%
    String path = request.getContextPath();
    String basePath = request.getScheme() + "://"
        + request.getServerName() + ":" + request.getServerPort()
        + path + "/";
%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>

```

```
<base href="<%=basePath%>">

<title>My JSP 'index.jsp' starting page</title>
<meta http-equiv="pragma" content="no-cache">
<meta http-equiv="cache-control" content="no-cache">
<meta http-equiv="expires" content="0">
<meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
<meta http-equiv="description" content="This is my page">
<!--
    <link rel="stylesheet" type="text/css" href="styles.css">
-->
</head>

<body>

    ${helloForm.msg}

</body>
</html>
```

----- 内容结束 -----

截图：

```

1  <%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>
2  <%
3      String path = request.getContextPath();
4      String basePath = request.getScheme() + "://"
5          + request.getServerName() + ":" + request.getServerPort()
6          + path + "/";
7  %>
8
9  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
10 <html>
11 <head>
12 <base href="<%=basePath%>">
13
14 <title>My JSP 'index.jsp' starting page</title>
15 <meta http-equiv="pragma" content="no-cache">
16 <meta http-equiv="cache-control" content="no-cache">
17 <meta http-equiv="expires" content="0">
18 <meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
19 <meta http-equiv="description" content="This is my page">
20 <!--
21     <link rel="stylesheet" type="text/css" href="styles.css">
22     -->
23 </head>
24
25 <body>
26
27     ${helloForm.msg}
28
29 </body>
30 </html>
31

```

### 3.3.4 运行检验

略，同 3.2.7

## 4. Hibernate 简介

Hibernate 框架 ORM 的实现原理-不是技术的技术

<http://blog.csdn.net/fenglibing/article/details/1767339>

### 4.1 什么是 ORM

ORM 的全称是 Object Relational Mapping，即对象关系映射。它的实现思想就是将关系数据库中表的数据映射成为对象，以对象的形式展现，这样开发人员就可以把对数据库的操作转化为对这些对象的操作。因此它的目的是为了更方便开发人员以面向对象的思想来实现对数据库的操作。

## 4.2 什么是 Hibernate

对于 Hibernate 的称呼有很多，比如工具、技术、框架以及解决方案等，这些都可以，重要的是大家要知道它的作用。在这里我习惯性称它为框架，它是一种能实现 ORM 的框架。能实现 ORM 这个功能的框架有很多，Hibernate 可以说是这些框架中最流行、最受开发者关注的，甚至连 JBoss 公司也把它吸收进来，利用它在自己的项目中实现 ORM 功能。

## 4.3 ORM 的实现原理

现在在 Java 领域大家对 Hibernate 的讨论很多，比如它的优缺点、如何应用、错误如何解决以及把它和 Struts/Spring 等框架相结合作为整个系统的解决方案。在这里我想和大家探讨一些更深层次的话题，那就是 Hibernate 是如何实现 ORM 的功能？如果让我们自己开发一款实现 ORM 功能的框架需要怎么做？其实这些问题就是围绕着一个词，那就是“映射”，如果我们知道如何实现这种映射那么我们也能够开发出自己的一款 ORM 框架。会使用 Hibernate 的开发人员都知道，在使用它实现 ORM 功能的时候，主要的文件有：映射类(\*.java)、映射文件 (\*.hbm.xml) 以及数据库配置文件 (\*.properties 或 \*.cfg.xml)，它们各自的作用如下。

### 4.3.1 映射类：

它的作用是描述数据库表的结构，表中的字段在类中被描述成属性，将来就可以实现把表中的记录映射成为该类的对象。

### 4.3.2 映射文件：

它的作用是指定数据库表和映射类之间的关系，包括映射类和数据库表的对应关系、表字段和类属性类型的对应关系以及表字段和类属性名称的对应关系等。

### 4.3.3 数据库配置文件：

它的作用是指定与数据库连接时需要的连接信息，比如连接哪中数据库、登录用户名、登录密码以及连接字符串等。

在这三种主要的文件中，映射类为普通 Java 源文件、映射文件为 XML 格式、数据库配置文件为 Properties 格式或者是 XML 格式。想理解“映射”首先我们需要知道如何解析这三种文件，即解析 XML 格式文件、解析 Properties 格式文件和解析 Java 类文件。

## 4.4 hibernate 的工作流程范例

## 5. Spring

### 5.1 工厂模式

### 5.2 Spring 的 IOC 原理

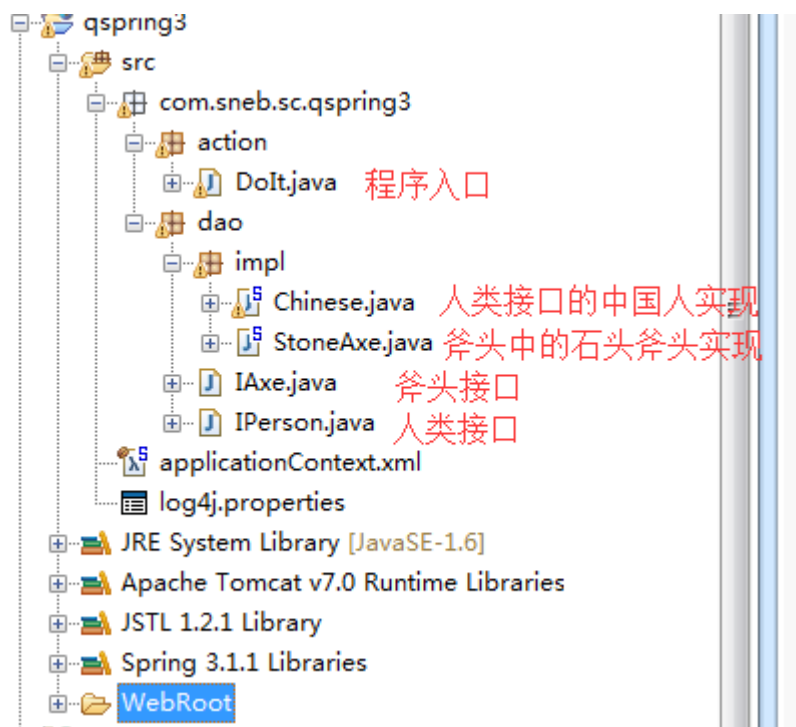
我们可以把 IOC 容器的工作模式看做是工厂模式的升华，可以把 IOC 容器看作是一个工厂，这个工厂里要生产的对象都在配置文件中给出定义，然后利用编程语言的反射编程，根据配置文件中给出的类名生成相应的对象。从实现来看，IOC 是把以前在工厂方法里写死的对象生成代码，改变为由配置文件来定义，也就是把工厂和对象生成这两者独立分隔开来，目的就是提高灵活性和可维护性。

<http://blog.csdn.net/m13666368773/article/details/7802126>

### 5.3 Spring 的设值注入范例

新建 Web Project 工程，取名 qspring3，目录结构和类文件如图所示：





### 5.3.1 接口定义

com.sneb.sc.qspring3.dao

IAxe.java

----- 内容 -----

```
package com.sneb.sc.qspring3.dao;
```

```
public interface IAxe {  
    public void chop();  
}
```

com.sneb.sc.qspring3.dao

IPerson.java

----- 内容 -----

```
package com.sneb.sc.qspring3.dao;
```

```
public interface IPerson {  
  
    public void userAxe();  
}
```

### 5.3.2 接口实现

com.sneb.sc.qspring3.dao.impl

Chinese.java

----- 内容 -----

```
package com.sneb.sc.qspring3.dao.impl;

import com.sneb.sc.qspring3.dao.IAxe;
import com.sneb.sc.qspring3.dao.IPerson;

public class Chinese implements IPerson{

    private IAxe axe;

    public IAxe getAxe() {
        return axe;
    }

    public void setAxe(IAxe axe) {
        this.axe = axe;
    }

    @Override
    public void userAxe() {
        // TODO Auto-generated method stub
        axe.chop();
    }

}
```

com.sneb.sc.qspring3.dao.impl

StoneAxe.java

----- 内容 -----

```
package com.sneb.sc.qspring3.dao.impl;
import com.sneb.sc.qspring3.dao.IAxe;

public class StoneAxe implements IAxe{

    @Override
    public void chop() {
        // TODO Auto-generated method stub
        System.out.println("石斧慢慢砍。。。");
    }

}
```

}

### 5.3.3 注入配置

src 目录下

applicationContext.xml

```
----- 内容 -----
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:p="http://www.springframework.org/schema/p"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-3.1.xsd">

    <bean id="stoneAxe" class="com.sneb.sc.qspring3.dao.impl.StoneAxe"
          abstract="false" lazy-init="default" autowire="default">
    </bean>

    <bean id="chinese" class="com.sneb.sc.qspring3.dao.impl.Chinese"
          abstract="false" lazy-init="default" autowire="default" p:axe-
ref="stoneAxe">
    </bean>

</beans>
```

截图:



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:p="http://www.springframework.org/schema/p"
4       xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-3.1.xsd">
5
6     <bean id="stoneAxe" class="com.sneb.sc.qspring3.dao.impl.StoneAxe"
7           abstract="false" lazy-init="default" autowire="default">
8     </bean>
9
10    <bean id="chinese" class="com.sneb.sc.qspring3.dao.impl.Chinese"
11          abstract="false" lazy-init="default" autowire="default" p:axe-ref="stoneAxe">
12    </bean>
13
14 </beans>
```

### 5.3.4 调用执行

com.sneb.sc.qspring3.action

DoIt.java

```
----- 内容 -----
package com.sneb.sc.qspring3.action;
```

```

import org.springframework.beans.factory.BeanFactory;
import org.springframework.beans.factory.xml.XmlBeanFactory;
import org.springframework.core.io.FileSystemResource;
import org.springframework.core.io.Resource;
import com.sneb.sc.qspring3.dao.IPerson;

public class DoIt {

    public static void main(String[] args) {

        Resource r = new FileSystemResource("src/applicationContext.xml");
        BeanFactory f = new XmlBeanFactory(r);

        IPerson person = null;

        person = (IPerson) f.getBean("chinese");
        person.userAxe();

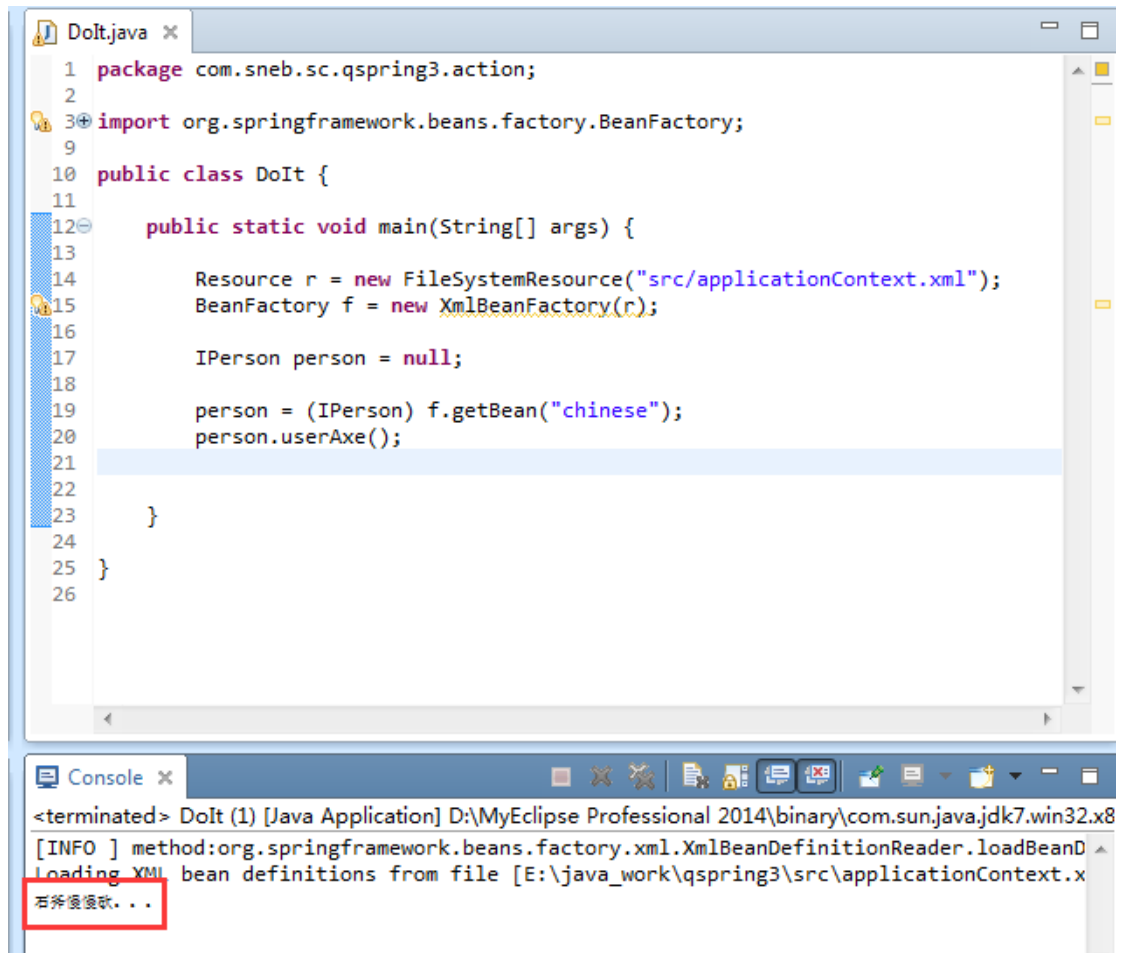
    }

}

```

### 5.3.5 运行校验

菜单 → Run → Run As → Java Application  
 效果截图：



## 5.4 Spring 的构造注入

略

## 6. 综合例子（一）

### 6.1 ide 中添加 mysql 数据库驱动创建连接

#### 6.1.1 下载 mysql 的 jdbc 驱动

官方地址：

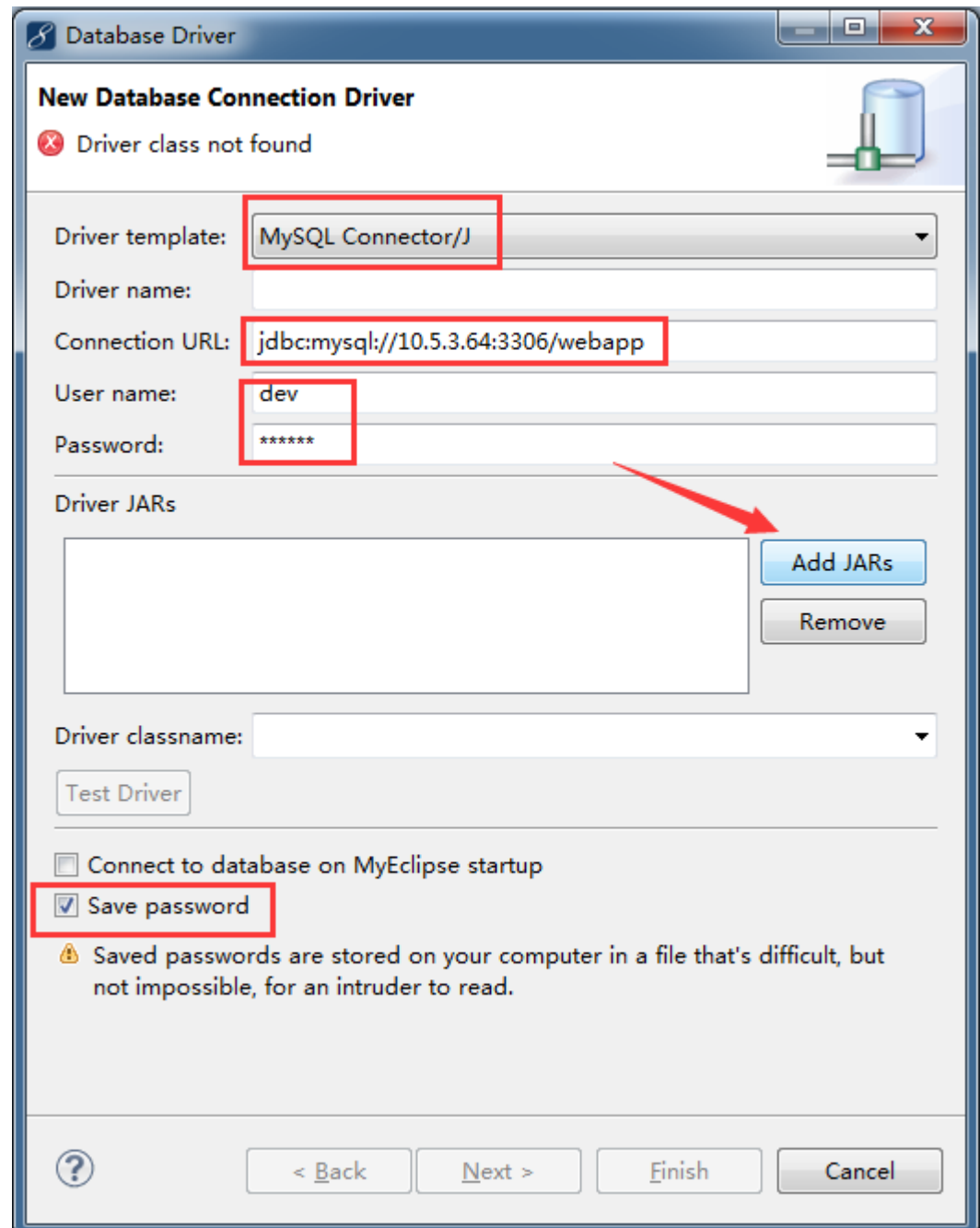
<https://dev.mysql.com/get/Downloads/Connector-J/mysql-connector-java-5.1.40.zip>

还需要下载 `asm-all-4.0.jar` 包，解决冲突异常问题。

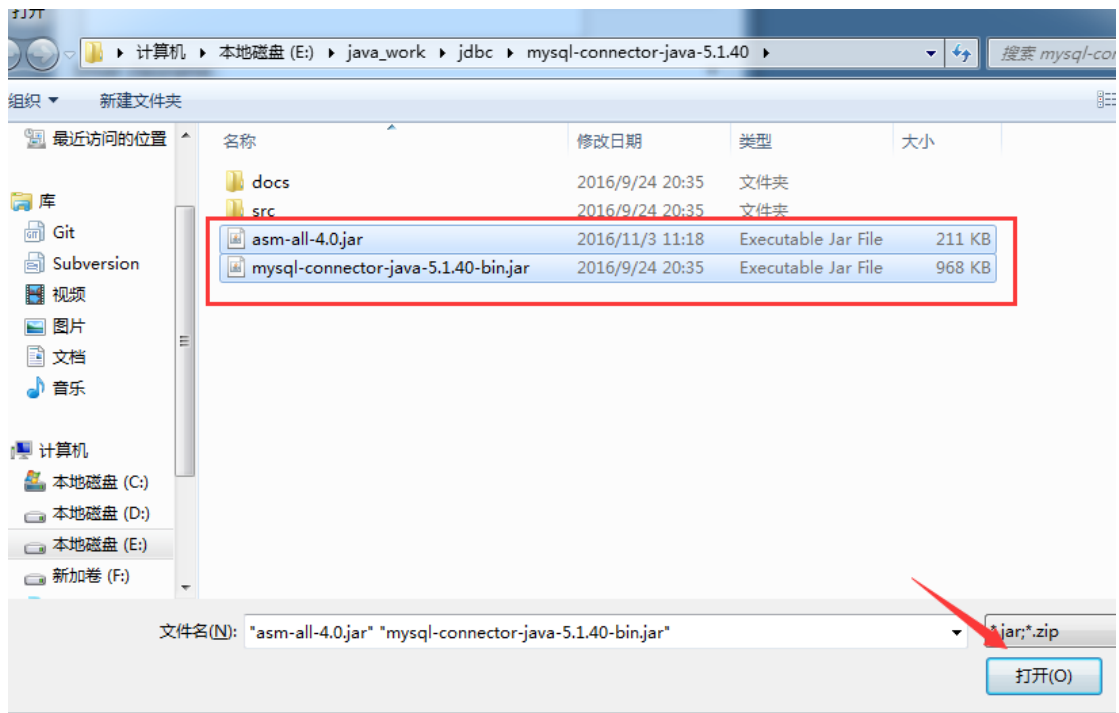
## 6.1.2 建立数据库连接

菜单栏→Windows → Show view →Other→DB Browser→OK

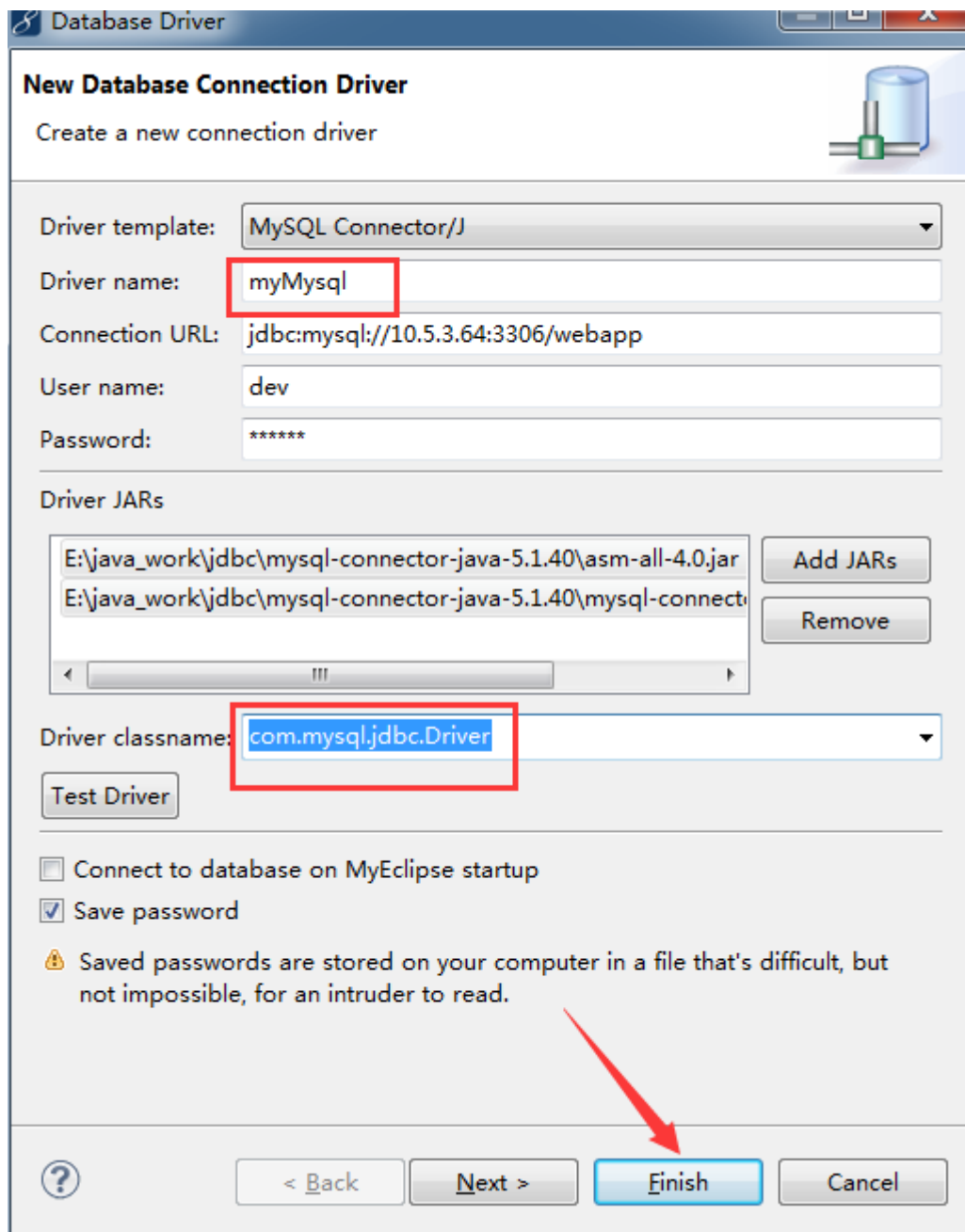
MyEclipse Derby → New →



→ Add JARs →



→



➔ 完成

可以查看数据库的表结构。

可以为 myeclipse 里的多个项目生成 bean

## 6.2 新建工程

### 6.2.1 新建一个 Web Project 工程

File ➔ New ➔ Web Project ➔



**Create a JavaEE Web Project**  
Specify project name and other details

Project name: **ssh2**

Project location  
☒ Use default location  
Location: E:\java\_work\ssh2 Browse...

