黑客派

JAVA 小白启蒙篇：第一个 SSM 框架搭建示例（附源码）

作者：relyn

原文链接：https://hacpai.com/article/1470110280017

来源网站：黑客派

<p># 背景
 前段时间，忙这忙那的，又是专题分析，又是 PMP 培训，一直都没有办法闲下来最近又换了班组，昨天才把家当都搬好，今天终于有空可以写点东西了。上次给班组成员进行了一期 Java 的基础培训，这次准备来个第二发，虽然这两期培训并没有太大的连贯性，但作为 Java 入门，掌这些内容，基本是够了。此外，因为框架这种东西，内容很多，原理很复杂，不是三言两语能讲得明的，因此也只能算是抛砖引玉了。
 #SSM 框架简介
 SSM 框架，是 Spring + Spring MVC + MyBatis 的缩写，这个是继 SSH 之后，目前比较主流的 Java EE 企业级框架，适用于搭建各种大的企业级应用系统。
 #Spring 简介
 Spring 是一个开源框架，Spring 是于 2003 年兴起一个轻量级的 Java 开发框架，由 Rod Johnson 在其著作 Expert One-On-One J2EE Development nd Design 中阐述的部分理念和原型衍生而来。它是为了解决企业应用开发的复杂性而创建的。Sprin 使用基本的 JavaBean 来完成以前只可能由 EJB 完成的事情。然而，Spring 的用途不仅限于服务器的开发。从简单性、可测试性和松耦合的角度而言，任何 Java 应用都可以从 Spring 中受益。 简单说，Spring 是一个轻量级的控制反转（IoC）和面向切面（AOP）的容器框架。
 ## 控反转是什么呢？
 打个比方，你晚上想约个妹子去看电影，假设这个妹子是一个类（温柔善良、小鸟依人），那么你需要实例化她，在 Java 中，通常的做法就是 new 一个类，让她变成一个体的对象。然后，你跟她约了时间、地点，说要请客吃饭，然后用尽你浑身解数，妹子终于答应要和去看电影了。看完电影之后，你还要送妹子回家，确保安全。当然，这一个步骤一般是由 JVM 的垃回收机制来处理，如果你没有主动及时送妹子回家，当妹子数量很多的时候，有可能会导致内存溢出系统宕机。
 而如果你用了 Spring，过程将会是这样的，你告诉 Spring 框架你晚上 7 点要和一温柔、善良、小鸟依人般的妹子去看电影，那么你 7 点准时来到电影院，此时，你会发现，Spring 经帮你把妹子送来了，完全就是你定义的那个类型。你们愉快地看完电影，然后你可以直接回家，Spring 又会帮你把妹子接走。你完全不用关心后续的事情。
 在以往，我们是主动地去控制一个对象实例化和销毁，现在交由 Spring 来控制，因此就叫做控制反转，Inversion of Control，是不是很好解，一点就通？
 ## 面向切面又是什么呢？
 首先，需要说明的一点，AOP 是 Spring 的特性，它就像 OOP 一样是一种编程思想，并不是某一种技术。
 我们再来打个比方在编程的时候，我们为了满足某个业务逻辑流程，通常需要一系列步骤。请看下面 3 个具体业务的流图：

> 图上可以看到，用户不管是查询余额、办理业务还是手机充值，都需要经过鉴权这个步骤，有没有过可以把这个鉴权的代码是提取出来，不放到主流程里去呢，这就是 AOP 的作用了，有了 AOP，你代码时不要把这个验证用户步骤写进去，即完全不考虑用户鉴权，你写完之后，在另一个地方，写好证用户的代码，然后告诉 Spring 你要把这段代码加到哪几个地方，Spring 就会帮你加过去，而不要自己 Copy 过去，这里还是两个地方，如果你有多个控制流呢，这个写代码的方法可以大大减少你的间，不过 AOP 的目的不是这样，这只是一个“副作用”，真正目的是，你写代码的时候，事先只需虑主流程，而不用考虑那些不重要的流程，懂 C 的都知道，良好的风格要求在函数起始处验证参数，果在 C 上可以用 AOP，就可以先不管校验参数的问题，事后使用 AOP 就可以隔山打牛的给所有函一次性加入校验代码，而你只需要写一次校验代码。不知道 C 的没关系，举一个通用的例子，经常在ebug 的时候要打 log 吧，你也可以写好主要代码之后，把打 log 的代码写到另一个单独的地方，然命令 AOP 把你的代码加过去，注意 AOP 不会把代码加到源文件里，但是它会正确的影响最终的机代码。就像在上帝视角的高纬度空间，把你需要的步骤像切面般的插入到特定的时空里面。
 #Spring MVC 简介
 Spring MVC 属于 Spring Framework 的后续产品，已经融合在 Spring Web Flw 里面，它原生支持的 Spring 特性，让开发变得非常简单规范。Spring MVC 分离了控制器、模型象、分派器以及处理程序对象的角色，这种分离让它们更容易进行定制。
 Spring MVC 的架构次培训已经介绍过了，下面这个是它主要的工作原理图：

