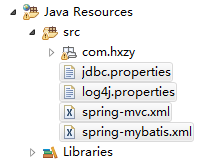
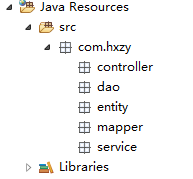
# 配置文件：



# 项目目录：



# 1、创建Page、book、user实体bean：

**private** **int** offset;//查询起始位置：0[无限制]

**private** **int** limit;//偏移量：0[无限制]

**private** **int** count;//总条数

**public** Page(){}

**public** Page(**int** offset, **int** limit) {

**super**();

**this**.offset = offset;

**this**.limit = limit;

}

Get、set……

# 2、创建DAO接口：

**public** **interface** BookDataBaseDao {

**public** List<Book> queryBookListByPage(Page page);

}

不需要实现

# 3、创建mapper配置文件：

<?xml version=*"1.0"* encoding=*"UTF-8"* ?>

<!DOCTYPE mapper

PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"

"http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace=*"com.hxzy.dao.BookDataBaseDao"*>

<!-- 对象参数传递到了语句中，id、username 和 password 属性将会被查找，然后将它们的值传入预处理语句的参数中 -->

<select id=*"queryBookListByPage"* resultType=*"com.hxzy.entity.Book"*

parameterType=*"com.hxzy.entity.Page"*>

select `id`,`name`,`author`,`publish`,left(publish\_date,10) as publishDate,`page`,`price`,`content`

from `book`

<if test=*"offset>=0 and limit!=0"*>

limit #{offset},#{limit}

</if>

</select>

</mapper>

# 4、创建Service（接口）用来申明增删改查得方法：

**public** **interface** BookDataBaseService {

**public** List<Book> queryBookListByPage(Page page);

}

# 5、实现Service接口：

//表示的Service层，如果不写Spring就找不到

@Service

**public** **class** BookDataBaseServiceImpl **implements** BookDataBaseService {

//表示Spring的自动注入

//依赖注入（DI）：依赖【需要某个类对象】

//,注入【Spring会自动创建并且注入到这个类对象文件中】

@Resource

**private** BookDataBaseDao dao;

/\*\*

\* 请求一页数据，根据page对象里面设置的起始位置和偏移量

\* **@return** 查询到的数据列表

\*/

**public** List<Book> queryBookListByPage(Page page) {

//第一次查询，只获取总条数

List<Book> bs = dao.queryBookListByPage(**new** Page());

page.setCount(bs.size());

//第二次查询，查询指定的N条

**return** dao.queryBookListByPage(page);

}

}

# 6、创建Controller：相当于servlet：

/\*\*

\* spring mvc处理方法支持的返回方式：ModelAndView, Model, ModelMap, Map,View, String, void

\* ModelAndView构造方法可以指定返回的页面名称，也可以通过setViewName()方法跳转到指定的页面 ,

\* 使用addObject()设置需要返回的值，addObject()有几个不同参数的方法，可以默认和指定返回对象的名字。

\* 调用addObject()方法将值设置到一个名为ModelMap的类属性，ModelMap是LinkedHashMap的子类

\* **@author** STAR

\*

\*/

@ResponseBody

@Controller

@Scope("prototype")

**public** **class** BookDataBaseController {

@Resource

**private** BookDataBaseService service;

/\*\*

\* 获取一页数据

\* **@param** session

\* **@param** myoffset 起始页

\* **@param** mylimit 请求条数

\* **@param** map

\* **@return**

\*/

@RequestMapping(value = "/booksPage.do")

**public** Map<String,Object> toBookListPage(HttpSession session, Page page, ModelMap map) {

Object obj = session.getAttribute("loginUser");

List<Book> list = **null**;

**if** (obj == **null**) {

map.put("message", "登入已过期，请重新登入！");

//return "book\_db/index";

}

System.***err***.println("Limit:"+page.getLimit()+":"+"Offset:"+page.getOffset());

Map<String,Object> result = **new** HashMap<String,Object>();

list = service.queryBookListByPage(page);

result.put("total",page.getCount());

result.put("rows",list);

System.***out***.println("总共"+list.size()+"条数据");

**return** result;

}

}