Project configuration  
Java EE version: JavaEE 6 - Web 3.0  
Java version: 1.6  
JSTL Version: 1.2.1  
☐ Add maven support [Learn more about Maven4MyEclipse...](#)

Target runtime  
**Apache Tomcat v7.0** Add New Runtime...

EAR membership  
☐ Add project to an EAR  
EAR project name: ssh2EAR New Project...

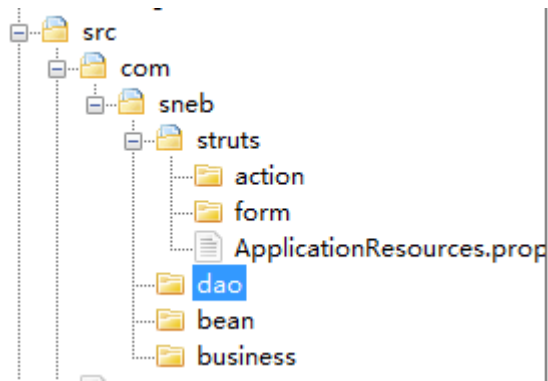
Working sets  
☐ Add project to working sets  
Working sets: Select...

? < Back Next > **Finish** Cancel

→Finish

## 6.2.2 创建好包目录名

先创建好包目录名，备用，目录结构如下：

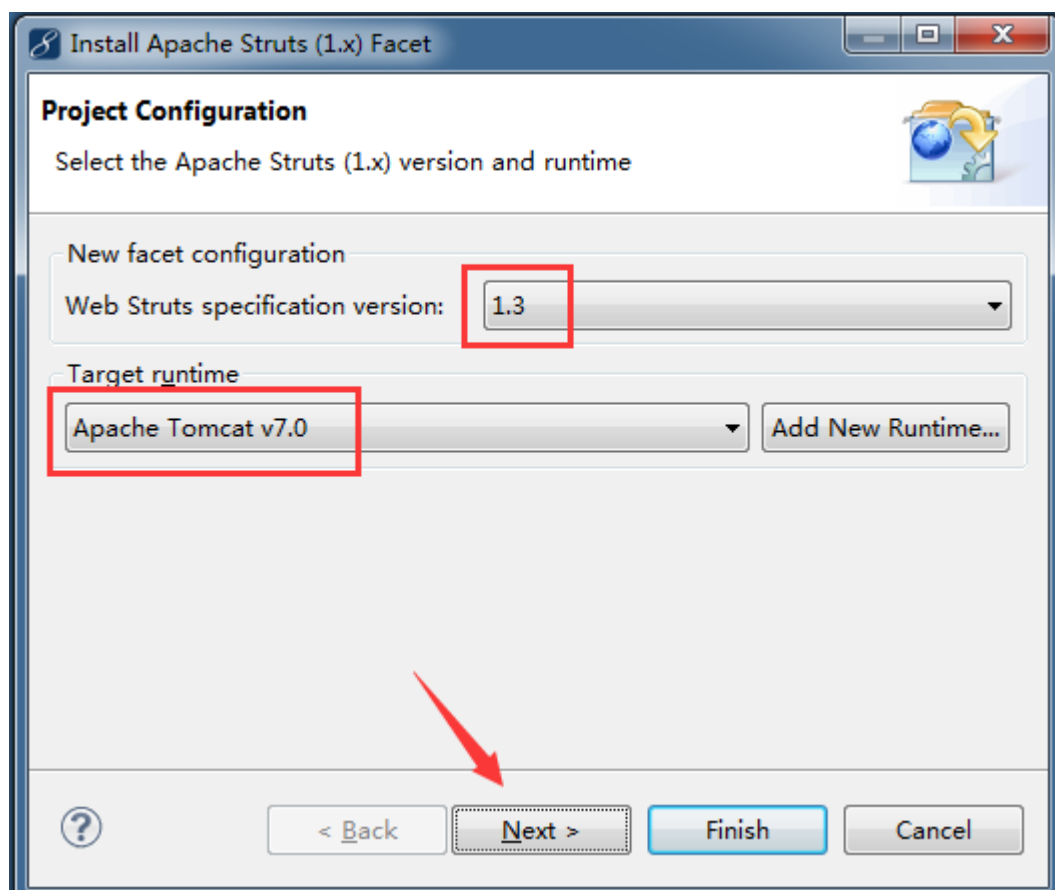


dao 和 business 目录下都创建 impl 子目录

## 6.3 导入 spring、struts 和 hibernate 包

### 6.3.1 导入 struts1.3 类库

选中 ssh2→右键 → MyEclipse → Project Facets [Capabilities] → Install Apache Struts(1.x)Facet



→

**Web Struts 1.x**  
Configure Web Struts 1.x settings


Struts config path:

ActionServlet name:

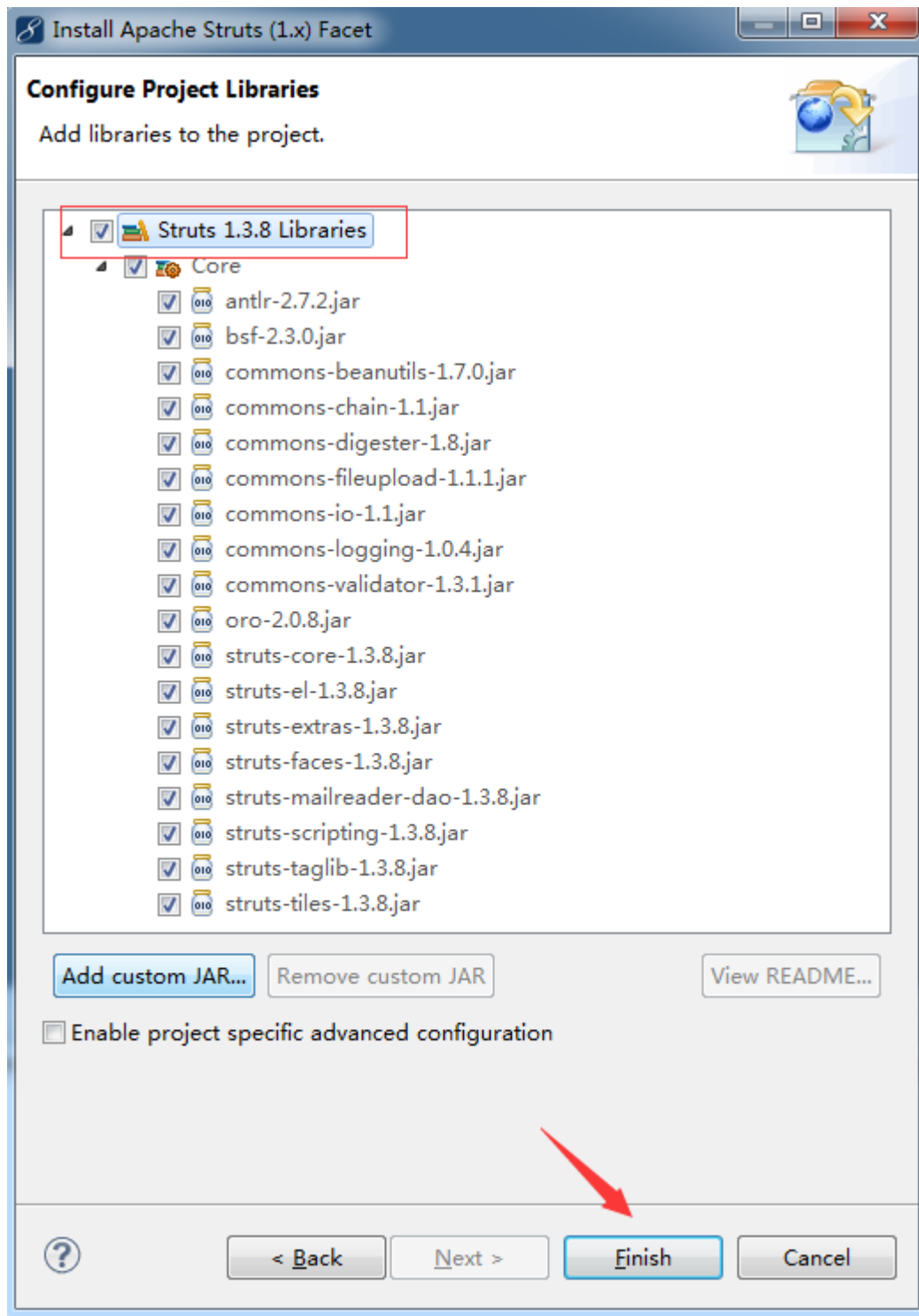
URL pattern: ☒ \*.do ☐ /do/\*

Base package for new classes:

Default application resources:



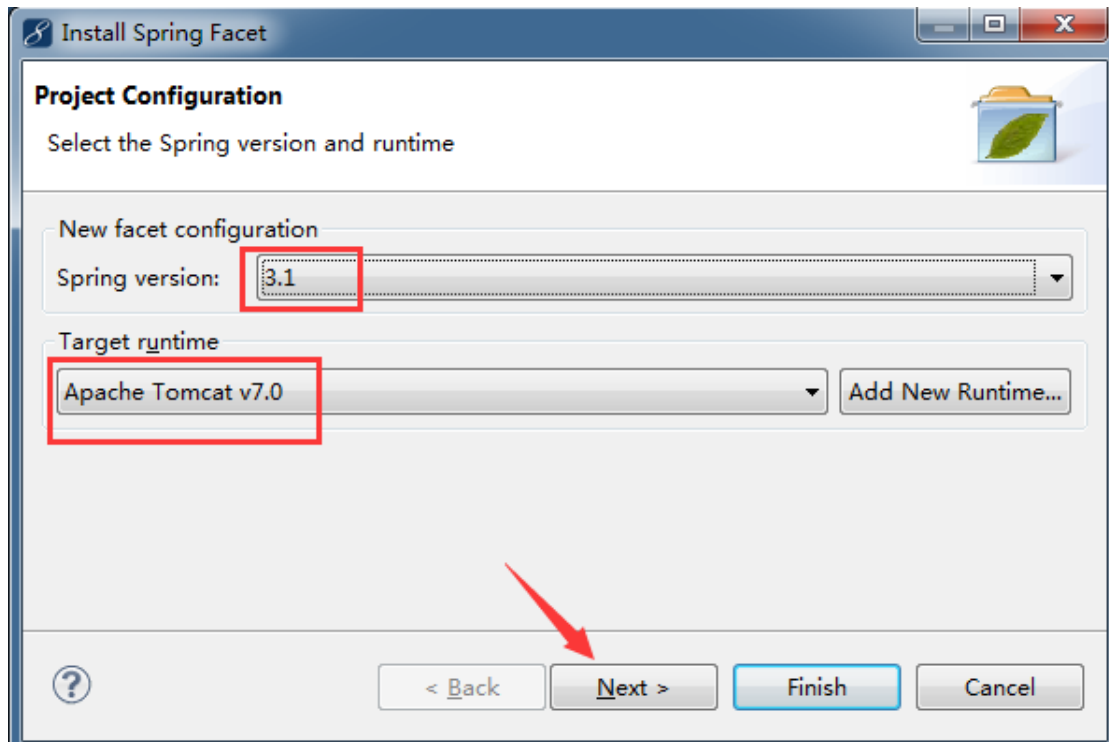
→



如果熟悉后，可以勾选需要导入的包。不熟悉的话，全选→ Finish

### 6.3.2 导入 spring 类包

选中 ssh2→右键 → MyEclipse → Project Facets [Capabilities] →Install Spring Facet

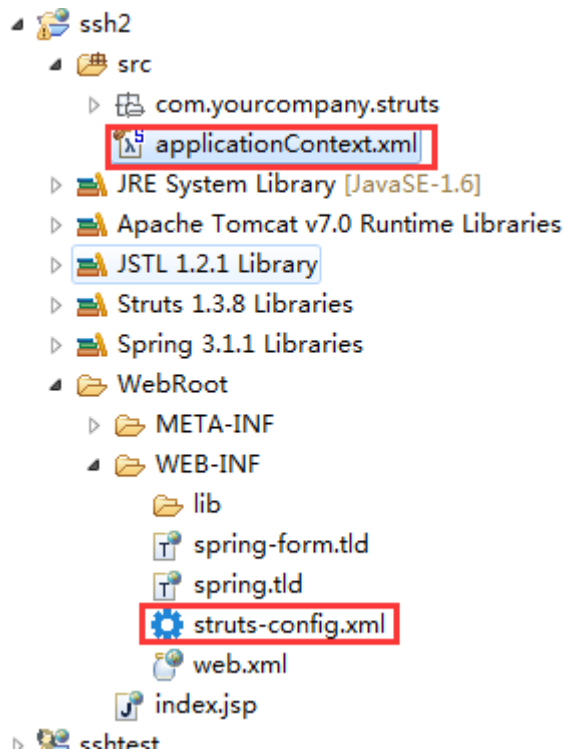


→ Finish

以上操作只是帮你导入类包到工程，并生成默认的配置文件

- ssh2
  - src
  - JRE System Library [JavaSE-1.6]
  - Apache Tomcat v7.0 Runtime Libraries
  - JSTL 1.2.1 Library
  - Struts 1.3.8 Libraries
    - antlr-2.7.2.jar - D:\MyEclipse Professional 2014\
    - bsf-2.3.0.jar - D:\MyEclipse Professional 2014\c
    - commons-beanutils-1.7.0.jar - D:\MyEclipse Prc
    - commons-chain-1.1.jar - D:\MyEclipse Professi
    - commons-digester-1.8.jar - D:\MyEclipse Profe
    - commons-fileupload-1.1.1.jar - D:\MyEclipse Pr
    - commons-io-1.1.jar - D:\MyEclipse Professiona
    - commons-logging-1.0.4.jar - D:\MyEclipse Prof
    - commons-validator-1.3.1.jar - D:\MyEclipse Prc
    - oro-2.0.8.jar - D:\MyEclipse Professional 2014\
    - struts-core-1.3.8.jar - D:\MyEclipse Professiona
    - struts-el-1.3.8.jar - D:\MyEclipse Professional 20
    - struts-extras-1.3.8.jar - D:\MyEclipse Professior
    - struts-faces-1.3.8.jar - D:\MyEclipse Professiona
    - struts-mailreader-dao-1.3.8.jar - D:\MyEclipse I
    - struts-scripting-1.3.8.jar - D:\MyEclipse Profess
    - struts-taglib-1.3.8.jar - D:\MyEclipse Profession
    - struts-tiles-1.3.8.jar - D:\MyEclipse Professional
  - Spring 3.1.1 Libraries
    - org.springframework.aop-3.1.1.RELEASE.jar - C
    - org.springframework.asm-3.1.1.RELEASE.jar - C
    - org.springframework.aspects-3.1.1.RELEASE.jar

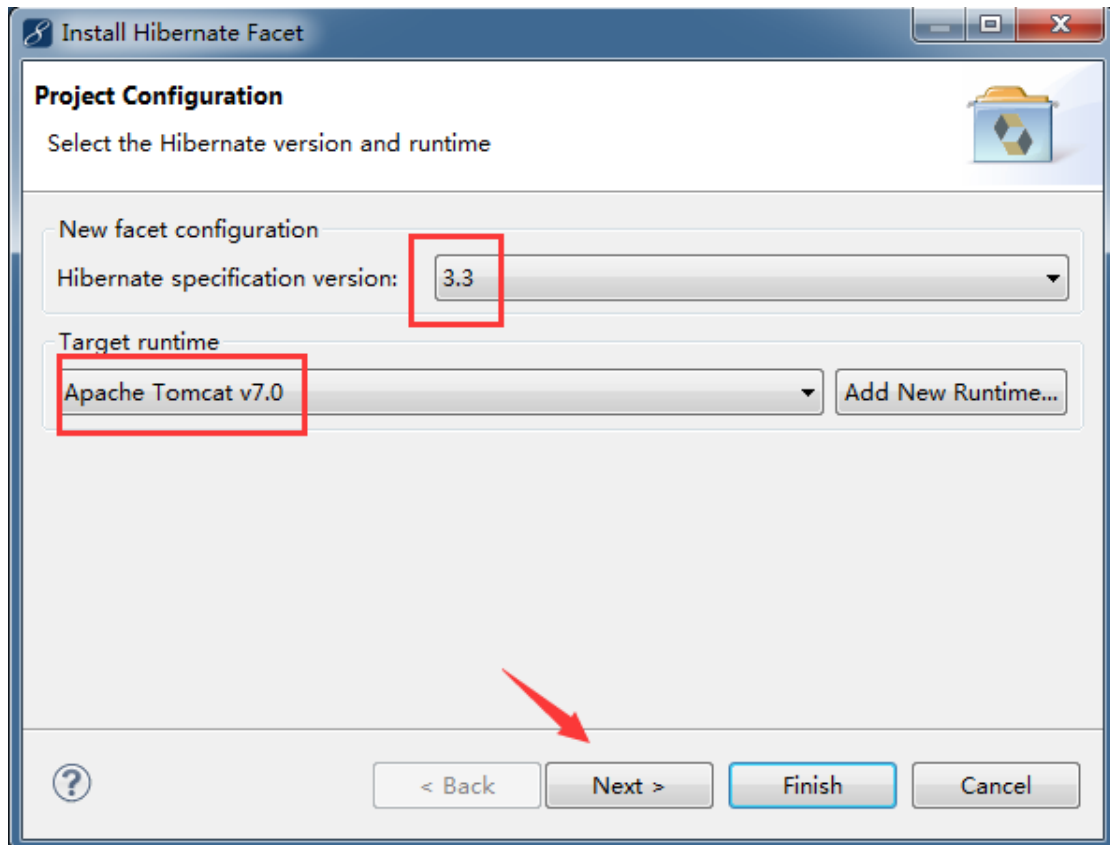
核心配置文件如下：



applicationContext.xml 是 Spring 的核心配置文件；  
struts-config.xml 是 Struts 的核心配置文件。

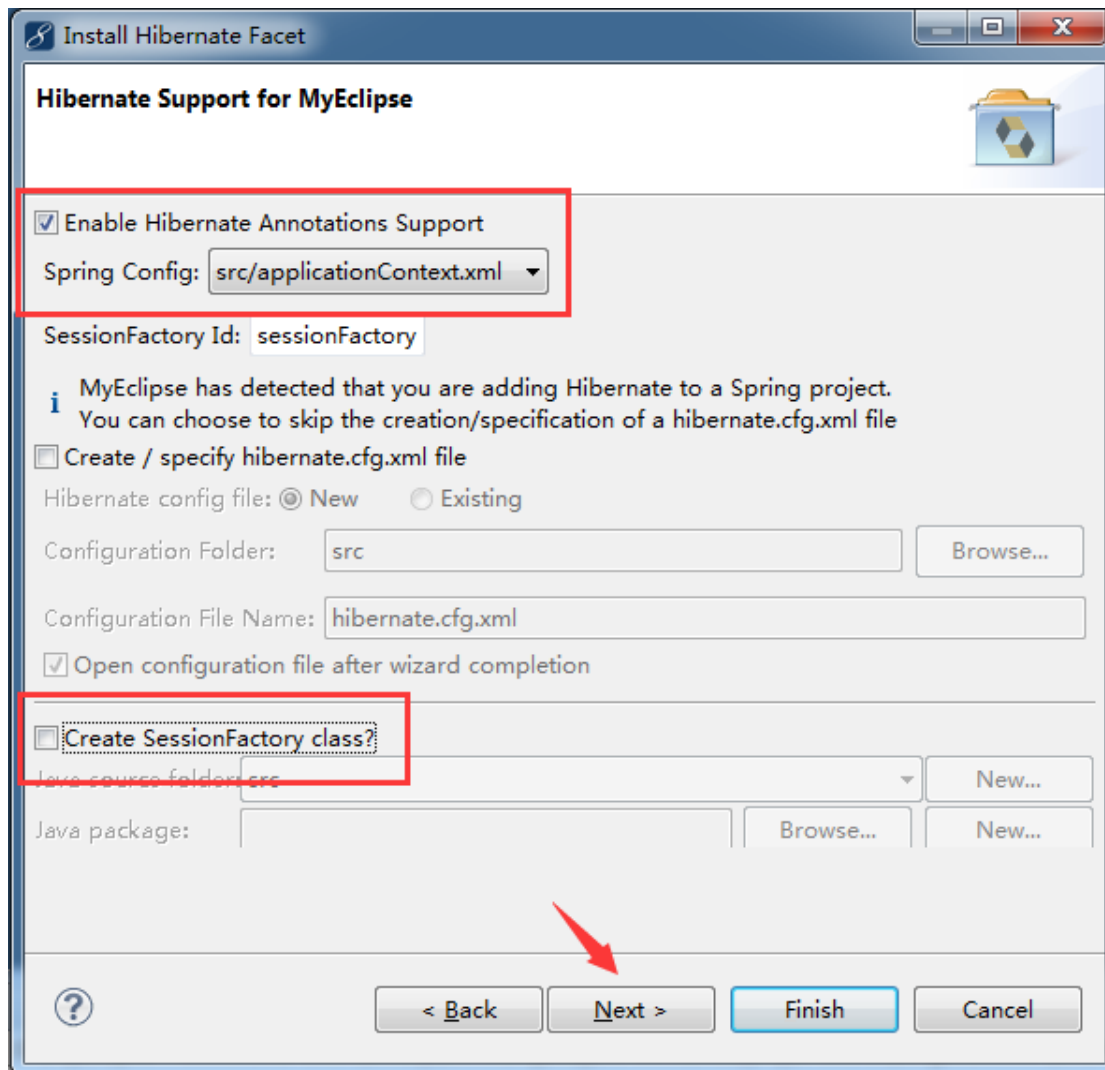
### 6.3.3 导入 hibernate 包

选中 ssh2 右键 → MyEclipse → Project Facets [Capabilities] → Install Hibernate Facet

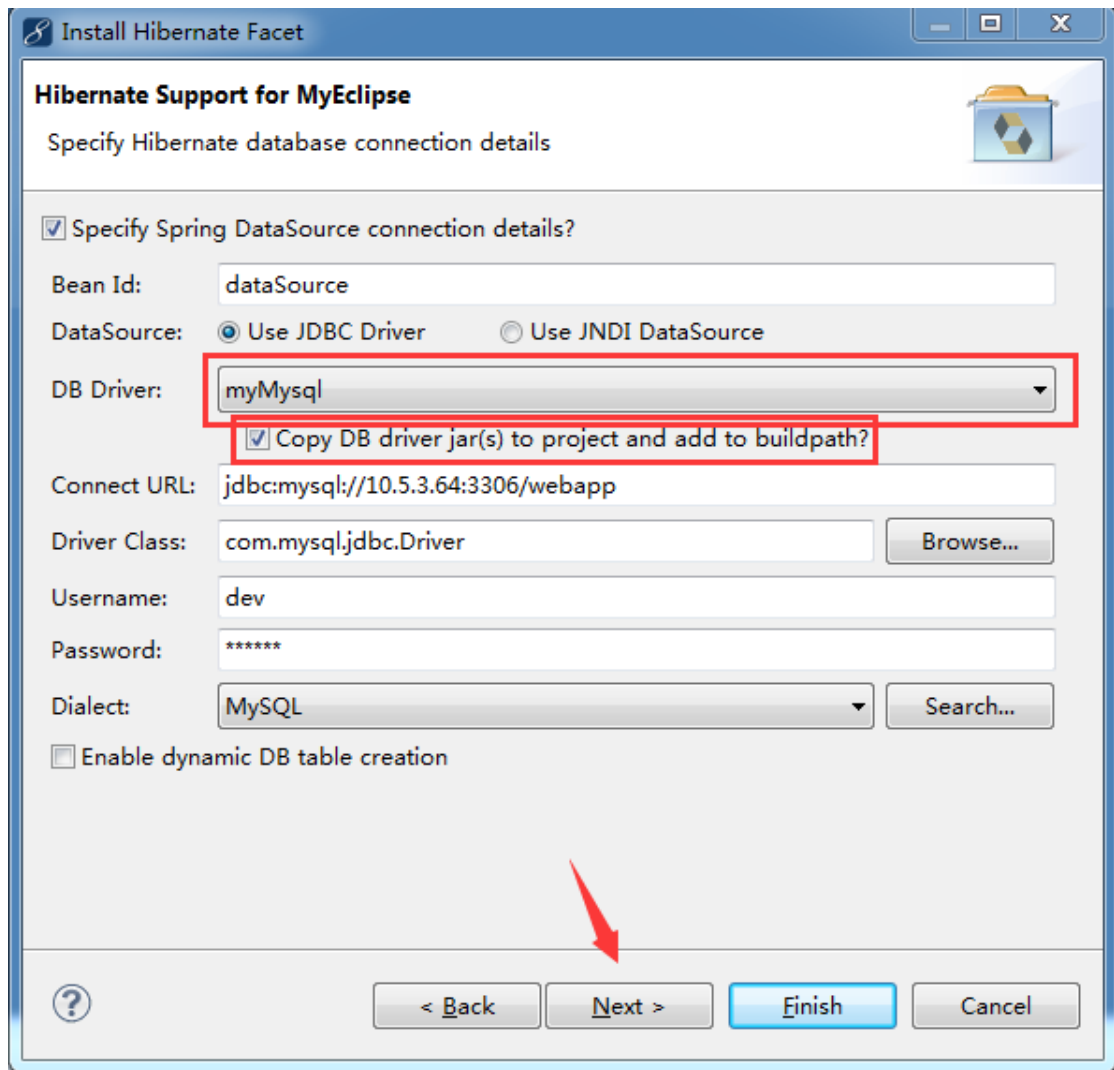


→





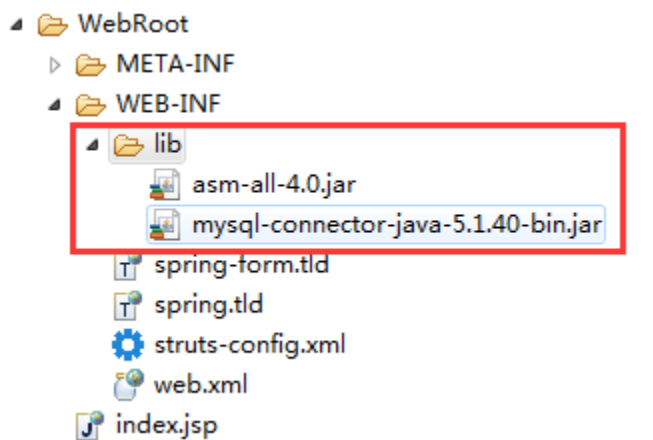
→ 选 db Driver



→

→ 弹出界面切换对话框，选 no → Finish。

查看是否已经导入了 2 个 jar 包



同时，IDE 会修改 Spring 的配置文件

```

applicationContext.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans
3     xmlns="http://www.springframework.org/schema/beans"
4     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5     xmlns:p="http://www.springframework.org/schema/p"
6     xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org
7
8
9 <bean id="dataSource"
10     class="org.apache.commons.dbcp.BasicDataSource">
11     <property name="url"
12         value="jdbc:mysql://10.5.3.64:3306/webapp">
13     </property>
14     <property name="username" value="dev"></property>
15     <property name="password" value="dev963"></property>
16 </bean>
17 <bean id="sessionFactory"
18     class="org.springframework.orm.hibernate3.annotation.AnnotationSessionFactoryBean">
19     <property name="dataSource">
20         <ref bean="dataSource" />
21     </property>
22     <property name="hibernateProperties">
23         <props>
24             <prop key="hibernate.dialect">
25                 org.hibernate.dialect.MySQLDialect
26             </prop>
27         </props>
28     </property>
29 </bean></beans>