#MyBatis 简介
 MyBatis 本是 apache 的一个开源项目 iBatis, 2010 年这个项目由 apache software foundation 迁移到了 Google code 并且改名为 MyBatis 。MyBatis 是一个基于 Java 的持久层框架。iBATIS 提供的持久层框架包括 SQL Maps 和 Data Access Objects（DAO）MyBatis 消除了几乎所有的 JDBC 代码和参数的手工设置以结果集的检索。MyBatis 使用简单的 XML 或注解用于配置和原始映射，将接口和 Java 的 POJOs（Plin Old Java Objects，普通的 Java 对象）映射成数据库中的记录。可以这么理解，MyBatis 是一个来帮你管理数据增删改查的框架。
 他的结构如下图所示：

源码解析

讲了那么多，都是云里雾里，还是看代码比较实在。
 我们建了一个 HelloWorld 工程，这个工整合了 SSM 框架，用了 MySQL 数据库（当然如果要改成其他数据库，只是一条配置语句的问题）<r> 首先看下这个 HelloWorld 工程的结构：
 <img src="https://static.hacpai.com/images/ig-loading.svg" alt="微信截图_20160802112157.png" data-src="http://7xw87f.com2.z0.glb.qinicdn.com/7298d6a6b70d473fb3a7d7d4d73443f9.png">
 遵循了 MVC 结构，可以看到有模（Model）、视图（View，下面的 src 文件夹）、控制器（Controller），而 Mapping 是 MyBatis 用来操作数据库的映射文件。
 好的，我们一个个来看：
 ## 模型（Model）
 这是一普通 Java 对象（POJO），<code>User.java</code>，源码如下：</p>

<pre><code class="language-java highlight-chroma">package com.relyn.model;

public class User {

public String userName = "";

public String userMsisdn = "";

public String getUserName() {

return userName;

}

public void setUserName(String userName) {

this.userName = userName;

}

public String getUserMsisdn() {

return userMsisdn;

}

public void setUserMsisdn(String userMsisdn) {

this.userMsisdn = userMsisdn;

}

}

</code></pre>

主要作用就是数据的临时存储，这个 User 对象一般和数据库中的 User 表结构保持一致。
视图（View）
 这就是一个普通的 HTML 页面，<code>index.jsp</code>，源码如下：

<pre><code class="language-html highlight-chroma"><html>
  <head>
    <title>HelloWorld</title>
  </head>
  <body>
    Hello, ${userName}, your phone is ${userMsisdn}
  </body>
</html>
</code></pre>
需要说明的是，这里用到了 EL 表达式，如：<code>${userName}</code> 和 <code>${usersisdn}</code> 表示的就是由 Spring 控制器推送过来的变量。
 ## 控制器（Controller）
 这就是 Spring 控制器，<code>UserController.java</code>，源码如下：
<pre><code class="language-java highlight-chroma">package com.relyn.controller;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.servlet.ModelAndView;

import com.relyn.dao.Userao;

import com.relyn.model.Usr ;

@Controller

@RequestMapping (value ="/user")

public class UserController {

}

</code></pre>

<script async src="https://pagead2.googlesyndication.com/pagead/js/adsbygoogle.js"></script>

