springboot(十五)：springboot+jpa+thymeleaf增删改查示例

 2017/09/23

这篇文章介绍如何使用jpa和thymeleaf做一个增删改查的示例。

先和大家聊聊我为什么喜欢写这种脚手架的项目，在我学习一门新技术的时候，总是想快速的搭建起一个demo来试试它的效果，越简单越容易上手最好。在网上找相关资料的时候总是很麻烦，有的文章写的挺不错的但是没有源代码，有的有源代码但是文章介绍又不是很清楚，所在找资料的时候稍微有点费劲。因此在我学习Spring Boot的时候，会写一些最简单基本的示例项目，一方面方便其它朋友以最快的方式去了解，一方面如果我的项目需要用到相关技术的时候，直接在这个示例版本去改造或者集成就可以。

现在的技术博客有很多的流派，有的喜欢分析源码，有的倾向于底层原理，我最喜欢写这种小而美的示例，方便自己方便他人。

其实以前写过thymeleaf和jpa的相关文章：[springboot(四)：thymeleaf使用详解](http://www.ityouknow.com/springboot/2016/05/01/springboot(%E5%9B%9B)-thymeleaf%E4%BD%BF%E7%94%A8%E8%AF%A6%E8%A7%A3.html)和[springboot(五)：spring data jpa的使用](http://www.ityouknow.com/springboot/2016/08/20/springboot(%E4%BA%94)-spring-data-jpa%E7%9A%84%E4%BD%BF%E7%94%A8.html) 里面的代码示例都给的云收藏的内容[Favorites-web](https://github.com/cloudfavorites/favorites-web)，云收藏的内容比较多，查找起来不是很方便，因此想重新整理一篇快速上手、简单的内容，来介绍jpa和thymeleaf的使用，也就是本文的内容。

这篇文章就不在介绍什么是jpa、thymeleaf，如果还不了解这些基本的概念，可以先移步前两篇相关文章。

**快速上手**

**配置文件**

**pom包配置**

pom包里面添加jpa和thymeleaf的相关包引用

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-thymeleaf</artifactId>

</dependency>

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

</dependency>

**在application.properties中添加配置**

spring.datasource.url=jdbc:mysql://127.0.0.1/test?useUnicode=true&characterEncoding=utf-8&serverTimezone=UTC&useSSL=true

spring.datasource.username=root

spring.datasource.password=root

spring.datasource.driver-class-name=com.mysql.jdbc.Driver

spring.jpa.properties.hibernate.hbm2ddl.auto=update

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL5InnoDBDialect

spring.jpa.show-sql= true

spring.thymeleaf.cache=false

其中propertiesspring.thymeleaf.cache=false是关闭thymeleaf的缓存，不然在开发过程中修改页面不会立刻生效需要重启，生产可配置为true。

在项目resources目录下会有两个文件夹：static目录用于放置网站的静态内容如css、js、图片；templates目录用于放置项目使用的页面模板。

**启动类**

启动类需要添加Servlet的支持

**@SpringBootApplication**

**public** **class** **JpaThymeleafApplication** **extends** SpringBootServletInitializer **{**

**@Override**

**protected** SpringApplicationBuilder **configure(**SpringApplicationBuilder application**)** **{**

**return** application**.**sources**(**JpaThymeleafApplication**.**class**);**

**}**

**public** **static** **void** **main(**String**[]** args**)** **throws** Exception **{**

SpringApplication**.**run**(**JpaThymeleafApplication**.**class**,** args**);**

**}**

**}**

**数据库层代码**

实体类映射数据库表

**@Entity**

**public** **class** **User** **{**

**@Id**

**@GeneratedValue**

**private** **long** id**;**

**@Column(**nullable **=** **false,** unique **=** **true)**

**private** String userName**;**

**@Column(**nullable **=** **false)**

**private** String password**;**

**@Column(**nullable **=** **false)**

**private** **int** age**;**

**...**

**}**

继承JpaRepository类会自动实现很多内置的方法，包括增删改查。也可以根据方法名来自动生成相关sql，具体可以参考：[springboot(五)：spring data jpa的使用](http://www.ityouknow.com/springboot/2016/08/20/springboot(%E4%BA%94)-spring-data-jpa%E7%9A%84%E4%BD%BF%E7%94%A8.html)