```

这里要修改，添加 bean 的属性，不然会报异常。

----- 内容 -----

<property name="driverClassName" value="com.mysql.jdbc.Driver" />

----- 内容结束 -----

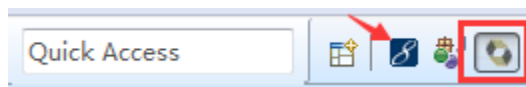
截图：

```

*applicationContext.xml
applicationContext.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans
3     xmlns="http://www.springframework.org/schema/beans"
4     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5     xmlns:p="http://www.springframework.org/schema/p"
6     xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.or
7
8
9 <bean id="dataSource"
10     class="org.apache.commons.dbcp.BasicDataSource">
11     <property name="driverClassName" value="com.mysql.jdbc.Driver" />
12     <property name="url"
13         value="jdbc:mysql://10.5.3.64:3306/webapp">
14     </property>
15     <property name="username" value="dev"></property>
16     <property name="password" value="dev963"></property>
17 </bean>
18 <bean id="sessionFactory"
19     class="org.springframework.orm.hibernate3.annotation.AnnotationSessionFactoryBean">
20     <property name="dataSource">
21         <ref bean="dataSource" />
22     </property>
23     <property name="hibernateProperties">
24         <props>
25             <prop key="hibernate.dialect">
26                 org.hibernate.dialect.MySQLDialect
27             </prop>
28         </props>
29     </property>
30 </bean></beans>

```

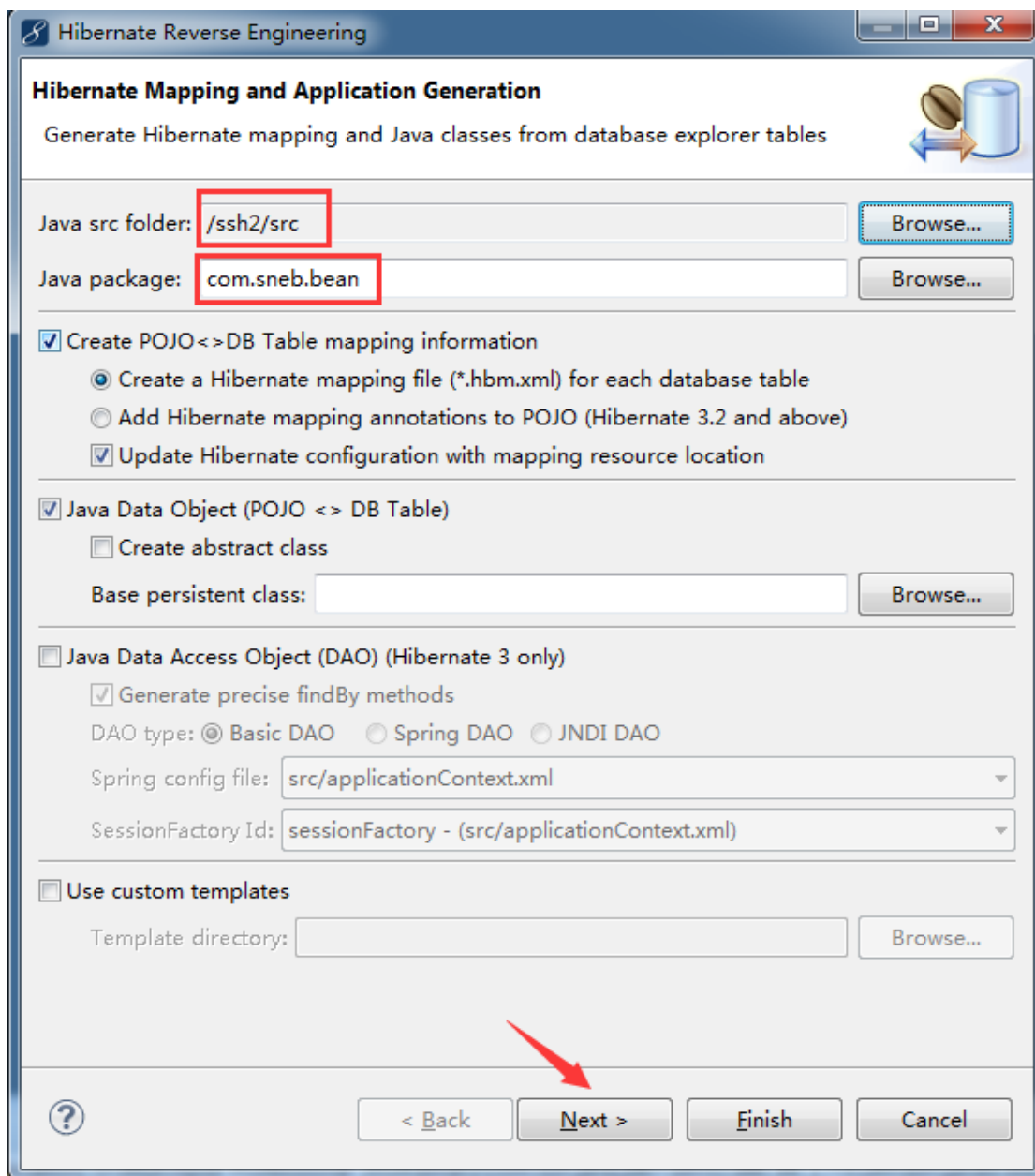
### 6.3.4 如果切换了 IDE 视图，如何回到原视图



点 Myeclipse Java Enterprise 即可

## 6.4 通过数据库生成 bean 类

Db Browser → myMysql → Connected to myMysql → webapp → TABLE → account → 右键 → Hibernate Reverse Engineering →



→

**Hibernate Reverse Engineering**

### Hibernate Mapping and Application Generation

Configure type mapping details

Rev-eng settings file:  **Setup...**

Custom rev-eng strategy:  **Browse...**

Type Mapping: ☒ Java types ☐ Hibernate types

Id Generator: **assigned**

☒ Generate basic typed composite Ids

☐ Generate version and timestamp tags

☐ Enable many-to-many detection

☐ Enable one-to-one detection

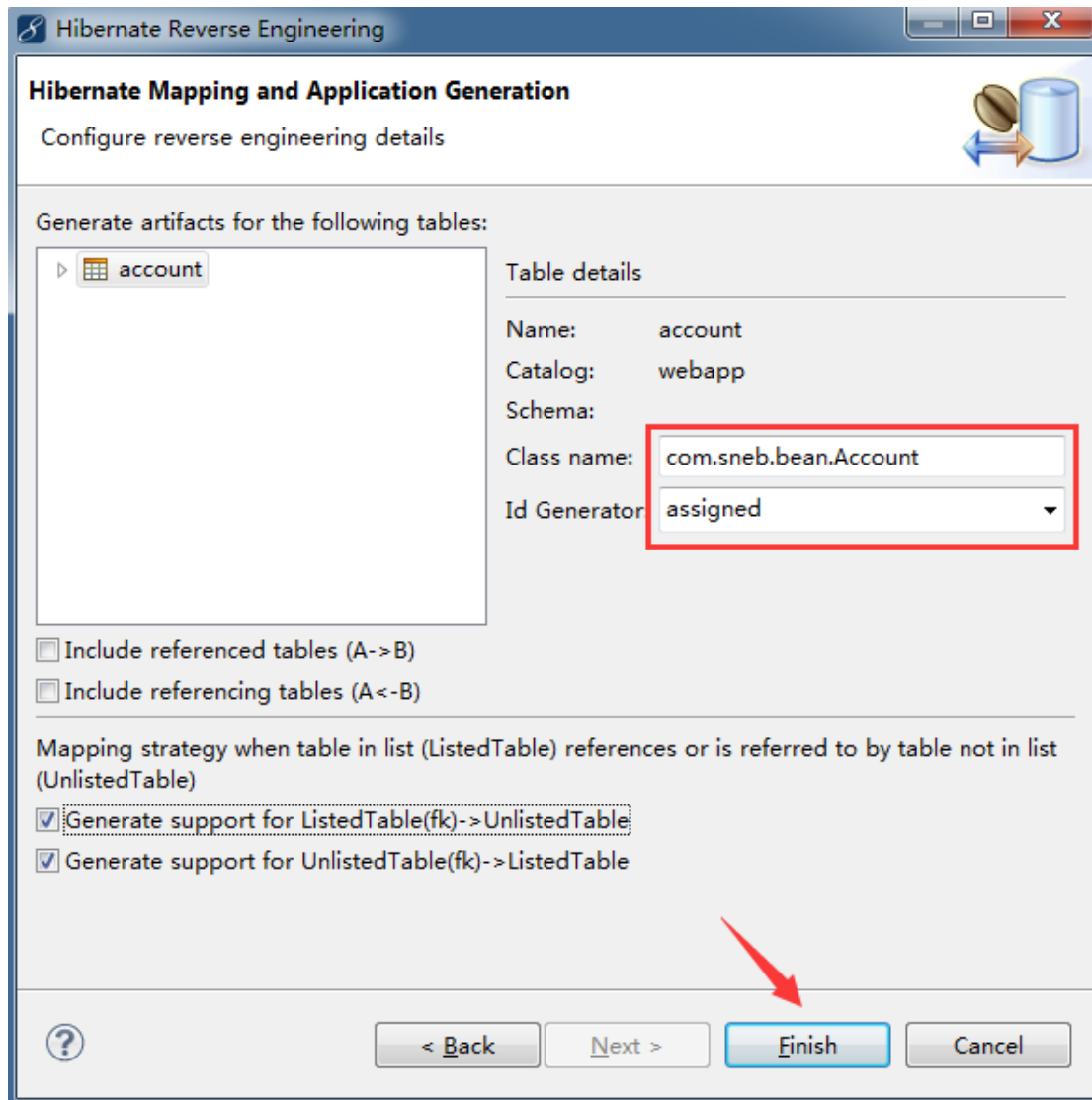
Customized Type Mappings:

JDBC Type	Hibernat...	Length	Scale	Precision	Not-Null

**Add**  
**Remove**  
**Up**  
**Down**

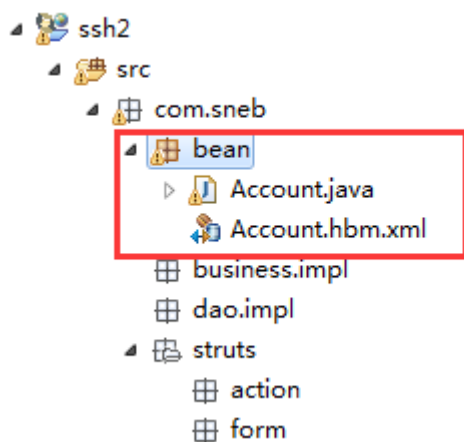
**< Back** **Next >** **Finish** **Cancel**

→



→Finish

查看做了什么：



```
applicationContext.xml x
1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans
3     xmlns="http://www.springframework.org/schema/beans"
4     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5     xmlns:p="http://www.springframework.org/schema/p"
6     xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans
7
8
9     <bean id="dataSource"
10         class="org.apache.commons.dbcp.BasicDataSource">
11         <property name="driverClassName" value="com.mysql.jdbc.Driver" />
12         <property name="url"
13             value="jdbc:mysql://10.5.3.64:3306/webapp">
14         </property>
15         <property name="username" value="dev"></property>
16         <property name="password" value="dev963"></property>
17     </bean>
18     <bean id="sessionFactory"
19         class="org.springframework.orm.hibernate3.annotation.AnnotationSessionFactoryBean">
20         <property name="dataSource">
21             <ref bean="dataSource" />
22         </property>
23         <property name="hibernateProperties">
24             <props>
25                 <prop key="hibernate.dialect">
26                     org.hibernate.dialect.MySQLDialect
27                 </prop>
28             </props>
29         </property>
30         <property name="mappingResources">
31             <list>
32                 <value>com/sneb/bean/Account.hbm.xml</value></list>
33         </property></bean></beans>
```

## 6.5 配置 struts，实现 Hello World

### 6.5.1 创建控制器

WebRoot/WEB-INF 目录下。

双击打开 struts-config.xml 文件 → 源代码窗口点击右键 → New → Action →

**New Action**

**Struts 1.3 Action Declaration**

Create Struts 1.3 Action

Config/Module:

Use case:

---

Path:

Action Type: ☒ Type ☐ Forward ☐ Include

☒ Cancellable

---

Action Impl: ☒ Create new Action class ☐ Use existing Action class

Superclass:

Type:

---

Form **Parameter** Method **Forwards** Exceptions

Forwards:

→

**New Forward**

Name:

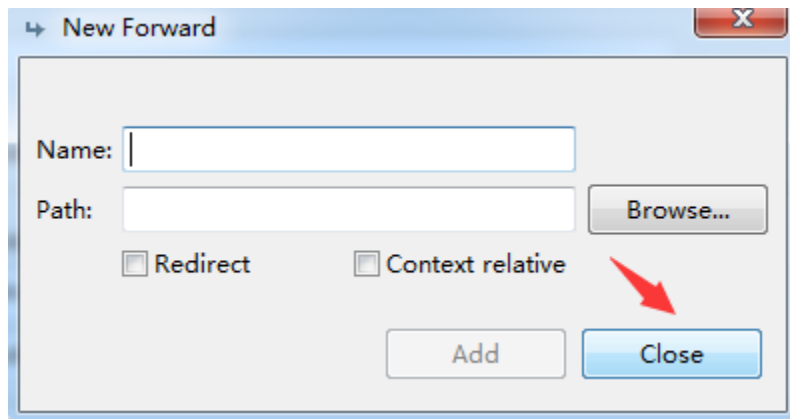
Path:

☐ Redirect ☐ Context relative

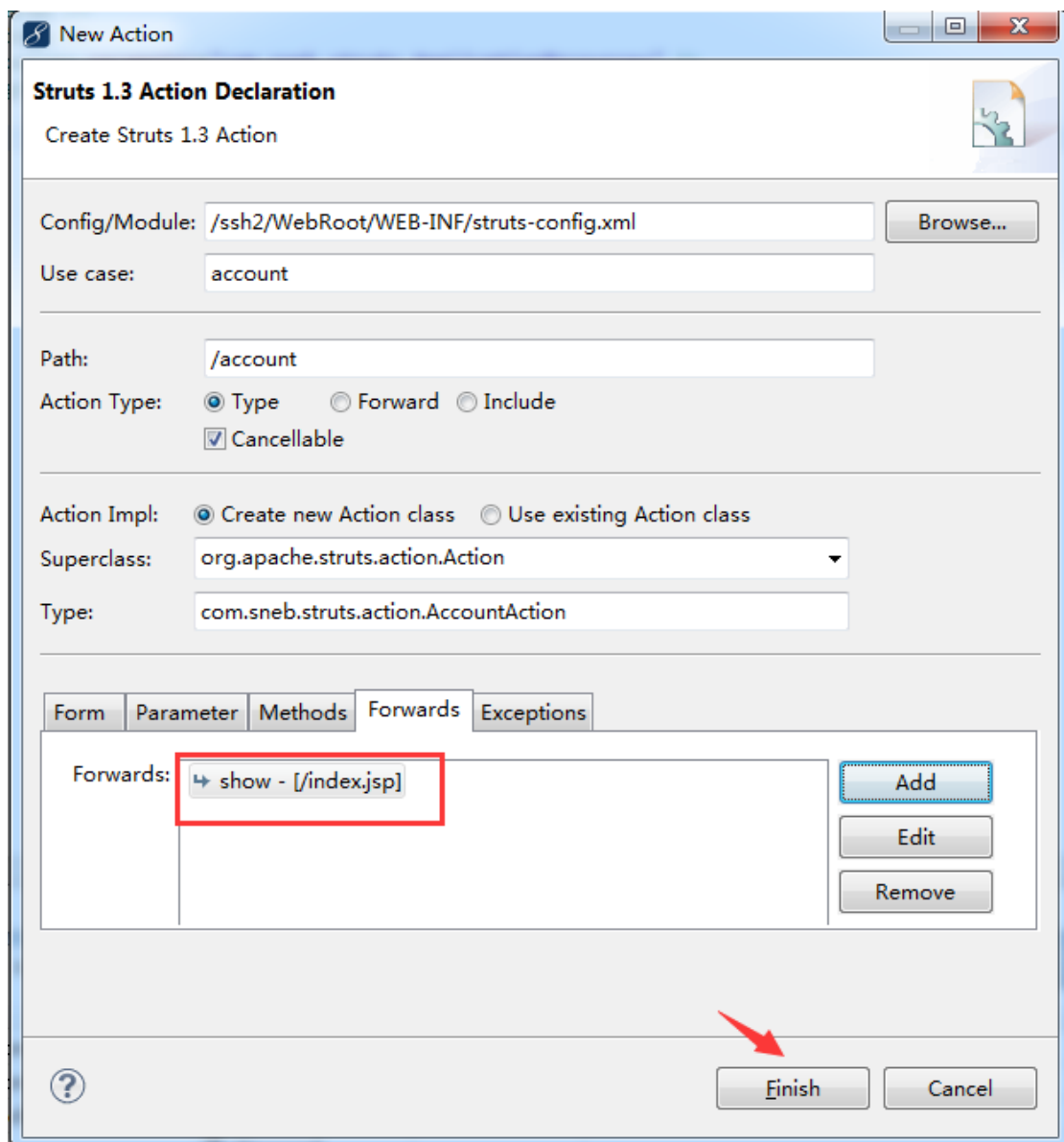
注意视图文件前加右斜杠，也可以点击“Browser”选择视图文件。

→



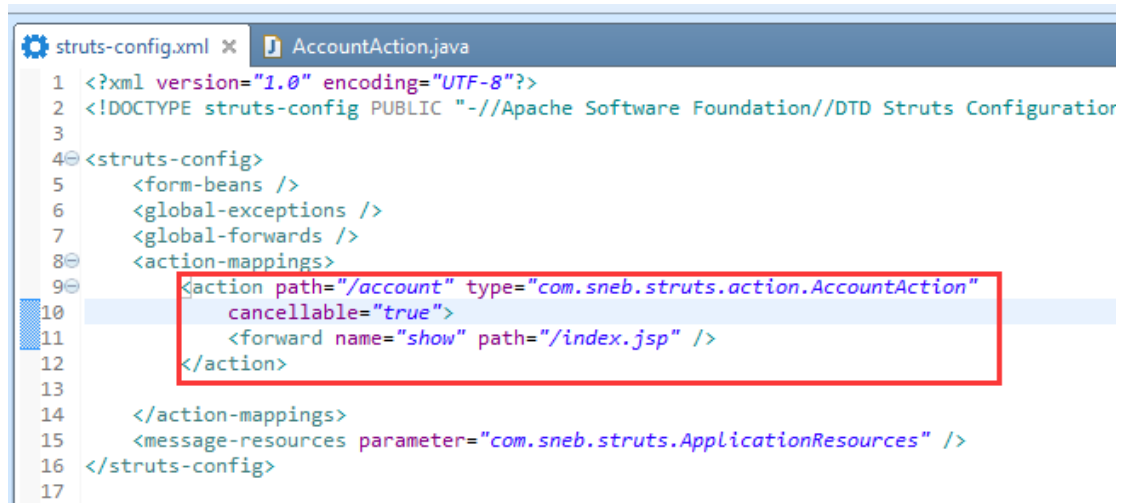


→



→ Finish

该步操作会自动修改 struts-config.xml 配置文件，添加一个 action 配置：



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE struts-config PUBLIC "-//Apache Software Foundation//DTD Struts Configuration
3
4 <struts-config>
5     <form-beans />
6     <global-exceptions />
7     <global-forwards />
8     <action-mappings>
9         <action path="/account" type="com.sneb.struts.action.AccountAction"
10             cancellable="true">
11             <forward name="show" path="/index.jsp" />
12         </action>
13     </action-mappings>
14     <message-resources parameter="com.sneb.struts.ApplicationResources" />
15 </struts-config>
16
17
```

同时会创建一个 AccountAction.java 类。

下面编辑 AccountAction.java，简单输出一个字符串。完整代码如下：

----- 内容 -----

```
/*
 * Generated by MyEclipse Struts
 * Template path: templates/java/JavaClass.vtl
 */
package com.sneb.struts.action;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;

/**
 * MyEclipse Struts
 * Creation date: 11-27-2016
 *
 * XDoclet definition:
 * @struts.action validate="true"
 * @struts.action-forward name="show" path="/index.jsp"
 */
public class AccountAction extends Action {
    /*
     * Generated Methods
     */

    /**
     * Method execute
     * @param mapping

```

```

    * @param form
    * @param request
    * @param response
    * @return ActionForward
    */
    public ActionForward execute(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response) {
        // TODO Auto-generated method stub

        request.setAttribute("data", "Hello Struts!");
        return mapping.findForward("show");
    }
}

```

----- 内容结束 -----

截图:



