学生清华开发手册

1. 项目管理
   1. 代码使用SVN进行管理。由于SAE代码容量太小，现已切换至Google Code。项目页面为：<http://code.google.com/p/sthu/>，SVN checkout的路径是<https://sthu.googlecode.com/svn>。用户名为gmail帐号，密码请去<https://code.google.com/hosting/settings> 页面查看
   2. 项目使用Redmine进行管理。地址为：<http://sthu.m.redmine.org/projects/sthu>
2. 环境搭建
   1. 项目使用Struts2 + Hibernate 4+Spring 3作为基础进行开发
   2. 推荐指导书：轻量级Java EE企业应用实战，下载地址：<http://sthu.m.redmine.org/issues/12>
   3. IDE为NetBeans 7.2.1，数据库使用MySQL。（强烈建议使用Windows Installer版进行安装，并且安装上workbench组件）数据库用户名密码请皆设置为root。安装后请新建一个名为sthu的数据库，编码设为utf8\_general\_ci（可以使用workbench进行操作）
   4. NetBeans自带Tomcat，理论上无需另外安装。如果无法运行，请安装Tomcat 7，并在Netbeans的工具->服务器选项中进行配置。
3. 开发概述
   1. 在Netbeans的Web页文件夹中存放web页面，在源包文件夹中存放Java类。
   2. 请将不同作用的类放入对应的包中，例如所有的Action都应该放在action包下，并且命名为XXXAction。Service，DAO，Entity，Message也是如此。每个包下都有一个BaseXXX的类，请extends它们。
   3. Action层用于输入输出处理（例如检查值的合法性，产生返回值等），Service层进行逻辑操作，DAO层进行纯粹的、原子的数据库操作（不进行逻辑处理）
   4. Entity中每一个类和一个数据库的Table对应，使用Annotation的方法进行映射，具体用法请搜索Hibernate Annotation Tutorial
   5. Message是Action向JSP页面传递数据的容器，一个Action只能有一个Message。如果需要，可以通过Message嵌套的方法来实现传递多个Message。
4. 开发示例

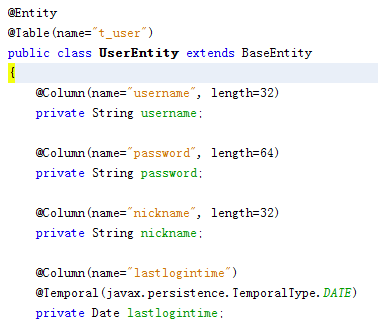
下面以开发新增用户、用户登录两个功能为例介绍如何进行开发。

1. 实现Entity

Entity规定了如何将数据存储在数据库中。在这里，我们需要一个UserEntity来保存用户信息，这也就对应了一个数据库的表。

我们考虑，（假设）我们需要存储用户名、密码、昵称和上次登录时间这几个信息。那么在包entity下创建UserEntity类，并且extends BaseEntity.

在BaseEntity中已经实现了Id字段，所以在这里不再需要定义Id，直接定义所需的字段即可。

  
这里使用了Annotation的方法实现了类成员和数据库表之间的映射。具体的含义请自行搜索学习。

注意1：Table名请以t\_开头，防止和mysql的保留关键词冲突（user就是一个保留词）

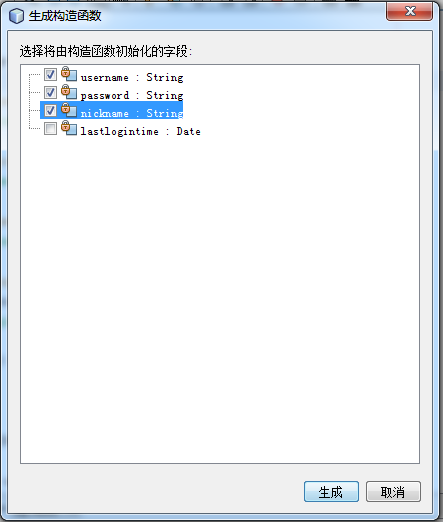
注意2：此处并不需要真正创建t\_user表，Hibernate会自动做这件事情

注意3：当需要联表查询的时候，只要设置好映射关系，Hibernate会自动进行处理（创建mapping表等），详见参考书6.1节（参考书使用xml，我们使用annotation，本质是一样的）

创建字段后，使用重构方法创建每个字段的getter和setter：



使用插入代码功能（Alt+Insert）创建构造函数



**记得同时创建一个无参数的构造函数！**

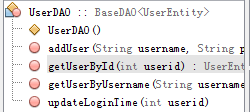
1. 实现DAO

DAO的作用是和操作Entity对象，和数据库进行操作。请注意，一个DAO和一个Entity对应！并且DAO中都应该是原子操作。

在dao包下新建一个UserDAO并且extends BaseDAO，使用super参数指定对应的Entity类

考虑一下实现新增用户和用户登录需要的数据库操作：新增记录、通过用户名查找用户、通过用户ID查找用户、更新用户登录时间。

于是实现了如下几个方法：

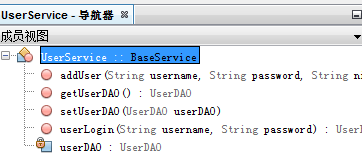


说明：BaseDAO感觉不太好，可能还会有变化

1. 实现Service

Service的作用是和不同的DAO进行交互，来实现一些特定的功能。一个特定的功能大类对应一个Service。

在service包下创建UserService并且extends BaseService



private UserDAO userDAO;