<!-- 黑客派PC帖子内嵌-展示 -->

<ins class="adsbygoogle" style="display:block" data-ad-client="ca-pub-5357405790190342" data-ad-slot="8316640078" data-ad-format="auto" data-full-width-responsive="true"></ins>
<script>
    (adsbygoogle = window.adsbygoogle || []).push({});
</script>
可以看到，首先它引入了 SpringFramework 的一些包，然后用注解的方式在代码中来声明这是个控制器 <code>@Controller</code>，用 <code>@RequestMapping</code> 来指定了这个制器的入口，用 <code>@Autowired</code> 来自动实例化这个 UserDao 的对象（这就是刚才说的控制反转），而第二个 <code>@RequestMapping</code> 表示的是这个方法的入口。最后，个控制器通过返回 ModelAndView 对象到视图（回忆一下刚才的那个 Spring MVC 工作流程图）。
 ## 数据访问对象（DAO）
 上述控制器源码中，有一条关键语句，用来从数据库读取数据
<pre><code class="language-java highlight-chroma">User user = userDao.getUserNameByMsisdn(userMsisdn);
</code></pre>
这里用到了 UserDao 这个接口，我们看下源码：
<pre><code class="language-java highlight-chroma">package com.relyn.dao;
import com.relyn.model.Usr;

public interface UserDao {

public User getUserNameByMsisdn(String userMsisdn);

}
</code></pre>

首先需要注意的是，这并不是一个 Class，而是一个 Interface，源码非常简单，主要作用通过方名称就能理解：getUserNameByMsisdn，通过号码获取用户姓名。
 那么，他是怎么做到的呢
 ## 数据库映射（Mapper）
 这时候就需要 MyBatis 的数据库映射配置，我们看下源码：/p>
<pre><code class="language-xml highlight-chroma"><?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.relyn.dao.UserDao">
    <resultMap type="com.relyn.model.User" id="userMap">
        <id property="id" column="id" />
        <result property="userMsisdn" column="user_msisdn" />
        <result property="userName" column="user_name" />
    </resultMap>
    <select id="getUserNameByMsisdn" parameterType="java.lang.String"
        resultMap="userMap">
        select * from user where user_msisdn=#{msisdn}
    </select>
</mapper>
</code></pre>
可以看到，这是一个 XML 文件，在 <code><select></code> 标签中我们看到了熟悉的 QL 语句：
<pre><code class="language-sql highlight-chroma">select * from user where user_msisdn=#{msisdn}
</code></pre>
<script async src="https://pagead2.googlesyndication.com/pagead/js/adsbygoogle.js"></script>
<!-- 黑客派PC帖子内嵌-展示 -->
<ins class="adsbygoogle" style="display:block" data-ad-client="ca-pub-5357405790190342" data-ad-slot="8316640078" data-ad-format="auto" data-full-width-responsive="true"></ins>
<script>

(adsbygoogle = window.adsbygoogle || []).push({});
</script>
这里的 <code>#{msisdn}</code> 就是由控制器传入的变量。这个映射的作用是，把从数据中取出的数据，映射到 User 类中，这就是刚才说的为什么 User 对象要和数据库中 user 表的结构保一致的原因了。
 # 最后
 刚才说的这几个文件源码，是 SSM 框架中最基本的应用，如果需增加不同的表或者业务，这几个文件一个都不能少。当然，如果你打算从头开始搭建框架，你就必须注以下几个配置文件：
 <img src="https://static.hacpai.com/images/img-loading.svg" alt=微信截图_20160802114949.png" data-src="http://7xw87f.com2.z0.glb.qiniucdn.com/ae0fc6ab0494baab123fc51d6604d6b.png">
 <code>config.properties</code> 为基本的配置文件主要用于配置数据库账号、密码等一些通用的定义
 <code>log4j.properties</code> 为 Log f r J 日志的配置文件，这是一个常用的 Java 日志输出工具，也可以换成其他的
 <code>spring-b se.xml</code> 为 Spring 基本配置文件，必不可少
 <code>spring-druid.xml</code> 为数库连接池的配置文件，同样也可以换成别的连接池
 <code>spring-mvc.xml</code> 顾名思，为 Spring MVC 的配置文件，也是必不可少
 <code>spring-mybatis.xml</code> 就是 My atis 和 Spring 整合的配置文件了，也是必不可少的
 最后，再说一下，JavaEE 工程中，所有请的入口来源都是 <code>web.xml</code> 这个文件，通常，我们使用了 Spring MVC 时，都需要定把所有的请求丢给 Spring 控制器来处理。
 #HelloWorld 工程源码下载