```

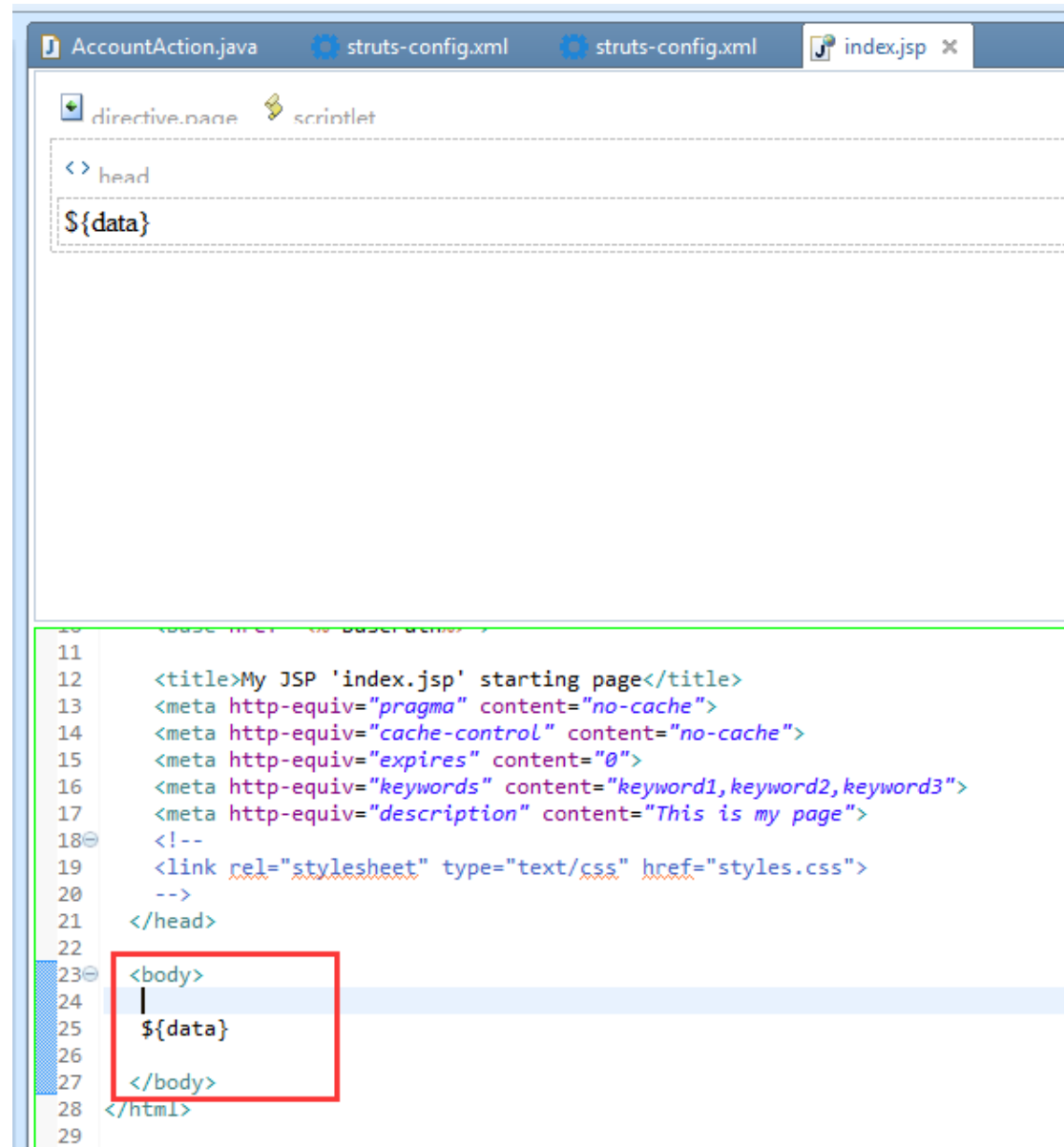
struts-config.xml  AccountAction.java x
2+ * Generated by MyEclipse Struts
5 package com.sneb.struts.action;
6
7+ import javax.servlet.http.HttpServletRequest;
13
14- /**
15  * MyEclipse Struts
16  * Creation date: 11-27-2016
17  *
18  * XDoclet definition:
19  * @struts.action validate="true"
20  * @struts.action-forward name="show" path="/index.jsp"
21  */
22 public class AccountAction extends Action {
23-     /*
24      * Generated Methods
25      */
26
27-     /**
28      * Method execute
29      * @param mapping
30      * @param form
31      * @param request
32      * @param response
33      * @return ActionForward
34      */
35-     public ActionForward execute(ActionMapping mapping, ActionForm form,
36         HttpServletRequest request, HttpServletResponse response) {
37         // TODO Auto-generated method stub
38
39         request.setAttribute("data", "Hello Struts!");
40
41         return mapping.findForward("show");
42     }
43 }

```

## 6.5.2 编写视图文件 index.jsp

文件位置：

WebRoot/index.jsp，代码截图：



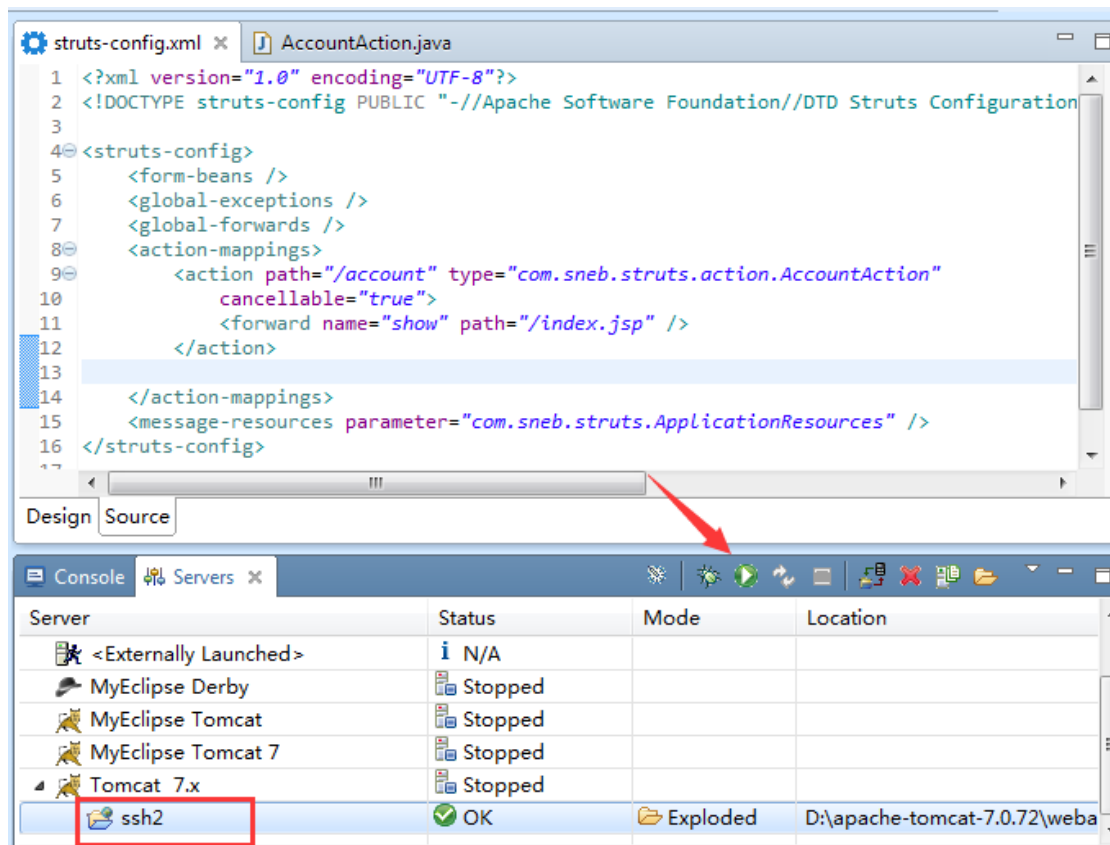
```
11
12 <title>My JSP 'index.jsp' starting page</title>
13 <meta http-equiv="pragma" content="no-cache">
14 <meta http-equiv="cache-control" content="no-cache">
15 <meta http-equiv="expires" content="0">
16 <meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
17 <meta http-equiv="description" content="This is my page">
18 <!--
19 <link rel="stylesheet" type="text/css" href="styles.css">
20 -->
21 </head>
22
23 <body>
24 |
25 |   ${data}
26 |
27 </body>
28 </html>
29
```

## 6.5.3 发布

ssh2 工程上右键 --> MyEclipse --> Add and Remove Project Deployment --> Add --> Server 栏选 “Tomcat 7.x” --> Finish --> ok

## 6.5.4 运行查看效果

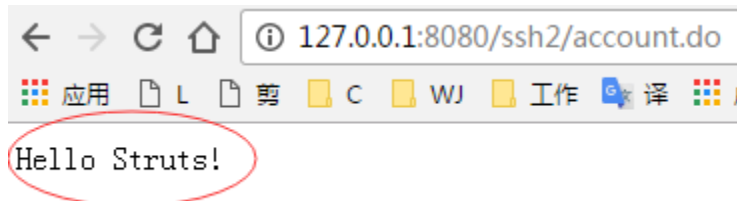
点 Servers 面板的启动按钮



在浏览器输入地址:

<http://127.0.0.1:8080/ssh2/account.do>

效果截图:

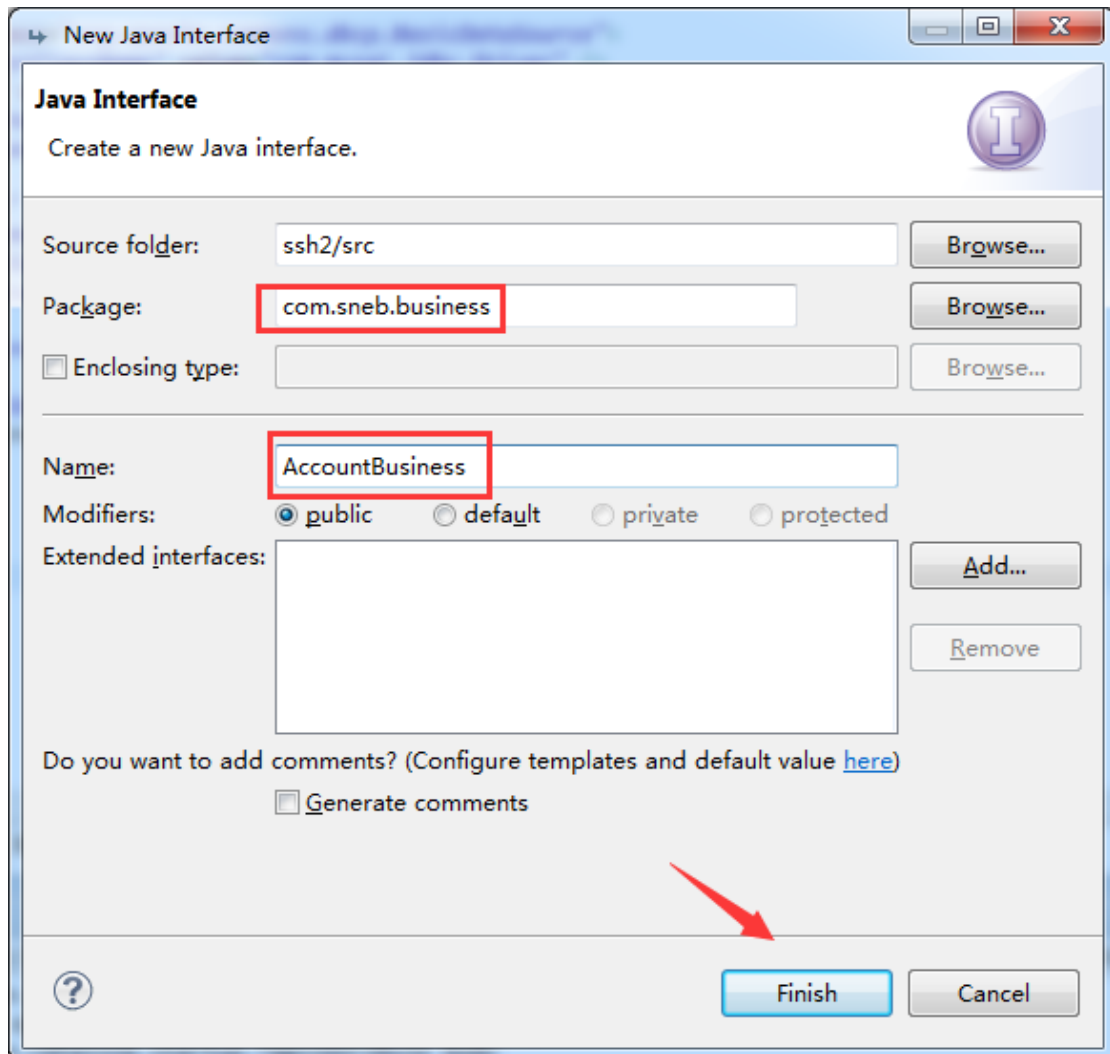


## 6.6 Spring 注入配置

要实现的目的, AccountAction 控制器调用 AccountBusiness 的方法, 返回字符串到控制器, 并显示到视图层。

### 6.6.1 定义业务逻辑接口

com.sneb.business 目录上右键 → New Interface



AccountBusiness.java 完整代码如下：

----- 内容 -----

```
package com.sneb.business;

public interface AccountBusiness {
    public String findAll();
}
```

### 6.6.2 业务接口实现

com.sneb.business.impl 目录下新建 class 文件，完整 AccountBusinessImpl.java 代码如下：

----- 内容 -----

```
package com.sneb.business.impl;
import com.sneb.business.AccountBusiness;

public class AccountBusinessImpl implements AccountBusiness{
```

```

@Override
public String findAll() {
    // TODO Auto-generated method stub
    return "Business Return!";
}

```

### 6.6.3 Spring 注入配置

多 Spring 配置文件的配置：

applicationContext.xml 另存为 applicationContext-Account.xml

编辑内容如下：

```

AccountBusiness.java  AccountBusinessImpl.java  applicationContext-Account.xml x
1  <?xml version="1.0" encoding="UTF-8"?>
2  <beans xmlns="http://www.springframework.org/schema/beans"
3      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:p="http://www.springframework.org/schema/p"
4      xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-3.1.xsd">
5
6
7
8  </beans>

```

然后定义 AccountAction 控制器的 bean 和 AccountBusiness 的 bean，并在 AccountAction 的 bean 里注入 AccountBusiness 的 bean。

注意重启一下 myeclipse ， 否则有时接下来的操作无效。


#### 6.6.3.1 生成 AccountBusiness 的 bean


id 取名为：accountBusiness，操作步骤如下：

applicationContext-Account.xml 上右键 → Sping → New Bean →

Bean Wizard

New Spring Bean



 Bean class is empty, generated bean will be implicitly abstract

Bean Id:

Name:

Add...

Creation method: ☒ Default ☐ Factory bean ☐ Static factory method

Bean class:

Browse...

Parent bean Id:

Abstract: ☐

Lazy init:

Scope:

Autowire:

Constructor Args

Properties

Dependencies

Life-cycle

Arguments:

Index	Class	Spring type	Value

Add...

Edit...

Remove

Pick Constructor...



< Back

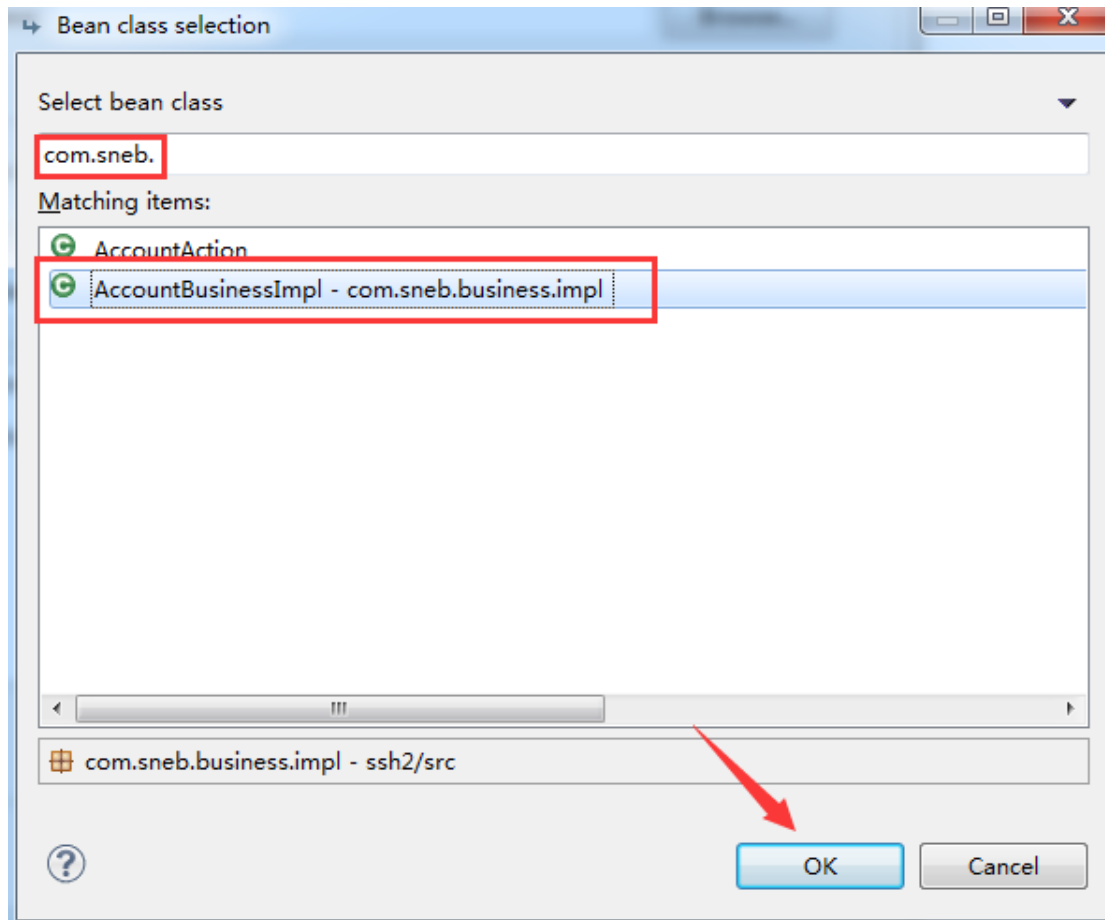
Next >

Finish

Cancel

→





→

Bean Wizard

### New Spring Bean

Create a new Spring bean

Bean Id:

Name:

Creation method: ☒ Default ☐ Factory bean ☐ Static factory method

Bean class:

Parent bean Id:

Abstract: ☐ Lazy init:

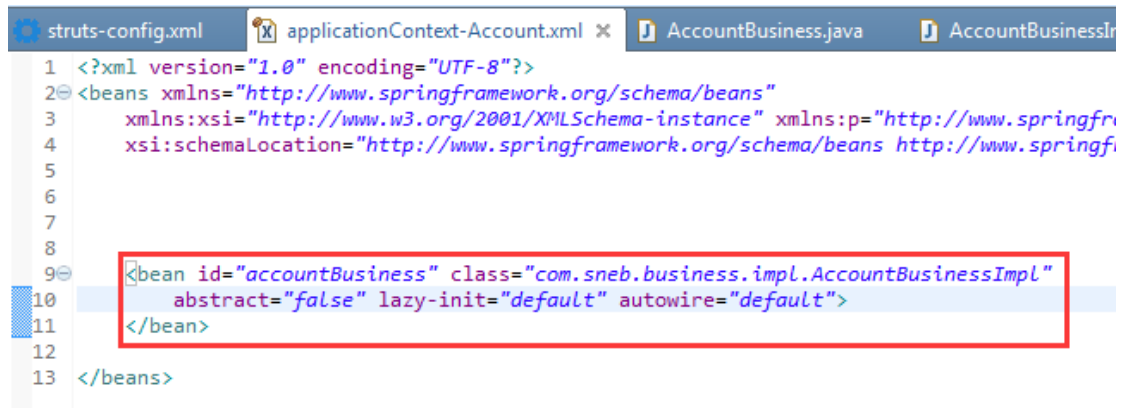
Scope:  Autowire:

Constructor Args Properties Dependencies Life-cycle

Arguments:

Index	Class	Spring type	Value

操作完成后，会自动在配置文件里增加 bean 的定义配置：



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:p="http://www.springfr
4       xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springfi
5
6
7
8
9 <bean id="accountBusiness" class="com.sneb.business.impl.AccountBusinessImpl"
10       abstract="false" lazy-init="default" autowire="default">
11 </bean>
12
13 </beans>
```

### 6.6.3.2 生成 AccountAction 的 bean 配置

applicationContext-Account.xml 上右键→ Spring→ New Bean →

Bean Wizard

### New Spring Bean

Create a new Spring bean

Bean Id:

Name:

Creation method: ☒ Default ☐ Factory bean ☐ Static factory method

Bean class:

Parent bean Id:

Abstract: ☐ Lazy init:

Scope:  Autowire:

Constructor Args **Properties** Dependencies Life-cycle

Name	Type	Value

→

**Property Wizard**

**New Property**  
Add a new property


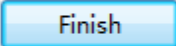

Name:

Spring type:

Property format: ☒ p-namespace ☐ Element

Reference type: ☒ Bean ☐ Local ☐ Parent

Reference:

→Finish

配置如下:

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:p="http://www.springframework.org/schema/beans"
4     xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans"
5
6
7     <bean id="accountBusiness" class="com.sneb.business.impl.AccountBusinessImpl"
8         abstract="false" lazy-init="default" autowire="default">
9     </bean>
10
11
12     <bean id="account" name="/account" class="com.sneb.struts.action.AccountAction"
13         abstract="false" lazy-init="default" autowire="default"
14         p:accountBusiness-ref="accountBusiness">
15     </bean>
16
17
18 </beans>

```

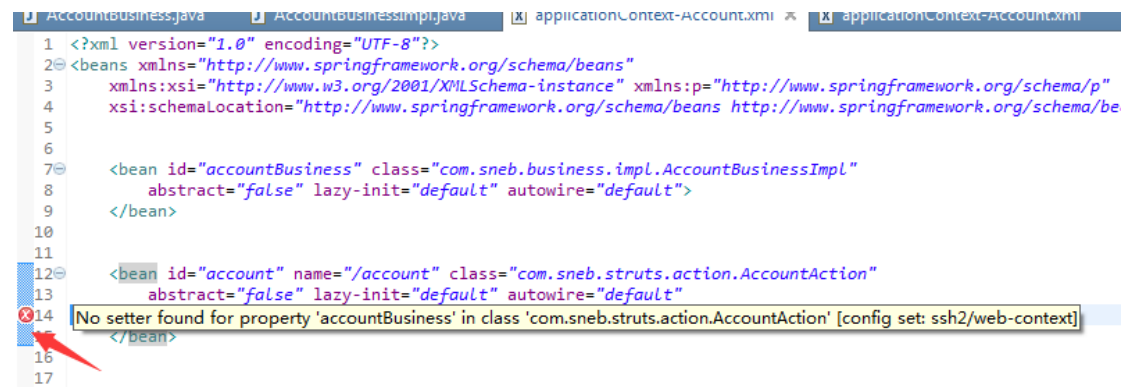
### 6.6.3.3 修改 web.xml 关联 applicationContext-Account.xml