因为需要调用UserDAO进行数据库操作，所以声明UserDAO，并且创建对应的getter和setter。注意，这里并不需要对UserDAO进行初始化，稍后通过配置bean来实现这一点。

为增加用户和用户登录创建对应的两个方法。注意要给这两个方法添加上@Transactional的注解开启事务，否则无法操作数据库。（也就是说，对于那些不需要操作数据库的服务，就不需要@Transactional注解）

1. 创建Message

Message用于从Action向JSP页面传递值，理论上每个Action对应一个Message。由于新增用户的操作我们使用已有的AlertMessage进行传递，所以只需要一个LoginMessage。

在message包下创建LoginMessage类extends BaseMessage，假设我们想要显示上次登录时间和用户名称，那么创建这两个字段，并且设置好getter和setter。另外还可以使用插入代码功能自动创建构造函数。**记得始终需要有一个默认的无参数的构造函数！**

1. 创建Action

理论上一个Action对应一个操作，在这里我们有两个操作，所以需要两个Action。

在Action包下新增AddUserAction和LoginAction，并且都需要extends BaseAction。

首先是LoginAction，登录需要输入用户名和密码，所以有username和password两个域，然后需要调用UserService进行业务操作，调用LoginMessage进行消息传递，一一声明这些域（但是不需要初始化这些域），创建getter和setter

LoginAction需要实现父类中abstract的onExecute()和valid()方法。在valid()方法中进行数据的校验（只需要进行表面上的校验即可），在execute()里调用Service的userlogin函数进行登录的校验，并且把相关的值赋给LoginMessage以便之后的数据显示。

onExecute的返回值用来指定返回的结果，默认有SUCCESS，ALERT和ERROR三种。如果需要更多，请在对应的Action中先定义字符串常量，然后返回这个常量。例如public static final String MANAGE = “manage”。

值得注意的是alertMessage和ALERT，在alertMessage中设定显示的消息，然后Return ALERT，用户就会获得一个提示。其中title和content用来设定消息的标题和内容，redirecturl指定用户点击确定后，跳转的位置。如果要让用户跳转到来源地址（默认），使用常量REFERER\_URL。默认的提示方式是弹窗，如果需要页面提示，设置type为BOX\_TYPE。

1. 创建JSP页面

在web页下创建test文件夹（建议类似功能的页面放在同一个目录下），创建add\_user.jsp, login.jsp和welcome.jsp。在add\_user.jsp和login.jsp中创建两个表单，用于提交数据，在welcome.jsp中，控制登录成功后的显示页面。

如何获取Action里的Message?使用下面的代码即可。

LoginMessage message = Util.getMessage(LoginAction.class);

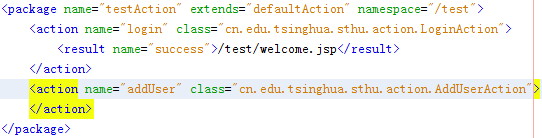
然后使用<%%>来插入java代码，使用<%=>来输出变量，或者在<%%>里使用out对象输出，具体请看参考书2.9节。

注意1：**JSP页面请使用根目录下的example.jsp作为模板！这个页面因为是之前写的所以未使用模板，今后新建的JSP页面请务必复制模板。**

注意2：jsp页面上的java代码只用来控制输出和输出逻辑，切勿用于进行任何业务逻辑判断

1. 配置文件将所有的东西粘合在一起

首先在struts.xml里进行配置。请注意，根目录下的一个文件夹对应一个package。例如有一个test文件夹，就要新建一个package



Namespace属性指定刚才那个新建的test文件夹。然后新建两个Action，分别对应LoginAction和AddUserAction。

浏览器端如何传输数据？在form表单的Action中设置对应的action名.do即可。例如login.do。action名就是指这里配置文件的action name=xxx的xxx，和class那里的名字无关。

然后用<result>配置结果页面，success就转向welcome.jsp页面，注意这里的success就是Action的SUCCESS常量的值，所以如果Action配置了其它的返回值，要在这里配置对应的结果页面。

然后在WEB-INF\MessageContext.xml DAOContext.xml和ServiceContext.xml中配置bean，bean的id要和对应的Service、DAO、Action中成员域的名字相同，这样Spring就能自动进行装配。

**以上配置当新建一个页面的时候，请直接复制已有的配置项然后修改对应的值即可**。