```
3 <context-param>
4   <param-name>contextConfigLocation</param-name>
5   <param-value>
6     classpath:applicationContext.xml,
7     classpath:applicationContext-Account.xml
8   </param-value>
9 </context-param>
10 </web-app>
```

多个 Spring 配置文件用逗号分割。

### 6.6.3.4 控制器 AccountAction 里实现 setter 方法

注意上一小节中配置文件左边有个错误提示，点击查看：



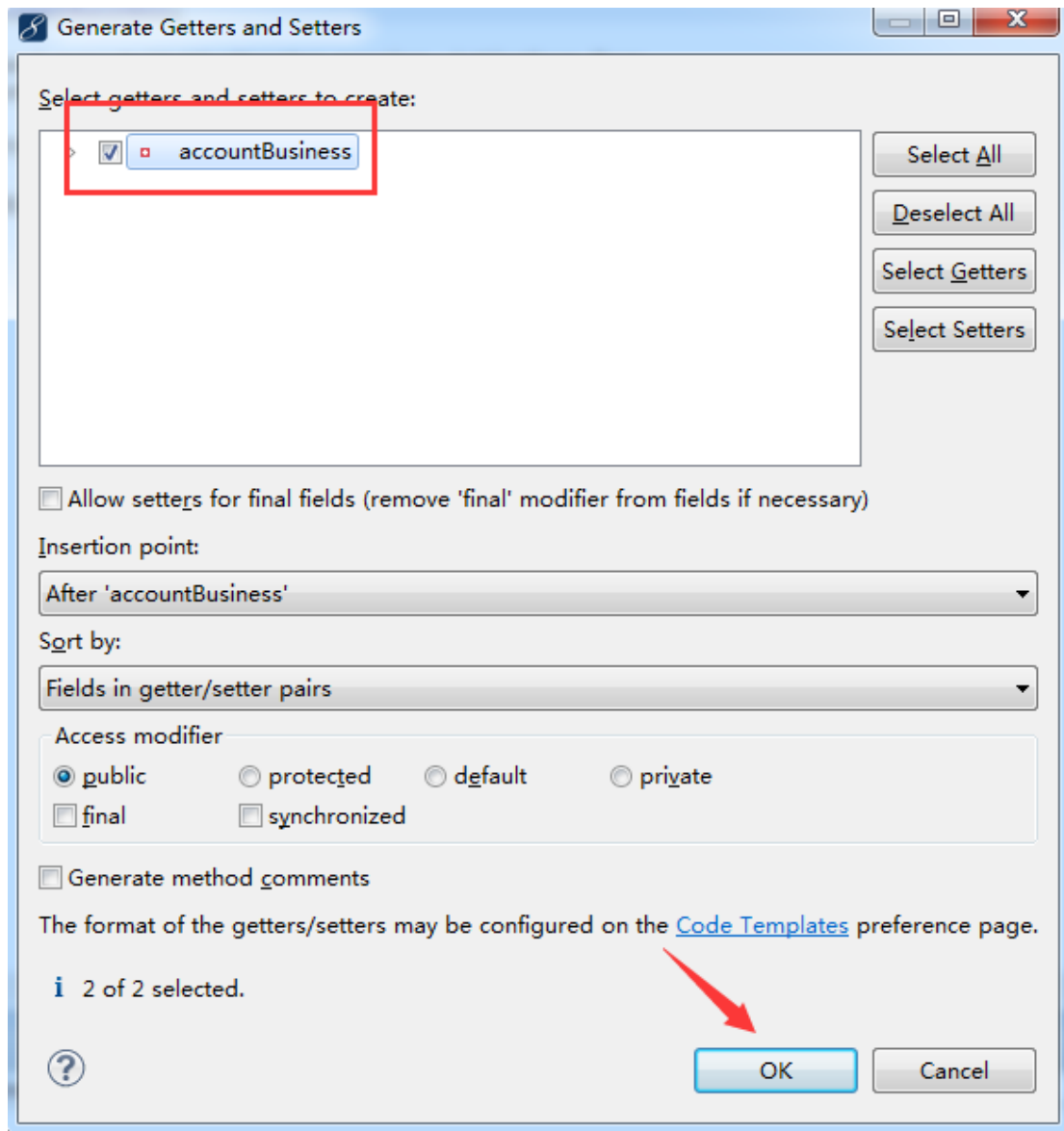
```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:p="http://www.springframework.org/schema/p"
4       xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/be
5
6
7 <bean id="accountBusiness" class="com.sneb.business.impl.AccountBusinessImpl"
8       abstract="false" lazy-init="default" autowire="default">
9 </bean>
10
11
12 <bean id="account" name="/account" class="com.sneb.struts.action.AccountAction"
13       abstract="false" lazy-init="default" autowire="default">
14 </bean>
```

意思是让我们在控制器里定义属性 accountBusiness 的 setter 方法，因为注入时，ioc 容器需要。

在 AccountAction 里定义私有变量：

```
private AccountBusiness accountBusiness;
```

然后右键 → Source → Generate Getters and Setters... →



→ Ok。将自动生成 get 和 set 方法。

同时修改一下代码，调用 **AccountBusiness** 类的 **findAll()**方法。

AccountAction.java 完整代码如下：

----- 内容 -----

```
/*
 * Generated by MyEclipse Struts
 * Template path: templates/java/JavaClass.vtl
 */
package com.sneb.struts.action;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
```

```

import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;

import com.sneb.business.AccountBusiness;

/**
 * MyEclipse Struts
 * Creation date: 11-27-2016
 *
 * XDoclet definition:
 * @struts.action validate="true"
 * @struts.action-forward name="show" path="/index.jsp"
 */
public class AccountAction extends Action {

    private AccountBusiness accountBusiness;

    /**
     * Generated Methods
     */

    public AccountBusiness getAccountBusiness() {
        return accountBusiness;
    }

    public void setAccountBusiness(AccountBusiness accountBusiness) {
        this.accountBusiness = accountBusiness;
    }

    /**
     * Method execute
     * @param mapping
     * @param form
     * @param request
     * @param response
     * @return ActionForward
     */
    public ActionForward execute(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response) {
        // TODO Auto-generated method stub
        String msg = accountBusiness.findAll();
        request.setAttribute("data", msg);
        return mapping.findForward("show");
    }
}

```



}

----- 内容结束 -----

关键代码截图：

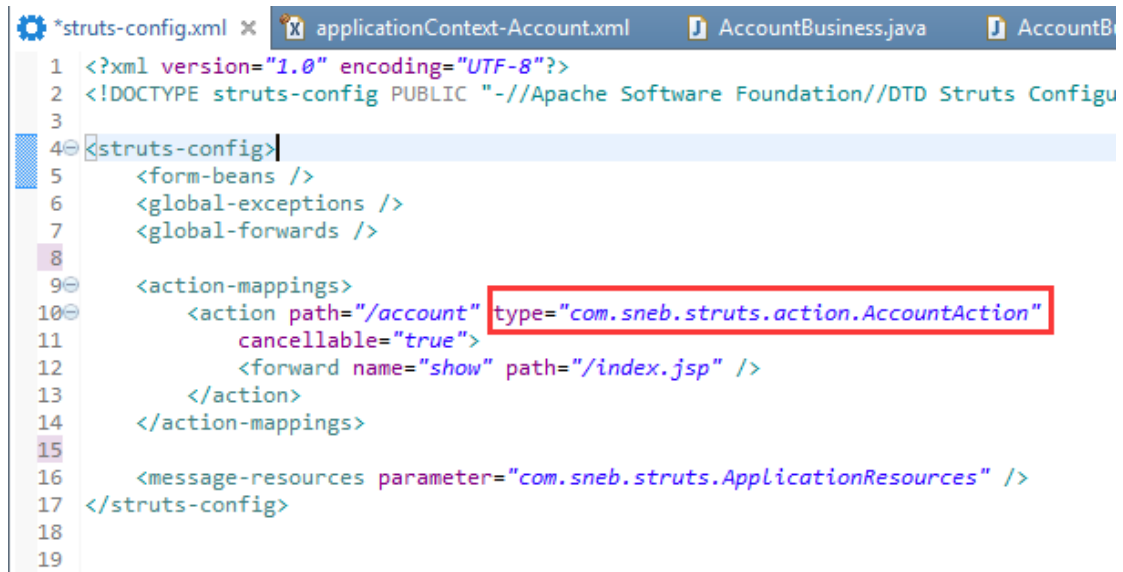


```
20 * Generated by MyEclipse Struts
5 package com.sneb.struts.action;
6
7 import javax.servlet.http.HttpServletRequest;
16
18 * MyEclipse Struts
25 public class AccountAction extends Action {
26
27     private AccountBusiness accountBusiness;
28
29     /*
30      * Generated Methods
31      */
32
33     public AccountBusiness getAccountBusiness() {
34         return accountBusiness;
35     }
36
37     public void setAccountBusiness(AccountBusiness accountBusiness) {
38         this.accountBusiness = accountBusiness;
39     }
40
41
42     * Method execute
49     public ActionForward execute(ActionMapping mapping, ActionForm form,
50         HttpServletRequest request, HttpServletResponse response) {
51         // TODO Auto-generated method stub
52
53         String msg = accountBusiness.findAll();
54         request.setAttribute("data", msg);
55
56         return mapping.findForward("show");
57     }
58 }
```

### 6.6.3.5 修改 struts 配置

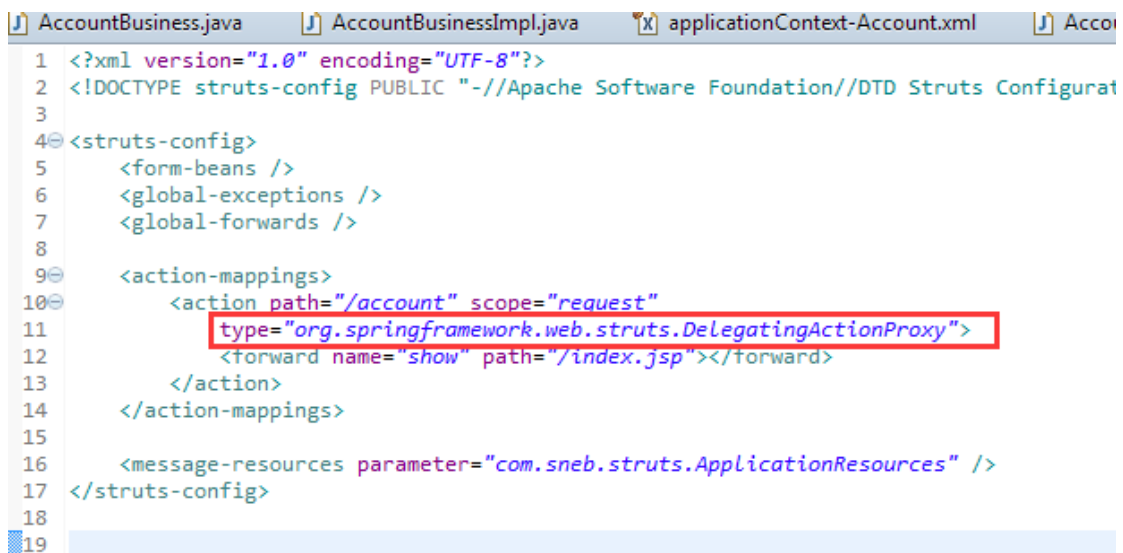
将控制权交给 spring，修改  
com.sneb.struts.action.AccountAction  
为  
org.springframework.web.struts.DelegatingActionProxy

修改前截图：



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE struts-config PUBLIC "-//Apache Software Foundation//DTD Struts Configur
3
4 <struts-config>
5     <form-beans />
6     <global-exceptions />
7     <global-forwards />
8
9     <action-mappings>
10         <action path="/account" type="com.sneb.struts.action.AccountAction"
11             cancellable="true">
12             <forward name="show" path="/index.jsp" />
13         </action>
14     </action-mappings>
15
16     <message-resources parameter="com.sneb.struts.ApplicationResources" />
17 </struts-config>
18
19
```

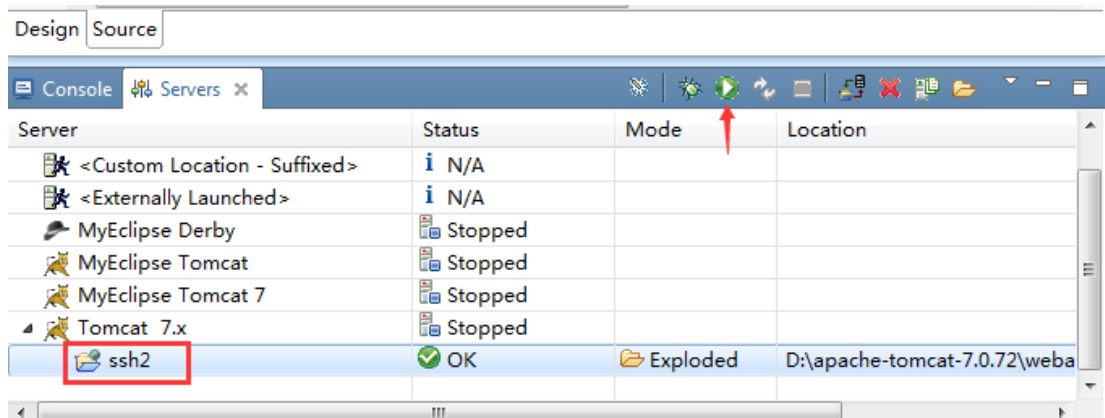
修改后截图：



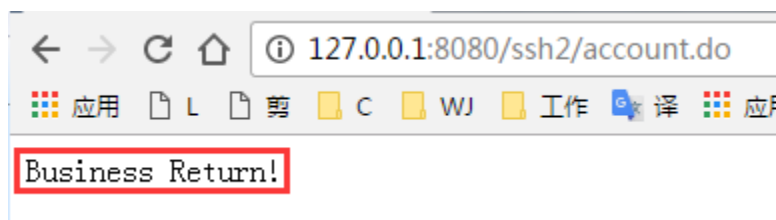
```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE struts-config PUBLIC "-//Apache Software Foundation//DTD Struts Configurati
3
4 <struts-config>
5     <form-beans />
6     <global-exceptions />
7     <global-forwards />
8
9     <action-mappings>
10         <action path="/account" scope="request"
11             type="org.springframework.web.struts.DelegatingActionProxy">
12             <forward name="show" path="/index.jsp"></forward>
13         </action>
14     </action-mappings>
15
16     <message-resources parameter="com.sneb.struts.ApplicationResources" />
17 </struts-config>
18
19
```

### 6.3.3.6 启动服务查看效果

启动截图：



效果截图：



## 6.7 HQL 读取表数据

HQL 跟 SQL 语言长得很像，但是不要弄混，HQL 是对 bean 对象的操作，不是直接对数据库操作。

Hibernate 查询语言（HQL）是一种面向对象的查询语言，类似于 SQL，但不是去对表和列进行操作，而是面向对象和它们的属性。HQL 查询被 Hibernate 翻译为传统的 SQL 查询从而对数据库进行操作。

尽管你能直接使用本地 SQL 语句，但我还是建议你尽可能的使用 HQL 语句，以避免数据库关于可移植性的麻烦，并且体现了 Hibernate 的 SQL 生成和缓存策略。

在 HQL 中一些关键字比如 SELECT，FROM 和 WHERE 等，是不区分大小写的，但是是一些属性比如表名和列名是区分大小写的。

参考：

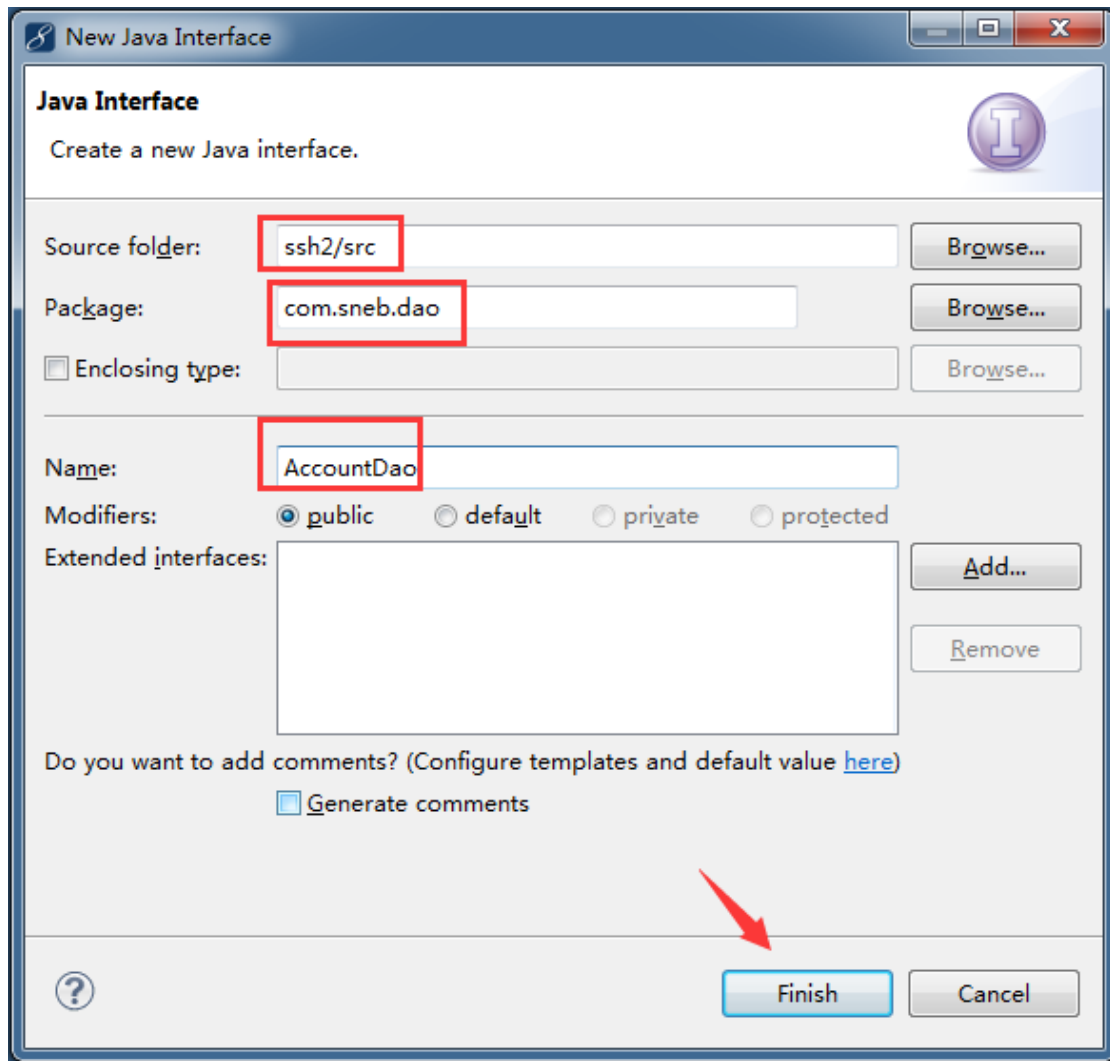
<http://wiki.jikexueyuan.com/project/hibernate/query-language.html>

### 6.7.1 创建数据库操作 dao 的接口

接下来在 com.sneb.dao 目录下，新建 AccountDao 接口类，并定义 findAll() 方法，用来返回数据库 account 表的数据。

详细操作步骤如下：

New → Interface



➔ Finish

----- 内容 -----

```
package com.sneb.dao;  
import java.util.List;  
import com.sneb.bean.Account;  
  
public interface AccountDao {  
    public List<Account> findAll();  
}
```

## 6.7.2 数据库操作类 dao 的实现

com.sneb.dao.impl 目录上右键 ➔ New Class ➔

**New Java Class**

**Java Class**  
Create a new Java class.

Source folder:

Package:

☐ Enclosing type:

Name:

Modifiers: ☒ public ☐ default ☐ private ☐ protected  
☐ abstract ☐ final ☐ static

Superclass:

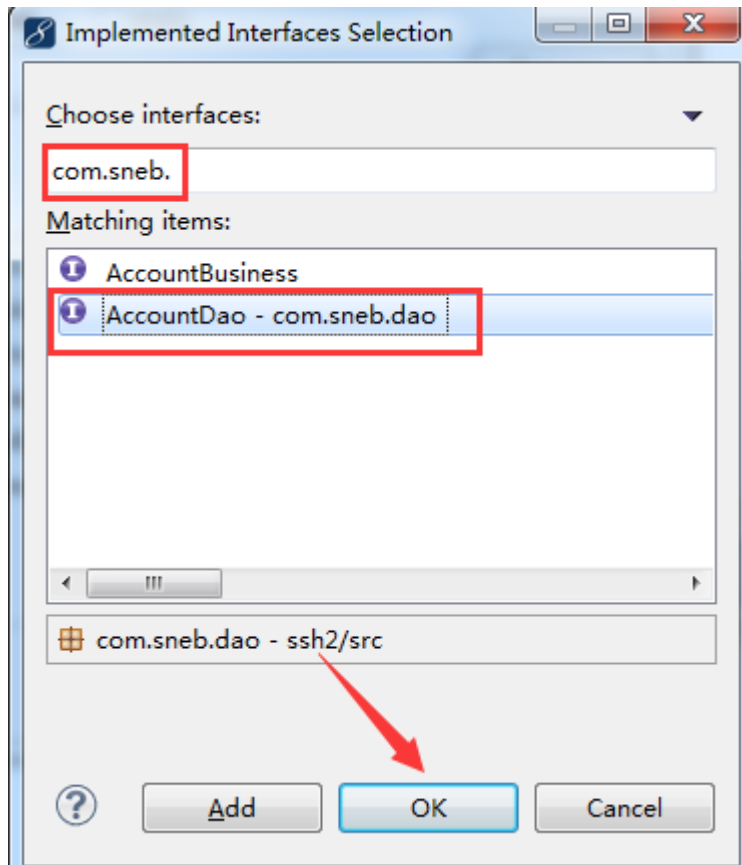
Interfaces:

Which method stubs would you like to create?

☐ public static void main(String[] args)  
☐ Constructors from superclass  
☒ Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))  
☐ Generate comments

→



➔ Finish

然后编写 HQL 语句，并调用 hibernate 的方法。完整代码如下：

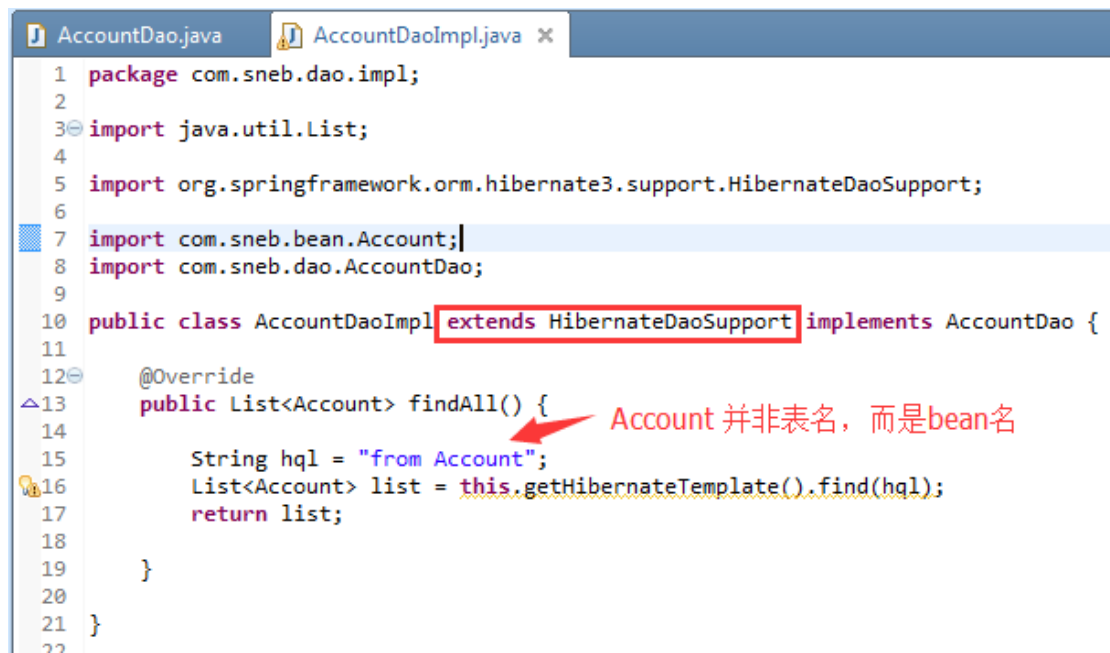
----- 内容 -----

```
package com.sneb.dao.impl;
import java.util.List;
import org.springframework.orm.hibernate3.support.HibernateDaoSupport;
import com.sneb.bean.Account;
import com.sneb.dao.AccountDao;
```

```
public class AccountDaoImpl extends HibernateDaoSupport implements
AccountDao {
```

```
    @Override
    public List<Account> findAll() {
        String hql = "from Account";
        List<Account> list = this.getHibernateTemplate().find(hql);
        return list;
    }
}
```

截图：



```
1 package com.sneb.dao.impl;
2
3 import java.util.List;
4
5 import org.springframework.orm.hibernate3.support.HibernateDaoSupport;
6
7 import com.sneb.bean.Account;
8 import com.sneb.dao.AccountDao;
9
10 public class AccountDaoImpl extends HibernateDaoSupport implements AccountDao {
11
12     @Override
13     public List<Account> findAll() {
14         String hql = "from Account";
15         List<Account> list = this.getHibernateTemplate().find(hql);
16         return list;
17     }
18 }
19
20
21 }
```

Account 并非表名，而是bean名

### 6.7.3 Srping 里配置 bean 及 bean 间的注入关系

我们要配置 bean: AccountDao，并注入 hibernate 的数据库 bean，操作步骤如下：  
applicationContext-Account.xml → 右键 → Spring → New Bean →

**Bean Wizard**

### New Spring Bean

Create a new Spring bean

Bean Id:

Name:

Creation method: ☒ Default ☐ Factory bean ☐ Static factory method

Bean class:

Parent bean Id:

Abstract: ☐ Lazy init:

Scope:  Autowire:

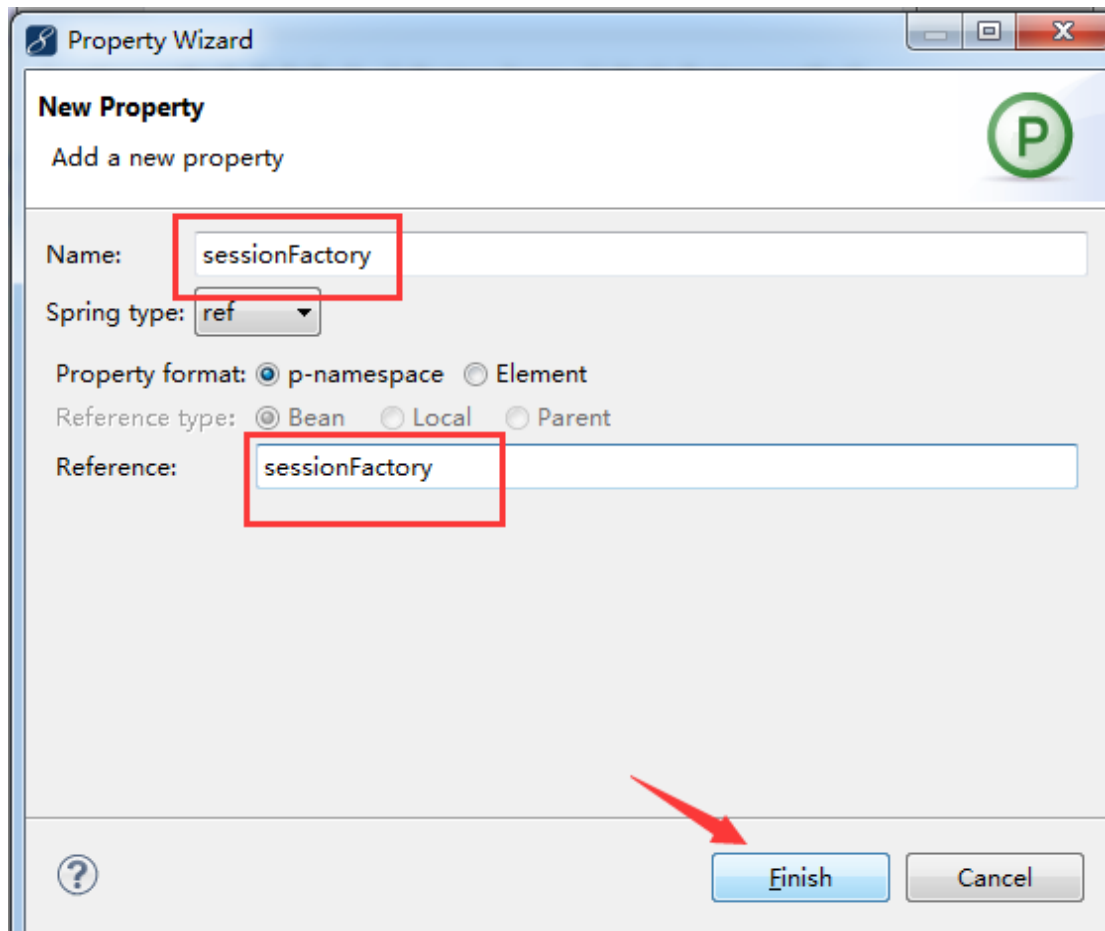
Constructor Args **Properties** Dependencies Life-cycle

Properties:

Name	Type	Value

→





注意，sessionFactory 是注入 applicationContext.xml 里定义的数据库连接的 bean id  
代码如下：

```
<bean id="accountDao" class="com.sneb.dao.impl.AccountDaoImpl"
      abstract="false" lazy-init="default" autowire="default"
      p:sessionFactory-ref="sessionFactory">
</bean>
```

#### 6.7.4 在 accountBusiness bean 里注入 accountDao

applicationContext-Account.xml 最后完整代码如下：

----- 内容 -----

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:p="http://www.springframework.org/schema/p"
       xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-3.1.xsd">
```

```

    <bean id="accountBusiness"
class="com.sneb.business.impl.AccountBusinessImpl"
    abstract="false" lazy-init="default" autowire="default"
    p:accountDao-ref="accountDao">
    </bean>

    <bean id="account" name="/account"
class="com.sneb.struts.action.AccountAction"
    abstract="false" lazy-init="default" autowire="default"
    p:accountBusiness-ref="accountBusiness">
    </bean>

    <bean id="accountDao" class="com.sneb.dao.impl.AccountDaoImpl"
    abstract="false" lazy-init="default" autowire="default"
    p:sessionFactory-ref="sessionFactory">
    </bean>

</beans>

```

## 6.7.5 重构一系列返回值类型

由于先前 hello world 例子，返回值类型为字符串，现在要从数据库读数据显示出来，类型自然为 List<Account>类型，所以相应返回值的地方均需要修改。

需要修改的类有：AccountBusinessImpl、AccountBusiness、AccountAction 和 index.jsp 文件，包括业务逻辑的接口定义及实现类，控制器和视图文件。

### 6.7.5.1 重构业务逻辑接口 AccountBusiness

上一节例子中，业务逻辑类 AccountBusiness 的 findAll() 方法，是直接返回的字符串，现在我们要调用 AccountDao 的 findAll() 方法，从数据库读取数据，现在重构 AccountBusiness 接口类的返回值为 List<Account>对象。

```

----- 内容 -----
package com.sneb.business;
import java.util.List;
import com.sneb.bean.Account;

public interface AccountBusiness {
    public List<Account> findAll();
}

```

### 6.7.5.2 重构业务逻辑实现类 AccountBusinessImpl

首先增加 accountDao 属性及 setter 方法，并修改返回值类型为 List<Account>，以前是 String，完整代码如下：

----- 内容 -----

```
package com.sneb.business.impl;
import java.util.List;
import com.sneb.bean.Account;
import com.sneb.business.AccountBusiness;
import com.sneb.dao.AccountDao;

public class AccountBusinessImpl implements AccountBusiness {
    private AccountDao accountDao;
    public AccountDao getAccountDao() {
        return accountDao;
    }

    public void setAccountDao(AccountDao accountDao) {
        this.accountDao = accountDao;
    }
    @Override
    public List<Account> findAll() {
        List<Account> list = accountDao.findAll();
        return list;
    }
}
```

### 6.7.5.3 重构控制器返回值类型

AccountAction.java

----- 内容 -----

```
/*
 * Generated by MyEclipse Struts
 * Template path: templates/java/JavaClass.vtl
 */
package com.sneb.struts.action;

import java.util.List;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
```

```

import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;

import com.sneb.bean.Account;
import com.sneb.business.AccountBusiness;

/**
 * MyEclipse Struts
 * Creation date: 11-27-2016
 *
 * XDoclet definition:
 * @struts.action validate="true"
 * @struts.action-forward name="show" path="/index.jsp"
 */
public class AccountAction extends Action {

    private AccountBusiness accountBusiness;

    /**
     * Generated Methods
     */

    public AccountBusiness getAccountBusiness() {
        return accountBusiness;
    }

    public void setAccountBusiness(AccountBusiness accountBusiness) {
        this.accountBusiness = accountBusiness;
    }

    /**
     * Method execute
     * @param mapping
     * @param form
     * @param request
     * @param response
     * @return ActionForward
     */
    public ActionForward execute(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response) {
        // TODO Auto-generated method stub

```

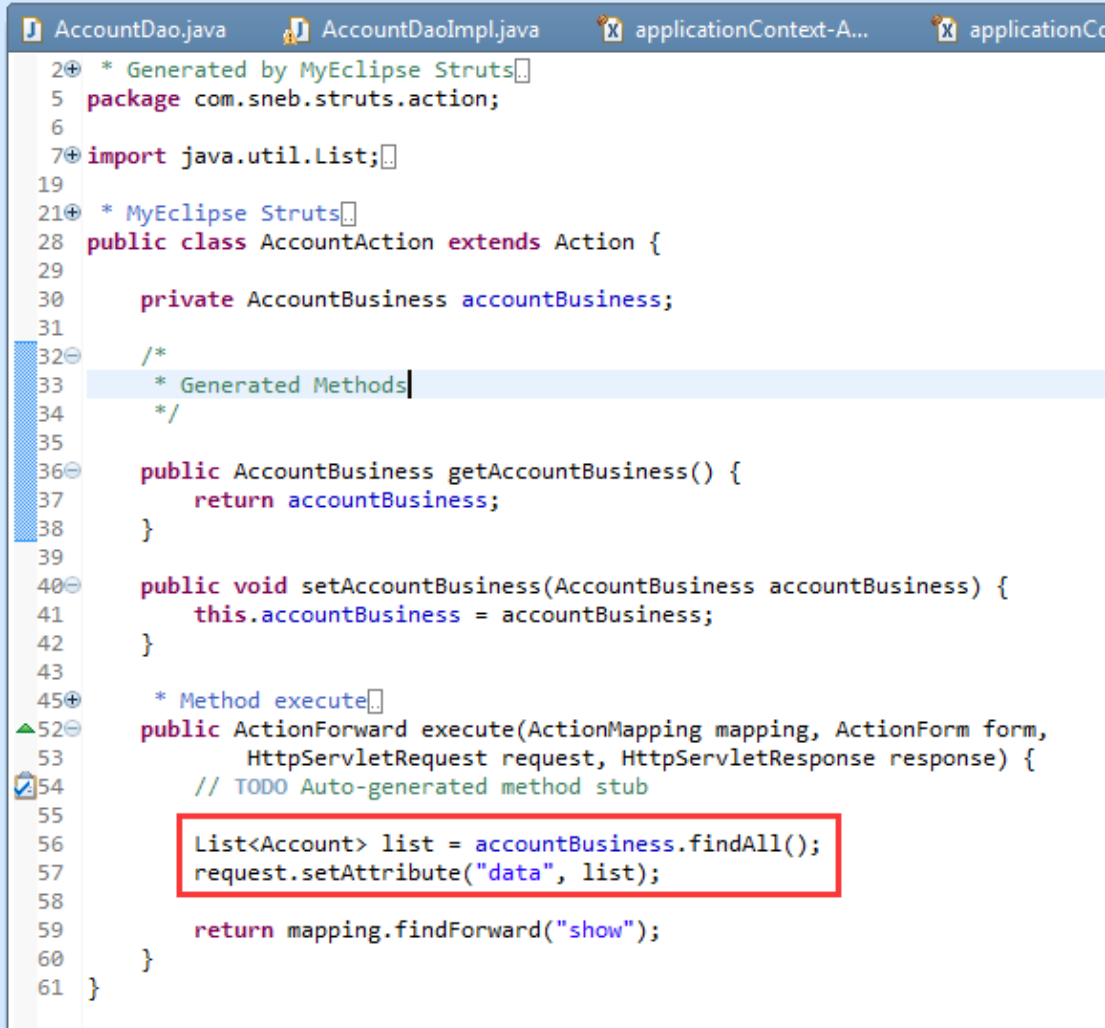
```

        List<Account> list = accountBusiness.findAll();
        request.setAttribute("data", list);

        return mapping.findForward("show");
    }
}

```

关键代码截图：



```

2+ * Generated by MyEclipse Struts
5 package com.sneb.struts.action;
6
7+ import java.util.List;
19
21+ * MyEclipse Struts
28 public class AccountAction extends Action {
29
30     private AccountBusiness accountBusiness;
31
32     /*
33      * Generated Methods
34      */
35
36     public AccountBusiness getAccountBusiness() {
37         return accountBusiness;
38     }
39
40     public void setAccountBusiness(AccountBusiness accountBusiness) {
41         this.accountBusiness = accountBusiness;
42     }
43
45     * Method execute
52+ public ActionForward execute(ActionMapping mapping, ActionForm form,
53     HttpServletRequest request, HttpServletResponse response) {
54     // TODO Auto-generated method stub
55
56     List<Account> list = accountBusiness.findAll();
57     request.setAttribute("data", list);
58
59     return mapping.findForward("show");
60 }
61 }

```

#### 6.7.5.4 重构视图文件 index.jsp

需要引入 struts 的标签文件。

```

<%@ taglib prefix="c" uri="http://java.sun.com/jstl/core_rt"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>

```

再要修改字符编码为 UTF-8，最后用 struts 逻辑循环标签，显示 list 里的数据。

完整代码如下：

```
----- 内容 -----
<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>

<%@ taglib prefix="c" uri="http://java.sun.com/jstl/core_rt"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>

<%
    String path = request.getContextPath();
    String basePath = request.getScheme() + "://"
        + request.getServerName() + ":" + request.getServerPort()
        + path + "/";
%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<base href="<%=basePath%>">

<title>My JSP 'index.jsp' starting page</title>
<meta http-equiv="pragma" content="no-cache">
<meta http-equiv="cache-control" content="no-cache">
<meta http-equiv="expires" content="0">
<meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
<meta http-equiv="description" content="This is my page">
<!--
    <link rel="stylesheet" type="text/css" href="styles.css">
-->
</head>

<body>

    <c:forEach var="list" items="${data}">
        ${list.userId} --- ${list.userEmail} <br />
    </c:forEach>

</body>
</html>
```

截图：

```

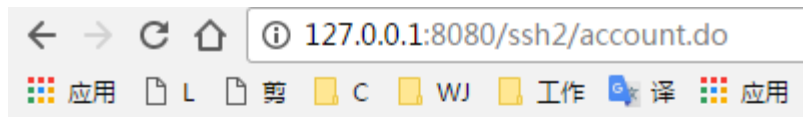
1  <%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>
2
3  <%@ taglib prefix="c" uri="http://java.sun.com/jstl/core_rt"%>
4  <%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>
5
6  <%
7      String path = request.getContextPath();
8      String basePath = request.getScheme() + "://"
9          + request.getServerName() + ":" + request.getServerPort()
10         + path + "/";
11 %>
12
13 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
14 <html>
15 <head>
16 <base href="<%=basePath%>">
17
18 <title>My JSP 'index.jsp' starting page</title>
19 <meta http-equiv="pragma" content="no-cache">
20 <meta http-equiv="cache-control" content="no-cache">
21 <meta http-equiv="expires" content="0">
22 <meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
23 <meta http-equiv="description" content="This is my page">
24 <!--
25 <link rel="stylesheet" type="text/css" href="styles.css">
26 -->
27 </head>
28
29 <body>
30
31 <c:forEach var="list" items="${data}">
32     ${list.userId} --- ${list.userEmail} <br />
33 </c:forEach>
34
35 </body>
36 </html>
37

```

请注意红框中的修改。

### 6.7.5.5 重启检验效果

<http://127.0.0.1:8080/ssh2/account.do>



```
146418944808873960036 --- 18900000000@qq.com
146418944808890520068 --- 18900000001@qq.com
146418944808897490048 --- 18900000002@qq.com
146418944808903700042 --- 18900000003@qq.com
146418944808910120006 --- 18900000004@qq.com
146418944808916440007 --- 18900000005@qq.com
146418944808922630046 --- 18900000006@qq.com
146418944808930490074 --- 18900000007@qq.com
146418944808935390054 --- 18900000008@qq.com
146418944808941230075 --- 18900000009@qq.com
146418944808946470050 --- 18900000010@qq.com
146418944808951990066 --- 18900000011@qq.com
146418944808956610018 --- 18900000012@qq.com
146418944808961230090 --- 18900000013@qq.com
146418944808965660020 --- 18900000014@qq.com
```

## 7 综合例子（二）

本章例子将完整实现对数据库的增删改操作

### 7.1 增加 **actionForm**

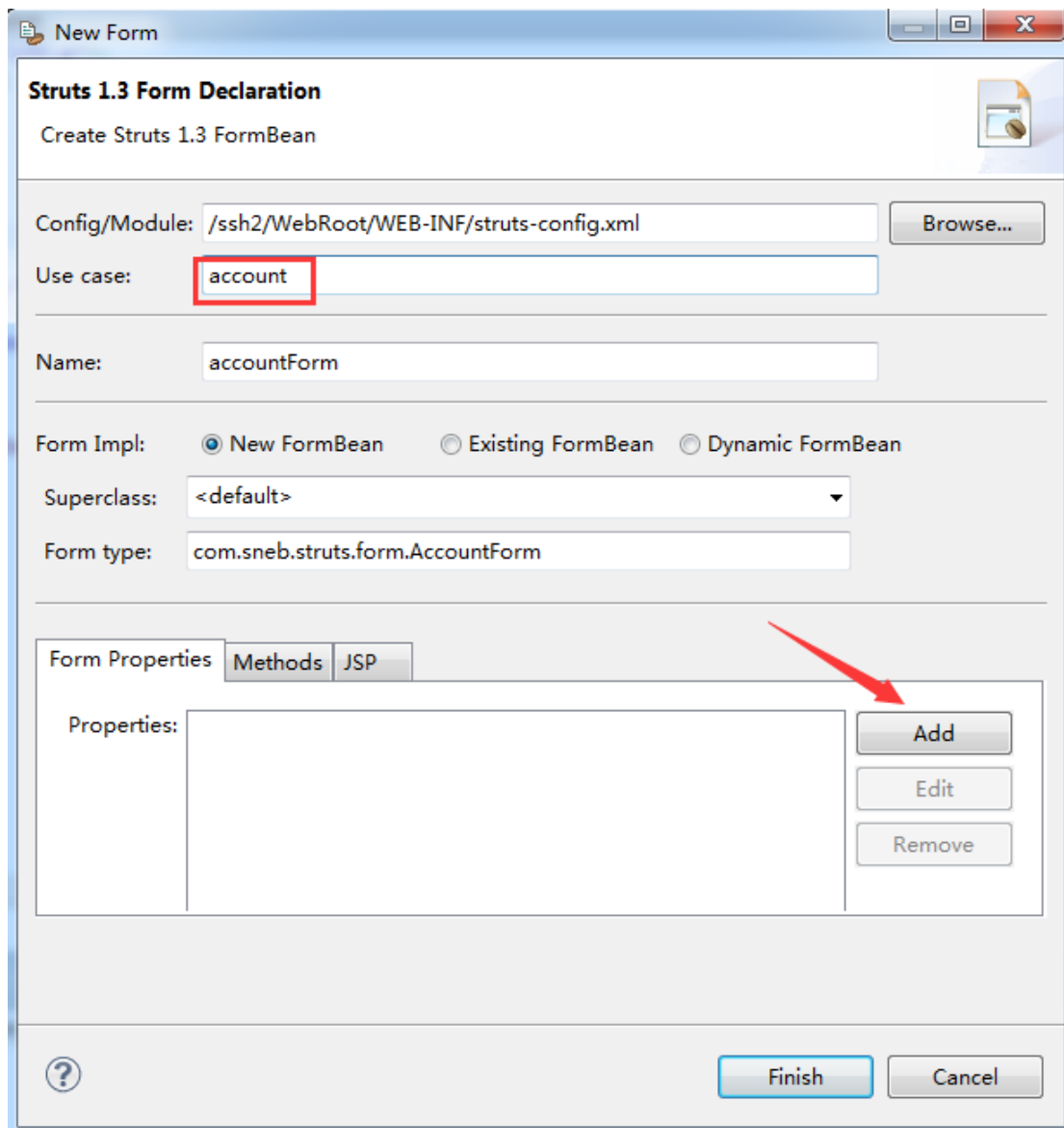
ActionForm 是控制器与视图层传递数据的重要 bean。

#### 7.1.1 创建 ActionForm

详细步骤：

Struts-config.xml 上右键 → New → Form→

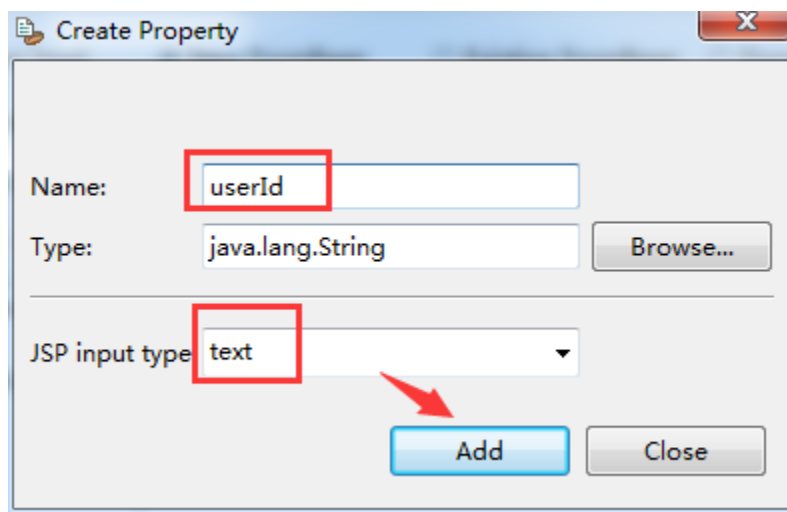




The 'New Form' dialog box for Struts 1.3 Form Declaration. It contains the following fields and controls:

- Config/Module:** /ssh2/WebRoot/WEB-INF/struts-config.xml (with a 'Browse...' button)
- Use case:** account (highlighted with a red box)
- Name:** accountForm
- Form Impl:** ☒ New FormBean, ☐ Existing FormBean, ☐ Dynamic FormBean
- Superclass:** <default> (dropdown menu)
- Form type:** com.sneb.struts.form.AccountForm
- Form Properties** tab is selected, showing a 'Properties:' list and three buttons: 'Add', 'Edit', and 'Remove'. A red arrow points to the 'Add' button.
- At the bottom are 'Finish' and 'Cancel' buttons.

→ 添加属性（对应数据表的字段）



The 'Create Property' dialog box. It contains the following fields and controls:

- Name:** userId (highlighted with a red box)
- Type:** java.lang.String (with a 'Browse...' button)
- JSP input type:** text (highlighted with a red box)
- At the bottom are 'Add' and 'Close' buttons. A red arrow points to the 'Add' button.

注意添加一个操作标志字段 operateType 用来从视图层传递增删改查的操作类型到控制器，

所有属性添加完后 Close。

**New Form**

**Struts 1.3 Form Declaration**

Create Struts 1.3 FormBean

Config/Module: /ssh2/WebRoot/WEB-INF/struts-config.xml Browse...

Use case: account

Name: accountForm

Form Impl: ☒ New FormBean ☐ Existing FormBean ☐ Dynamic FormBean

Superclass: <default>

Form type: com.sneb.struts.form.AccountForm

**Form Properties** Methods JSP

Properties:

- operateType - [java.lang.String] <html:text/>
- userId - [java.lang.String] <html:text/>
- userEmail - [java.lang.String] <html:text/>
- userMobile - [java.lang.String] <html:text/>
- userPwd - [java.lang.String] <html:text/>
- demo - [java.lang.String] <html:text/>

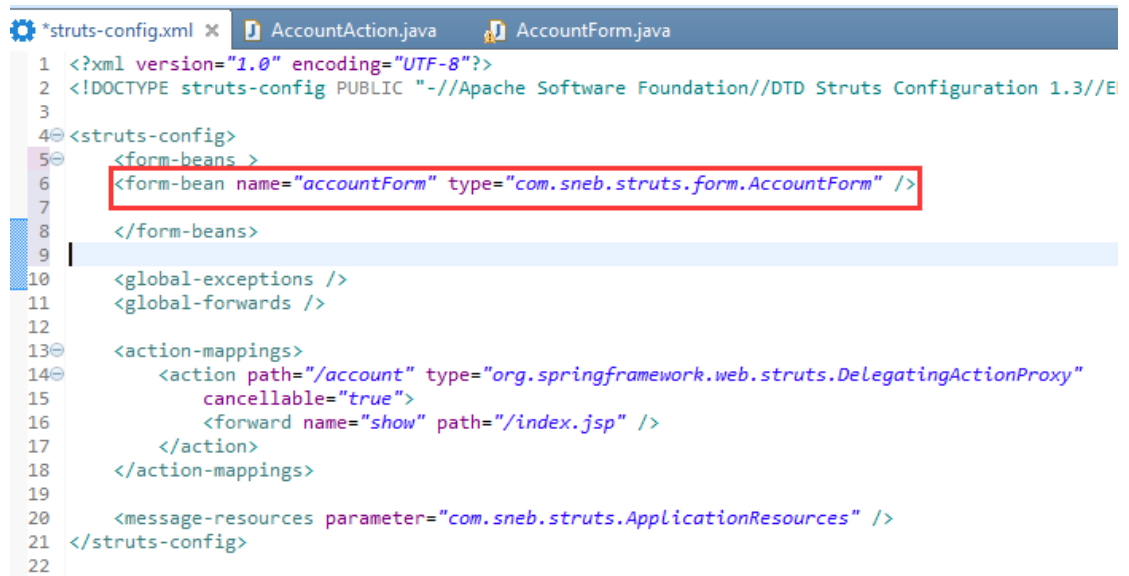
Add Edit Remove

? Finish Cancel

→

→ Finish

将会自动生成一个 AccountForm.java 类，同时在 struts-config.xml 里增加了一个 form-bean 的配置项。



```
*struts-config.xml AccountAction.java AccountForm.java
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE struts-config PUBLIC "-//Apache Software Foundation//DTD Struts Configuration 1.3//E
3
4 <struts-config>
5   <form-beans>
6     <form-bean name="accountForm" type="com.sneb.struts.form.AccountForm" />
7   </form-beans>
8
9   <global-exceptions />
10  <global-forwards />
11
12  <action-mappings>
13    <action path="/account" type="org.springframework.web.struts.DelegatingActionProxy"
14      cancellable="true">
15      <forward name="show" path="/index.jsp" />
16    </action>
17  </action-mappings>
18
19  <message-resources parameter="com.sneb.struts.ApplicationResources" />
20 </struts-config>
21
22
```

## 7.1.2 修改控制器

完整代码如下：

----- 内容 -----

```
/*
 * Generated by MyEclipse Struts
 * Template path: templates/java/JavaClass.vtl
 */
package com.sneb.struts.action;

import java.util.List;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;

import com.sneb.bean.Account;
import com.sneb.business.AccountBusiness;
import com.sneb.struts.form.AccountForm;

/**
 * MyEclipse Struts Creation date: 11-27-2016
```

```

*
* XDoclet definition:
*
* @struts.action validate="true"
* @struts.action-forward name="show" path="/index.jsp"
*/
public class AccountAction extends Action {

    private AccountBusiness accountBusiness;

    /**
     * Generated Methods
     */

    public AccountBusiness getAccountBusiness() {
        return accountBusiness;
    }

    public void setAccountBusiness(AccountBusiness accountBusiness) {
        this.accountBusiness = accountBusiness;
    }

    /**
     * Method execute
     *
     * @param mapping
     * @param form
     * @param request
     * @param response
     * @return ActionForward
     */
    public ActionForward execute(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response) {
        // TODO Auto-generated method stub

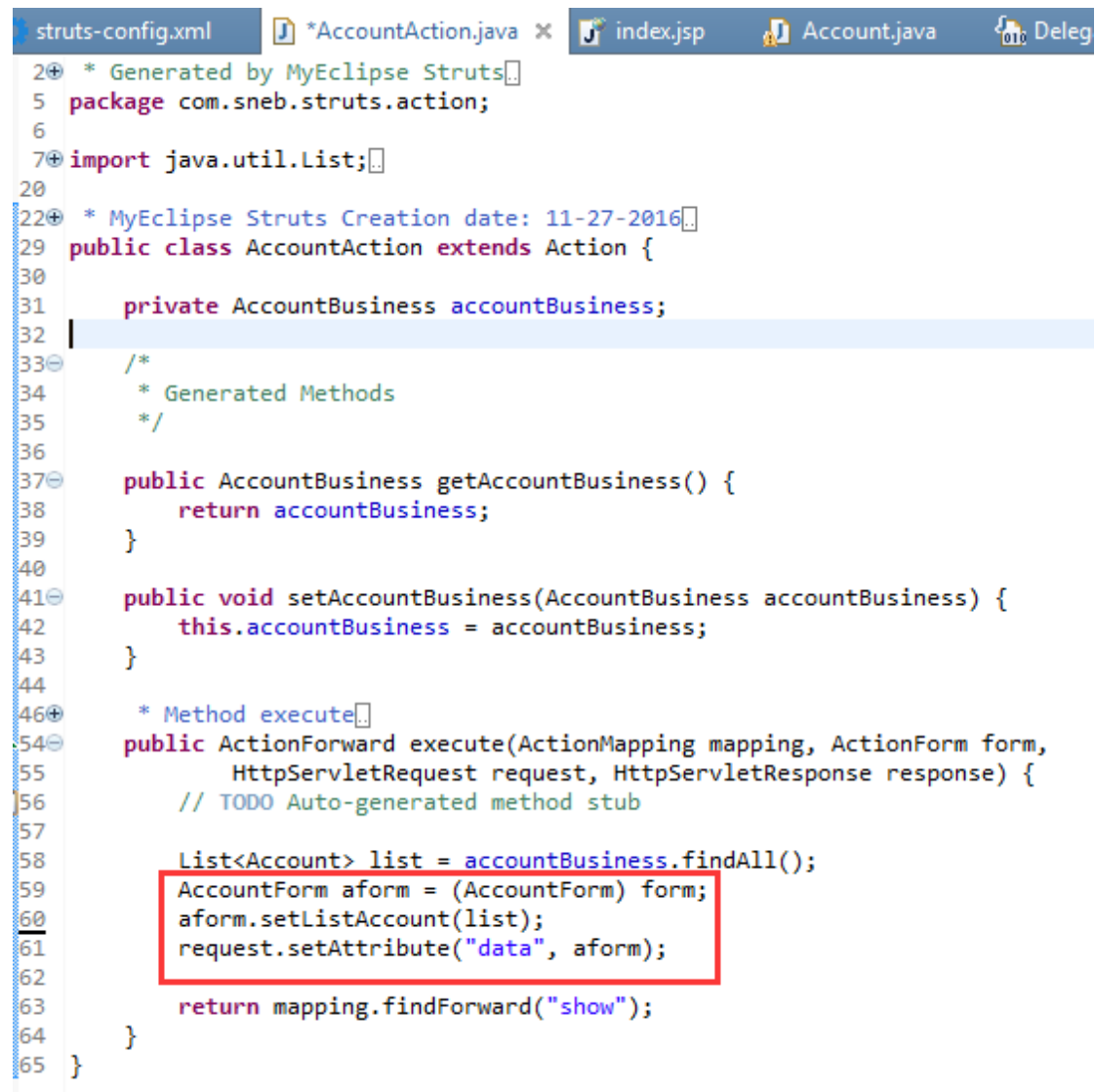
        List<Account> list = accountBusiness.findAll();
        AccountForm aform = (AccountForm) form;
        aform.setListAccount(list);

        request.setAttribute("data", aform);

        return mapping.findForward("show");
    }
}

```

截图:



```
struts-config.xml  *AccountAction.java x  index.jsp  Account.java  Deleg
2+ * Generated by MyEclipse Struts
5 package com.sneb.struts.action;
6
7+ import java.util.List;
20
22+ * MyEclipse Struts Creation date: 11-27-2016
29 public class AccountAction extends Action {
30
31     private AccountBusiness accountBusiness;
32
33     /*
34      * Generated Methods
35      */
36
37     public AccountBusiness getAccountBusiness() {
38         return accountBusiness;
39     }
40
41     public void setAccountBusiness(AccountBusiness accountBusiness) {
42         this.accountBusiness = accountBusiness;
43     }
44
46     * Method execute
54     public ActionForward execute(ActionMapping mapping, ActionForm form,
55         HttpServletRequest request, HttpServletResponse response) {
56         // TODO Auto-generated method stub
57
58         List<Account> list = accountBusiness.findAll();
59         AccountForm aform = (AccountForm) form;
60         aform.setListAccount(list);
61         request.setAttribute("data", aform);
62
63         return mapping.findForward("show");
64     }
65 }
```

### 7.1.3 修改 struts 配置

在 struts-config.xml 配置里增加 action 与 actionForm 的关联

----- 内容 -----

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE struts-config PUBLIC "-//Apache Software Foundation//DTD Struts
Configuration 1.3//EN" "http://struts.apache.org/dtds/struts-
config_1_3.dtd">

<struts-config>
    <form-beans >
        <form-bean name="accountForm" type="com.sneb.struts.form.AccountForm"
    />
    </form-beans>
```

```

<global-exceptions />
<global-forwards />

<action-mappings>
    <action name="accountForm" path="/account"
type="org.springframework.web.struts.DelegatingActionProxy"
        cancellable="true">
        <forward name="show" path="/index.jsp" />
    </action>
</action-mappings>

    <message-resources parameter="com.sneb.struts.ApplicationResources" />
</struts-config>

```

截图说明：



```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE struts-config PUBLIC "-//Apache Software Foundation//DTD Struts Configuration 1.3//EN" "http://struts.ap
<struts-config>
    <form-beans>
        <form-bean name="accountForm" type="com.sneb.struts.form.AccountForm" />
    </form-beans>

    <global-exceptions />
    <global-forwards />

    <action-mappings>
        <action name="accountForm" path="/account" type="org.springframework.web.struts.DelegatingActionProxy"
            cancellable="true">
            <forward name="show" path="/index.jsp" />
        </action>
    </action-mappings>

    <message-resources parameter="com.sneb.struts.ApplicationResources" />
</struts-config>

```

增加红框中的代码。

## 7.1.4 修改视图文件

视图文件修改，从 actionForm 对象里读取数据。

----- 内容 -----

```

<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jstl/core_rt"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>
<%
String path = request.getContextPath();
String basePath =
request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort
()+path+"/";
%>

```

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <base href="<%=basePath%>">

    <title>My JSP 'index.jsp' starting page</title>
    <meta http-equiv="pragma" content="no-cache">
    <meta http-equiv="cache-control" content="no-cache">
    <meta http-equiv="expires" content="0">
    <meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
    <meta http-equiv="description" content="This is my page">
    <!--
    <link rel="stylesheet" type="text/css" href="styles.css">
    -->
  </head>

  <body>

    <c:forEach var="account" items="${data}">
      ${account.userId} ---- ${account.userEmail} ---- ${account.userMobile}
    <br />
    </c:forEach>

  </body>
</html>
```

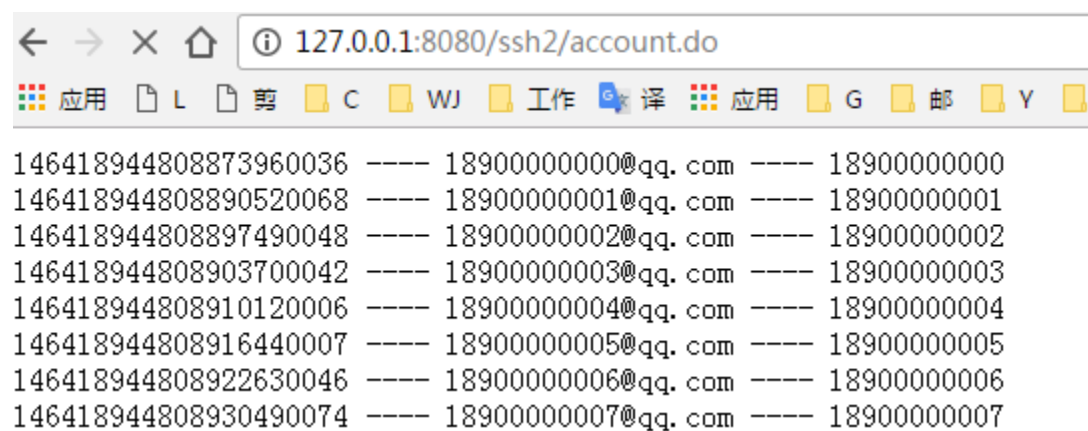
截图如下：

```

1 <%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>
2
3 <%@ taglib prefix="c" uri="http://java.sun.com/jstl/core_rt"%>
4 <%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>
5
6 <%
7 String path = request.getContextPath();
8 String basePath = request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+path+"/";
9 %>
10
11 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
12 <html>
13 <head>
14 <base href="<%=basePath%>">
15
16 <title>My JSP 'index.jsp' starting page</title>
17 <meta http-equiv="pragma" content="no-cache">
18 <meta http-equiv="cache-control" content="no-cache">
19 <meta http-equiv="expires" content="0">
20 <meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
21 <meta http-equiv="description" content="This is my page">
22 <!--
23 <link rel="stylesheet" type="text/css" href="styles.css">
24 -->
25 </head>
26
27 <body>
28
29 <c:forEach var="account" items="${data}">
30     ${account.userId} ---- ${account.userEmail} ---- ${account.userMobile} <br />
31 </c:forEach>
32
33 </body>
34 </html>
35

```

效果检验:



```

146418944808873960036 ---- 18900000000@qq.com ---- 18900000000
146418944808890520068 ---- 18900000001@qq.com ---- 18900000001
146418944808897490048 ---- 18900000002@qq.com ---- 18900000002
146418944808903700042 ---- 18900000003@qq.com ---- 18900000003
146418944808910120006 ---- 18900000004@qq.com ---- 18900000004
146418944808916440007 ---- 18900000005@qq.com ---- 18900000005
146418944808922630046 ---- 18900000006@qq.com ---- 18900000006
146418944808930490074 ---- 18900000007@qq.com ---- 18900000007

```

## 7.2 增删改查完整例子

### 7.2.1 完善列表页面

将数据表数据的列表页面视图文件另存为 list.jsp, 并存放到了/Account/目录下, 同时完善视图, 增加修改、删除和添加的链接等。



## 7.2.2 项目欢迎页

修改 web.xml 将项目入口文件改成 index.jsp 文件，在配置文件最后（/web-app）前插入以下代码：

```
----- 内容 -----
<welcome-file-list>
    <welcome-file>index.jsp</welcome-file>
</welcome-file-list>
----- 内容结束 -----
```

## 7.2.3 数据表 bean 主键类型修改

Account.hbm.xml 修改



```
struts-config.xml  AccountAction.java  list.jsp  add.jsp  Account.hbm.xml x
1 <?xml version="1.0" encoding="utf-8"?>
2 <!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
3 "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
4 <!--
5 Mapping file autogenerated by MyEclipse Persistence Tools
6 -->
7 <hibernate-mapping>
8   <class name="com.sneb.bean.Account" table="account" catalog="webapp">
9     <id name="id" type="java.lang.Integer">
10       <column name="id" />
11       <generator class="assigned"></generator>
12     </id>
13     <property name="userId" type="java.lang.String">
14       <column name="user_id" length="22" not-null="true" unique="true" />
15     </property>
16     <property name="userEmail" type="java.lang.String">
17       <column name="user_email" length="120" />
18     </property>
19     <property name="userMobile" type="java.lang.String">
20       <column name="user_mobile" length="13" />
21     </property>
22     <property name="userPwd" type="java.lang.String">
23       <column name="user_pwd" length="60" />
24     </property>
25     <property name="userStatus" type="java.lang.Short">
26       <column name="user_status" not-null="true">
27         <comment>是否有效</comment>
28       </column>
29     </property>
30     <property name="demo" type="java.lang.String">
31       <column name="demo" length="200" />
32     </property>
33   </class>
34 </hibernate-mapping>
35
```

修改成：

```
struts-config.xml AccountAction.java list.jsp add.jsp Account.hbm.xml x
Account.hbm.xml
1 <?xml version="1.0" encoding="utf-8"?>
2 <!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
3 "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
4 <!--
5 Mapping file autogenerated by MyEclipse Persistence Tools
6 -->
7 <hibernate-mapping>
8   <class name="com.sneb.bean.Account" table="account" catalog="webapp">
9     <id name="id" type="java.lang.Integer">
10       <column name="id" />
11       <generator class="native" />
12     </id>
13     <property name="userId" type="java.lang.String">
14       <column name="user_id" length="22" not-null="true" unique="true" />
15     </property>
16     <property name="userEmail" type="java.lang.String">
17       <column name="user_email" length="120" />
18     </property>
19     <property name="userMobile" type="java.lang.String">
20       <column name="user_mobile" length="13" />
21     </property>
22     <property name="userPwd" type="java.lang.String">
23       <column name="user_pwd" length="60" />
24     </property>
25     <property name="userStatus" type="java.lang.Short">
26       <column name="user_status" not-null="true">
27         <comment>是否有效</comment>
28       </column>
29     </property>
30     <property name="demo" type="java.lang.String">
31       <column name="demo" length="200" />
32     </property>
33   </class>
34 </hibernate-mapping>
```

否则报这个错:

```
127.0.0.1:8080/ssh2/account.do
应用 L 剪 C WJ 工作 译 应用 G 邮 Y 社 闻 牛博 AI 具 册 卓 购

HTTP Status 500 -
org.springframework.orm.hibernate3.HibernateSystemException: ids for this
class must be manually assigned before calling save(): com.sneb.bean.Account;
nested exception is org.hibernate.id.IdentifierGenerationException: ids for this
class must be manually assigned before calling save(): com.sneb.bean.Account
```

## 7.3 最终代码大全

### 7.3.1 AccountAction.java

```
----- 内容 -----
/*
 * Generated by MyEclipse Struts
 * Template path: templates/java/JavaClass.vtl
 */
```

```

package com.sneb.struts.action;

import java.io.UnsupportedEncodingException;
import java.util.List;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.apache.struts.action.Action;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionForward;
import org.apache.struts.action.ActionMapping;

import com.sneb.bean.Account;
import com.sneb.business.AccountBusiness;
import com.sneb.struts.form.AccountForm;

/**
 * MyEclipse Struts Creation date: 11-27-2016
 *
 * XDoclet definition:
 *
 * @struts.action validate="true"
 * @struts.action-forward name="show" path="/index.jsp"
 */
public class AccountAction extends Action {

    private AccountBusiness accountBusiness;

    /**
     * Generated Methods
     */

    public AccountBusiness getAccountBusiness() {
        return accountBusiness;
    }

    public void setAccountBusiness(AccountBusiness accountBusiness) {
        this.accountBusiness = accountBusiness;
    }

    /**
     * Method execute
     */

```

```

* @param mapping
* @param form
* @param request
* @param response
* @return ActionForward
* @throws UnsupportedOperationException
*/
public ActionForward execute(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response) {
    // TODO Auto-generated method stub

    AccountForm aform = (AccountForm) form;
    if ("list".endsWith(aform.getOperateType())) {
        return list(mapping, form, request, response);
    } else if ("add".endsWith(aform.getOperateType())) {
        return add(mapping, form, request, response);
    }
    if ("addPost".endsWith(aform.getOperateType())) {
        return addPost(mapping, form, request, response);
    }
    if ("update".endsWith(aform.getOperateType())) {
        return update(mapping, form, request, response);
    }
    if ("updatePost".endsWith(aform.getOperateType())) {
        return updatePost(mapping, form, request, response);
    }
    if ("detail".endsWith(aform.getOperateType())) {
        return detail(mapping, form, request, response);
    }
    if ("delete".endsWith(aform.getOperateType())) {
        return delete(mapping, form, request, response);
    } else {
        return null;
    }
}

/**
 * 列表页
 *
 * @param mapping
 * @param form
 * @param request
 * @param response

```

```

    * @return ActionForward
    */
    public ActionForward list(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response) {

        List<Account> list = accountBusiness.findAll();
        AccountForm aform = (AccountForm) form;
        aform.setListAccount(list);
        request.setAttribute("data", aform);

        return mapping.findForward("show");
    }

    /**
     * 显示添加表单
     *
     * @param mapping
     * @param form
     * @param request
     * @param response
     * @return ActionForward
     */
    public ActionForward add(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response) {
        return mapping.findForward("add");
    }

    /**
     * 添加处理
     *
     * @param mapping
     * @param form
     * @param request
     * @param response
     * @return ActionForward
     */
    public ActionForward addPost(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response) {

        AccountForm aform = (AccountForm) form;
        accountBusiness.add(aform);
        return list(mapping, form, request, response);
    }
}

```

```

/**
 * 显示修改表单
 *
 * @param mapping
 * @param form
 * @param request
 * @param response
 * @return ActionForward
 */
public ActionForward update(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response) {

    AccountForm aform = (AccountForm) form;
    aform = accountBusiness.findByIdOfForm(aform);
    request.setAttribute("data", aform);

    return mapping.findForward("update");
}

/**
 * 修改表单提交
 *
 * @param mapping
 * @param form
 * @param request
 * @param response
 * @return ActionForward
 */
public ActionForward updatePost(ActionMapping mapping, ActionForm form,
    HttpServletRequest request, HttpServletResponse response) {

    AccountForm aform = (AccountForm) form;
    accountBusiness.update(aform);

    return list(mapping, form, request, response);
}

/**
 * 显示详细信息
 *
 * @param mapping
 * @param form
 * @param request
 * @param response

```

```

    * @return ActionForward
    */
    public ActionForward detail(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response) {

        AccountForm aform = (AccountForm) form;
        aform = accountBusiness.findByIdOfForm(aform);
        request.setAttribute("data", aform);

        return mapping.findForward("detail");
    }

    /**
     * 删除
     *
     * @param mapping
     * @param form
     * @param request
     * @param response
     * @return ActionForward
     */
    public ActionForward delete(ActionMapping mapping, ActionForm form,
        HttpServletRequest request, HttpServletResponse response) {

        AccountForm aform = (AccountForm) form;
        String userId = aform.getUserId();
        Account a = accountBusiness.findById(userId);
        accountBusiness.delete(a);
        return list(mapping, form, request, response);
    }
}

```

### 7.3.2 AccountForm.java

```

/*
 * Generated by MyEclipse Struts
 * Template path: templates/java/JavaClass.vtl
 */
package com.sneb.struts.form;

import java.util.List;

```

```
import javax.servlet.http.HttpServletRequest;

import org.apache.struts.action.ActionErrors;
import org.apache.struts.action.ActionForm;
import org.apache.struts.action.ActionMapping;

import com.sneb.bean.Account;

/**
 * MyEclipse Struts Creation date: 11-28-2016
 *
 * XDoclet definition:
 *
 * @struts.form name="accountForm"
 */
public class AccountForm extends ActionForm {
    /**
     * Generated fields
     */
    /** demo property */
    private String demo;

    /** operateType property */
    private String operateType;

    /** userPwd property */
    private String userPwd;

    /** userId property */
    private String userId;

    /** userMobile property */
    private String userMobile;

    /** userEmail property */
    private String userEmail;

    private List<Account> listAccount;

    private Integer id;

    public Integer getId() {
        return id;
    }
}
```



```

}

public void setId(Integer id) {
    this.id = id;
}

public List<Account> getListAccount() {
    return listAccount;
}

public void setListAccount(List<Account> listAccount) {
    this.listAccount = listAccount;
}

/**
 * Method validate
 *
 * @param mapping
 * @param request
 * @return ActionErrors
 */
public ActionErrors validate(ActionMapping mapping,
    HttpServletRequest request) {
    // TODO Auto-generated method stub
    return null;
}

/**
 * Method reset
 *
 * @param mapping
 * @param request
 */
public void reset(ActionMapping mapping, HttpServletRequest request) {
    // TODO Auto-generated method stub
}

/**
 * Returns the demo.
 *
 * @return String
 */
public String getDemo() {
    return demo;
}

```

```

}

/**
 * Set the demo.
 *
 * @param demo
 *         The demo to set
 */
public void setDemo(String demo) {
    this.demo = demo;
}

/**
 * Returns the operateType.
 *
 * @return String
 */
public String getOperateType() {
    return operateType;
}

/**
 * Set the operateType.
 *
 * @param operateType
 *         The operateType to set
 */
public void setOperateType(String operateType) {
    this.operateType = operateType;
}

/**
 * Returns the userPwd.
 *
 * @return String
 */
public String getUserPwd() {
    return userPwd;
}

/**
 * Set the userPwd.
 *
 * @param userPwd

```

```

    *           The userPwd to set
    */
    public void setUserPwd(String userPwd) {
        this.userPwd = userPwd;
    }

    /**
     * Returns the userId.
     *
     * @return String
     */
    public String getUserId() {
        return userId;
    }

    /**
     * Set the userId.
     *
     * @param userId
     *           The userId to set
     */
    public void setUserId(String userId) {
        this.userId = userId;
    }

    /**
     * Returns the userMobile.
     *
     * @return String
     */
    public String getUserMobile() {
        return userMobile;
    }

    /**
     * Set the userMobile.
     *
     * @param userMobile
     *           The userMobile to set
     */
    public void setUserMobile(String userMobile) {
        this.userMobile = userMobile;
    }

```

```

/**
 * Returns the userEmail.
 *
 * @return String
 */
public String getUserEmail() {
    return userEmail;
}

/**
 * Set the userEmail.
 *
 * @param userEmail
 *         The userEmail to set
 */
public void setUserEmail(String userEmail) {
    this.userEmail = userEmail;
}
}

```

### 7.3.3 AccountBusiness.java

```

package com.sneb.business;

import java.util.List;

import com.sneb.bean.Account;
import com.sneb.struts.form.AccountForm;

public interface AccountBusiness {
    public List<Account> findAll();

    public void add(AccountForm aform);

    public void update(AccountForm aform);

    public void delete(Account a);

    AccountForm findByIdOfForm(AccountForm aform);

    Account findById(String userId);
}

```

### 7.3.4 AccountBusinessImpl.java

```
package com.sneb.business.impl;

import java.util.List;

import com.sneb.bean.Account;
import com.sneb.business.AccountBusiness;
import com.sneb.dao.AccountDao;
import com.sneb.struts.form.AccountForm;

public class AccountBusinessImpl implements AccountBusiness {

    private AccountDao accountDao;

    public AccountDao getAccountDao() {
        return accountDao;
    }

    public void setAccountDao(AccountDao accountDao) {
        this.accountDao = accountDao;
    }

    @Override
    public List<Account> findAll() {
        List<Account> list = accountDao.findAll();
        return list;
    }

    @Override
    public void add(AccountForm aform) {
        // TODO Auto-generated method stub
        Account a = new Account();
        a.setUserId(aform.getUserId());
        a.setUserMobile(aform.getUserMobile());
        a.setUserEmail(aform.getUserEmail());
        a.setUserPwd(aform.getUserPwd());
        a.setDemo(aform.getDemo());
        a.setUserStatus((short) 0);
        accountDao.save(a);
    }

    @Override
```

```

public void update(AccountForm aform) {
    // TODO Auto-generated method stub
    Account a = new Account();
    a.setId(aform.getId());
    a.setUserId(aform.getUserId());
    a.setUserEmail(aform.getUserEmail());
    a.setUserMobile(aform.getUserMobile());
    a.setUserPwd(aform.getUserPwd());
    a.setDemo(aform.getDemo());
    accountDao.update(a);
}

```

```

@Override
public void delete(Account a) {
    // TODO Auto-generated method stub
    accountDao.delete(a);
}

```

```

@Override
public AccountForm findByIdOfForm(AccountForm aform) {
    // TODO Auto-generated method stub

    String userId = aform.getUserId();
    List<Account> list = accountDao.findById(userId);
    if ((null != list) && (list.size() > 0)) {
        Account account = list.get(0);
        aform.setId(account.getId());
        aform.setUserMobile(account.getUserMobile());
        aform.setUserEmail(account.getUserEmail());
        aform.setUserPwd(account.getUserPwd());
        aform.setDemo(account.getDemo());
    }

    return aform;
}

```

```

@Override
public Account findById(String userId) {
    // TODO Auto-generated method stub

    List<Account> list = accountDao.findById(userId);
    Account u = null;
    if ((null != list) && (list.size() > 0)) {

```

```
        u = list.get(0);
    }

    return u;
}
}
```

### 7.3.5 AccountDao.java

```
package com.sneb.dao;

import java.util.List;

import com.sneb.bean.Account;

public interface AccountDao {
    public List<Account> findAll();

    public void save(Account a);

    public void update(Account a);

    public void delete(Account a);

    public List<Account> findByld(String userId);
}
```

### 7.3.6 AccountDaoImpl.java

```
package com.sneb.dao.impl;

import java.util.List;

import org.hibernate.Query;
import org.springframework.orm.hibernate3.support.HibernateDaoSupport;

import com.sneb.bean.Account;
import com.sneb.dao.AccountDao;

public class AccountDaoImpl extends HibernateDaoSupport implements AccountDao {

    private AccountDao accountDao;
```

```

public AccountDao getAccountDao() {
    return accountDao;
}

public void setAccountDao(AccountDao accountDao) {
    this.accountDao = accountDao;
}

@Override
public List<Account> findAll() {

    String hql = "from Account a order by a.userId DESC";

    Query query = this.getSession().createQuery(hql);
    query.setFirstResult(0); // limit 0,20 中的 0
    query.setMaxResults(20); // limit 0,20 中的 20

    List<Account> list = query.list();

    return list;
}

@Override
public void save(Account a) {
    // TODO Auto-generated method stub
    this.getHibernateTemplate().save(a);
}

@Override
public void update(Account a) {
    // TODO Auto-generated method stub
    this.getHibernateTemplate().update(a);
}

@Override
public void delete(Account a) {
    // TODO Auto-generated method stub
    this.getHibernateTemplate().delete(a);
}

@Override
public List<Account> findById(String userId) {

```



```

        // TODO Auto-generated method stub

        String hql = "from Account a where a.userId='" + userId + "'";
        List<Account> list = this.getHibernateTemplate().find(hql);
        return list;
    }

}

```

### 7.3.7 struts-config.xml

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE struts-config PUBLIC "-//Apache Software Foundation//DTD Struts
Configuration 1.3//EN" "http://struts.apache.org/dtds/struts-
config_1_3.dtd">

<struts-config>
    <form-beans >
        <form-bean name="accountForm" type="com.sneb.struts.form.AccountForm"
        />
    </form-beans>

    <global-exceptions />
    <global-forwards />

    <action-mappings>
        <action name="accountForm" path="/account"
type="org.springframework.web.struts.DelegatingActionProxy"
        cancellable="true">
            <forward name="show" path="/Account/list.jsp" />
            <forward name="add" path="/Account/add.jsp" />
            <forward name="update" path="/Account/edit.jsp" />
            <forward name="detail" path="/Account/detail.jsp" />
        </action>
    </action-mappings>

    <message-resources parameter="com.sneb.struts.ApplicationResources" />
</struts-config>

```

### 7.3.8 web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://java.sun.com/xml/ns/javaee"
  xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
  version="3.0">
  <display-name>ssh2</display-name>
  <servlet>
    <servlet-name>action</servlet-name>
    <servlet-class>org.apache.struts.action.ActionServlet</servlet-
class>
    <init-param>
      <param-name>config</param-name>
      <param-value>/WEB-INF/struts-config.xml</param-value>
    </init-param>
    <init-param>
      <param-name>debug</param-name>
      <param-value>3</param-value>
    </init-param>
    <init-param>
      <param-name>detail</param-name>
      <param-value>3</param-value>
    </init-param>
    <load-on-startup>0</load-on-startup>
  </servlet>
  <servlet-mapping>
    <servlet-name>action</servlet-name>
    <url-pattern>*.do</url-pattern>
  </servlet-mapping>
  <listener>
    <listener-
class>org.springframework.web.context.ContextLoaderListener</listener-
class>
  </listener>
  <context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>
classpath:applicationContext.xml,
classpath:applicationContext-Account.xml
</param-value>
  </context-param>
```

```

<welcome-file-list>
    <welcome-file>index.jsp</welcome-file>
</welcome-file-list>

<filter>
    <filter-name>encodingFilter</filter-name>
    <filter-
class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
    <init-param>
        <param-name>encoding</param-name>
        <param-value>UTF-8</param-value>
    </init-param>
</filter>
<filter-mapping>
    <filter-name>encodingFilter</filter-name>
    <url-pattern>/*</url-pattern>
</filter-mapping>

</web-app>

```

### 7.3.9 index.jsp

```

<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>

<%@ taglib prefix="c" uri="http://java.sun.com/jstl/core_rt"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>

<%
    String path = request.getContextPath();
    String basePath = request.getScheme() + "://"
        + request.getServerName() + ":" + request.getServerPort()
        + path + "/";
%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<base href="<%=basePath%>">

<title>SSH-MYSQL-DEMO</title>
<meta http-equiv="pragma" content="no-cache">
<meta http-equiv="cache-control" content="no-cache">

```

```

<meta http-equiv="expires" content="0">
<meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
<meta http-equiv="description" content="This is my page">
<!--
    <link rel="stylesheet" type="text/css" href="styles.css">
-->
</head>

<body>

    <br />
    <br />

    <form action="account.do" method="post">
        <input type="hidden" name="operateType" value="List" /> <input
            type="submit" value="进入列表页" />
    </form>

</body>
</html>

```

### 7.3.10 Account/list.jsp

```

<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>

<%@ taglib prefix="c" uri="http://java.sun.com/jstl/core_rt"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>

<%
    String path = request.getContextPath();
    String basePath = request.getScheme() + "://"
        + request.getServerName() + ":" + request.getServerPort()
        + path + "/";
%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<base href="<%=basePath%>">

<title>My JSP 'index.jsp' starting page</title>
<meta http-equiv="pragma" content="no-cache">
<meta http-equiv="cache-control" content="no-cache">

```

```

<meta http-equiv="expires" content="0">
<meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
<meta http-equiv="description" content="This is my page">
<!--
    <link rel="stylesheet" type="text/css" href="styles.css">
-->

<script type="text/javascript">
    function detail(userId) {
        document.getElementById("operateType").value = "detail";
        document.getElementById("userId").value = userId;
        document.forms[0].submit();
    }
    function add() {
        document.getElementById("operateType").value = "add";
        document.forms[0].submit();
    }
    function update(userId) {
        document.getElementById("operateType").value = "update";
        document.getElementById("userId").value = userId;
        document.forms[0].submit();
    }
    function del(userId) {
        document.getElementById("operateType").value = "delete";
        document.getElementById("userId").value = userId;
        document.forms[0].submit();
    }
</script>

</head>

<body>

    <form action="account.do" method="post">
        <input type="hidden" id="operateType" name="operateType" value="">
        <input type="hidden" id="userId" name="userId" value=""> <input
            type="button" value="添加" onclick="add(${account.userId});">

        <hr />
        ---- 用户ID ---- 邮箱 ---- 手机 ----
        <hr />

        <c:forEach var="account" items="${data.listAccount}">
            ---- ${account.userId} ---- ${account.userEmail} ----

```

```

${account.userMobile} ----
    <input type="button" value="详情"
           onclick="detail('${account.userId}');">&nbsp;
    <input type="button" value="修改"
           onclick="update('${account.userId}');">&nbsp;
    <input type="button" value="删除"
    onclick="del('${account.userId}');">&nbsp;
    <br />
    </c:forEach>
</form>

</body>
</html>

```

### 7.3.11 Account/add.jsp

```

<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>

<%@ taglib prefix="c" uri="http://java.sun.com/jstl/core_rt"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>

<%
    String path = request.getContextPath();
    String basePath = request.getScheme() + "://"
        + request.getServerName() + ":" + request.getServerPort()
        + path + "/";
%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<base href="<%=basePath%>">

<title>add</title>
<meta http-equiv="pragma" content="no-cache">
<meta http-equiv="cache-control" content="no-cache">
<meta http-equiv="expires" content="0">
<meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
<meta http-equiv="description" content="This is my page">
<!--
    <link rel="stylesheet" type="text/css" href="styles.css">

```

```

-->

<script type="text/javascript">
    function saveForm() {
        document.getElementById("operateType").value = "addPost";
        document.forms[0].submit();
    }
    function toList() {
        document.getElementById("operateType").value = "list";
        document.forms[0].submit();
    }
</script>

</head>

<body>

    <form action="account.do" method="post">
        <input type="hidden" id="operateType" name="operateType" value="">

        用户 ID :<input type="text" name="userId" value=""><br />
        用户手机:<input type="text" name="userMobile" value=""><br />
        用户邮箱:<input type="text" name="userEmail" value=""><br />
        用户密码:<input type="text" name="userPwd" value=""><br />
        其他备注:<input type="text" name="demo" value=""><br />

        <input
            type="button" name="btnSaveForm" value="保存"
onOnClick="saveForm()" />
        <input type="reset" name="btnBack" value="重置" /> <input
type="button"
value="返回" onclick="toList();" />

    </form>

</body>
</html>

```

### 7.3.12 Account/edit.jsp

```
<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>

<%@ taglib prefix="c" uri="http://java.sun.com/jstl/core_rt"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>

<%
    String path = request.getContextPath();
    String basePath = request.getScheme() + "://"
        + request.getServerName() + ":" + request.getServerPort()
        + path + "/";
%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<base href="<%=basePath%>">

<title>edit</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<meta http-equiv="pragma" content="no-cache">
<meta http-equiv="cache-control" content="no-cache">
<meta http-equiv="expires" content="0">
<!--
    <link rel="stylesheet" type="text/css" href="styles.css">
-->

<script type="text/javascript">
    function saveForm() {
        document.getElementById("operateType").value = "updatePost";
        document.forms[0].submit();
    }
    function toList() {
        document.getElementById("operateType").value = "list";
        document.forms[0].submit();
    }
</script>

</head>

<body>
```



```

<form action="account.do" method="post">
    <input type="hidden" id="operateType" name="operateType" value="">
    <input type="hidden" id="id" name="id" value="${data.id}"> 用户
    ID :<input type="text" size="40" name="userId" readonly="readonly"
        value="${data.userId}"><br /> 用户手机:<input type="text"
        size="40" name="userMobile" value="${data.userMobile}"><br />
    用户邮箱:<input type="text" size="40" name="userEmail"
        value="${data.userEmail}"><br /> 用户密码:<input type="text"
        size="40" name="userPwd" value="${data.userPwd}"><br />
    其他备注:<input type="text" size="40" name="demo"
value="${data.demo}"><br />

    <input type="button" name="btnSaveForm" value="保存"
        onClick="saveForm()" /> <input type="reset" name="btnBack"
        value="重置" /> <input type="button" value="返回"
onClick="toList();" />

</form>

</body>
</html>

```

### 7.3.13 Account/detail.jsp

```

<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>

<%@ taglib prefix="c" uri="http://java.sun.com/jstl/core_rt"%>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>

<%
    String path = request.getContextPath();
    String basePath = request.getScheme() + "://"
        + request.getServerName() + ":" + request.getServerPort()
        + path + "/";
%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<base href="<%=basePath%>">

```

```

<title>add</title>
<meta http-equiv="pragma" content="no-cache">
<meta http-equiv="cache-control" content="no-cache">
<meta http-equiv="expires" content="0">
<meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
<meta http-equiv="description" content="This is my page">
<!--
    <link rel="stylesheet" type="text/css" href="styles.css">
-->

</head>

<body>

    <form action="account.do" method="post">
        <input type="hidden" id="operateType" name="operateType"
value="List">

        用户 ID :${data.userId}<br />
        用户手机:${data.userMobile}<br />
        用户邮箱:${data.userEmail}<br />
        用户密码:${data.userPwd}<br />
        其他备注:${data.demo}<br />

        <input type="submit" value="返回" />

    </form>

</body>
</html>

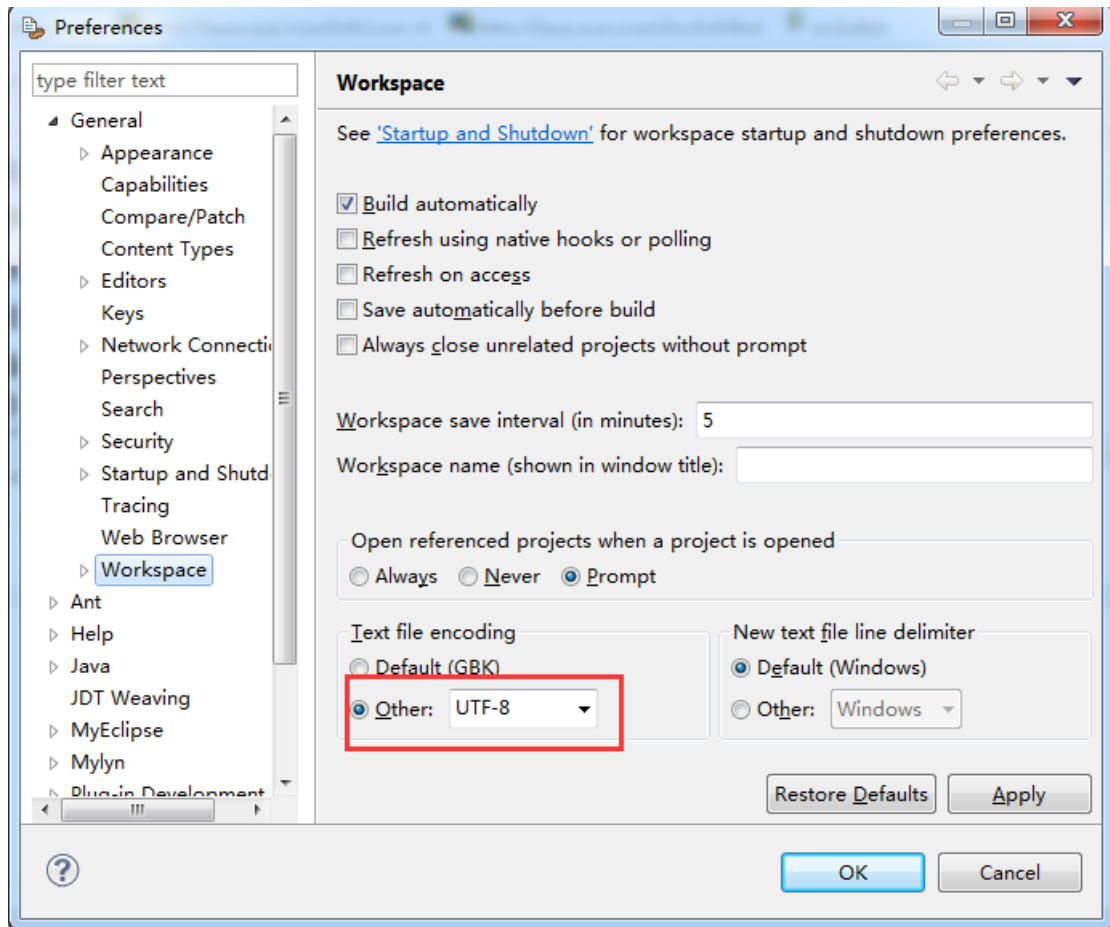
```

## 8.连接数据库乱码问题解决

推荐所有地方的编码均设置成 utf-8 编码，避免乱码问题。

### 8.1 设置 myeclipse 字符集

菜单 Window--Preferences--General--Workspace, Text file encoding 选项中默认的 Default(GBK) 选项更改为 Other，并将值设为 utf-8



## 8.2 修改 myeclipse 的视图和配置文件编码

菜单 Window--Preferences--MyEclipse--Files and Editors, 将这个选项下面的: ASP and PHP、CSS、DTD、HTML、JSP、XML 中的字符编码全部更改为 utf-8

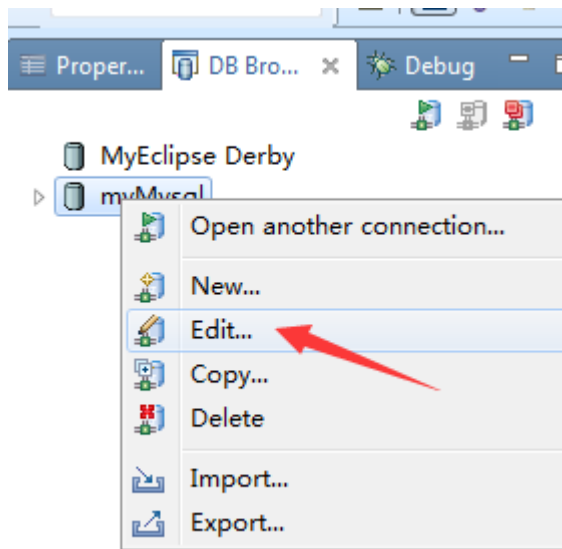
## 8.3 设置 mysql 的字符集编码

略

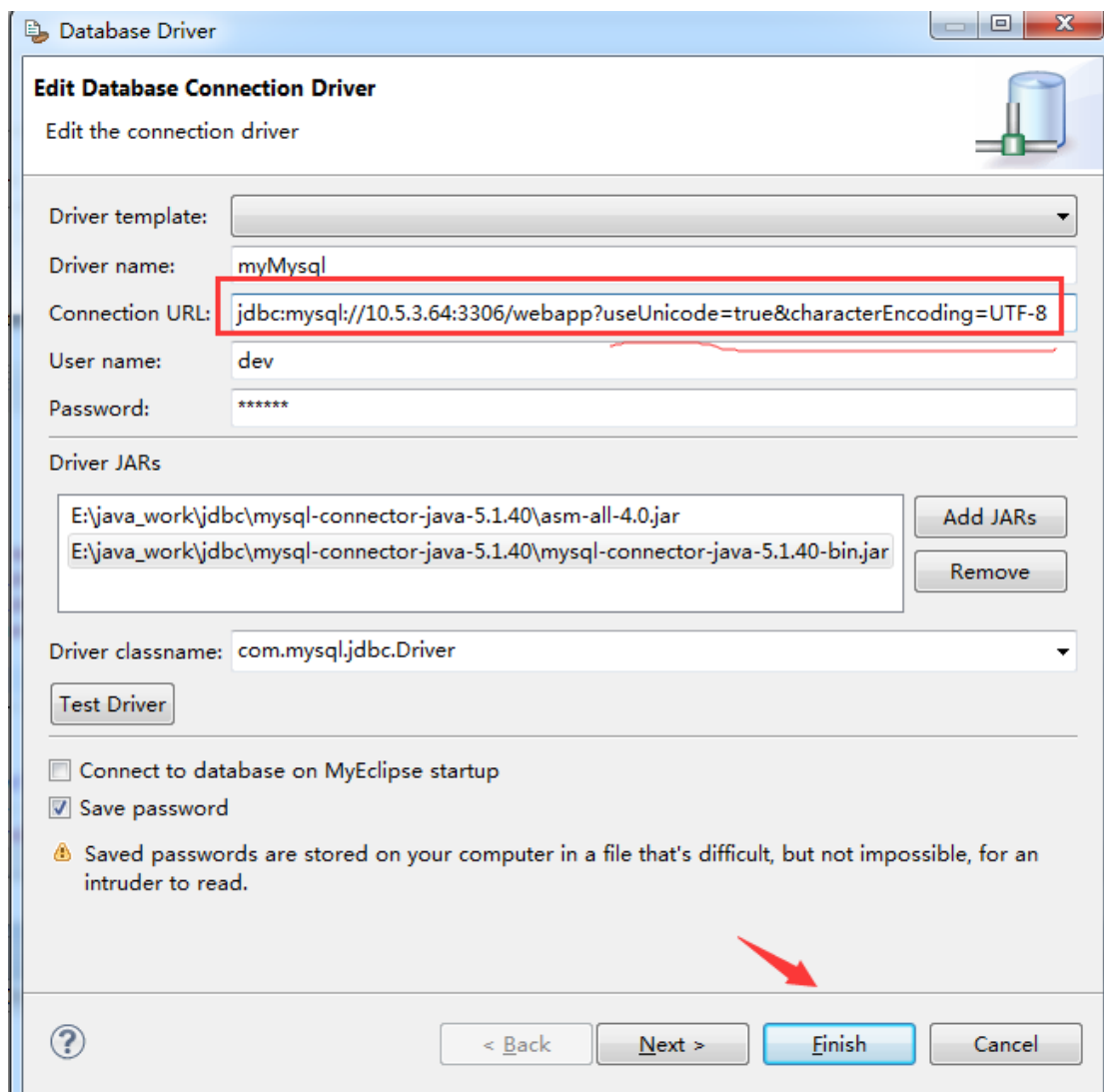
## 8.4 修改数据库连接路径

### 8.4.1 编辑 myeclipse 中的 Db Browser

myMysql 上有右键 → Edit  
→



→



→ Finish

在连接 url 后面增加:

?useUnicode=true&characterEncoding=UTF-8

## 8.4.2 修改项目中的 hibernate 配置

applicationContext.xml 中删除 dataSource 的 bean 配置，然后右键 → Spring → New DataSource → 略。

会自动加上字符集，如下图：



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:p="http://www.springframework.org/schema/p"
4       xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/
5
6 <bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource">
7   <property name="driverClassName" value="com.mysql.jdbc.Driver">
8   </property>
9   <property name="url"
10     value="jdbc:mysql://10.5.3.64:3306/webapp?useUnicode=true&characterEncoding=UTF-8">
11   </property>
12   <property name="username" value="dev"></property>
13   <property name="password" value="dev963"></property>
14 </bean>
15
16 <bean id="sessionFactory"
17       class="org.springframework.orm.hibernate3.annotation.AnnotationSessionFactoryBean">
18   <property name="dataSource">
19     <ref bean="dataSource" />
20   </property>
21   <property name="hibernateProperties">
22     <props>
23       <prop key="hibernate.dialect">
24         org.hibernate.dialect.MySQLDialect
25       </prop>
26     </props>
27   </property>
28   <property name="mappingResources">
29     <list>
30       <value>com/sneb/bean/Account.hbm.xml</value>
31     </list>
32   </property>
33 </bean>
34
35
36 </beans>
```

----- 内容 -----

```
<bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource">
  <property name="driverClassName" value="com.mysql.jdbc.Driver">
  </property>
  <property name="url"

  value="jdbc:mysql://10.5.3.64:3306/webapp?useUnicode=true&character
Encoding=UTF-8">
  </property>
  <property name="username" value="dev"></property>
  <property name="password" value="dev963"></property>
</bean>
```

## 8.5 添加 struts 字符集过滤器

修改 web.xml 文件，最后（</web-app>前）添加：

----- 内容 -----

```
<filter>
  <filter-name>encodingFilter</filter-name>
  <filter-
class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
  <init-param>
    <param-name>encoding</param-name>
    <param-value>UTF-8</param-value>
  </init-param>
</filter>
<filter-mapping>
  <filter-name>encodingFilter</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

===== 教程完 =====