**public** **interface** **UserRepository** **extends** JpaRepository**<**User**,** Long**>** **{**

User **findById(long** id**);**

Long **deleteById(**Long id**);**

**}**

**业务层处理**

service调用jpa实现相关的增删改查，实际项目中service层处理具体的业务代码。

**@Service**

**public** **class** **UserServiceImpl** **implements** UserService**{**

**@Autowired**

**private** UserRepository userRepository**;**

**@Override**

**public** List**<**User**>** **getUserList()** **{**

**return** userRepository**.**findAll**();**

**}**

**@Override**

**public** User **findUserById(long** id**)** **{**

**return** userRepository**.**findById**(**id**);**

**}**

**@Override**

**public** **void** **save(**User user**)** **{**

userRepository**.**save**(**user**);**

**}**

**@Override**

**public** **void** **edit(**User user**)** **{**

userRepository**.**save**(**user**);**

**}**

**@Override**

**public** **void** **delete(long** id**)** **{**

userRepository**.**delete**(**id**);**

**}**

**}**

Controller负责接收请求，处理完后将页面内容返回给前端。

**@Controller**

**public** **class** **UserController** **{**

**@Resource**

UserService userService**;**

**@RequestMapping(**"/"**)**

**public** String **index()** **{**

**return** "redirect:/list"**;**

**}**

**@RequestMapping(**"/list"**)**

**public** String **list(**Model model**)** **{**

List**<**User**>** users**=**userService**.**getUserList**();**

model**.**addAttribute**(**"users"**,** users**);**

**return** "user/list"**;**

**}**

**@RequestMapping(**"/toAdd"**)**

**public** String **toAdd()** **{**

**return** "user/userAdd"**;**

**}**

**@RequestMapping(**"/add"**)**

**public** String **add(**User user**)** **{**

userService**.**save**(**user**);**

**return** "redirect:/list"**;**

**}**

**@RequestMapping(**"/toEdit"**)**

**public** String **toEdit(**Model model**,**Long id**)** **{**

User user**=**userService**.**findUserById**(**id**);**

model**.**addAttribute**(**"user"**,** user**);**

**return** "user/userEdit"**;**

**}**

**@RequestMapping(**"/edit"**)**

**public** String **edit(**User user**)** **{**

userService**.**edit**(**user**);**

**return** "redirect:/list"**;**

**}**

**@RequestMapping(**"/delete"**)**

**public** String **delete(**Long id**)** **{**

userService**.**delete**(**id**);**

**return** "redirect:/list"**;**

**}**

**}**

* return "user/userEdit"; 代表会直接去resources目录下找相关的文件。
* return "redirect:/list"; 代表转发到对应的controller，这个示例就相当于删除内容之后自动调整到list请求，然后再输出到页面。

**页面内容**

list列表

**<!DOCTYPE html>**

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8"/>

<title>userList</title>

<link rel="stylesheet" th:href="@{/css/bootstrap.css}"></link>

</head>

<body class="container">

<br/>

<h1>用户列表</h1>

<br/><br/>

<div class="with:80%">

<table class="table table-hover">

<thead>

<tr>

<th>#</th>

<th>User Name</th>

<th>Password</th>

<th>Age</th>

<th>Edit</th>

<th>Delete</th>

</tr>

</thead>

<tbody>

<tr th:each="user : ${users}">

<th scope="row" th:text="${user.id}">1</th>

<td th:text="${user.userName}">neo</td>

<td th:text="${user.password}">Otto</td>

<td th:text="${user.age}">6</td>

<td><a th:href="@{/toEdit(id=${user.id})}">edit</a></td>

<td><a th:href="@{/delete(id=${user.id})}">delete</a></td>

</tr>

</tbody>

</table>

</div>

<div class="form-group">

<div class="col-sm-2 control-label">

<a href="/toAdd" th:href="@{/toAdd}" class="btn btn-info">add</a>

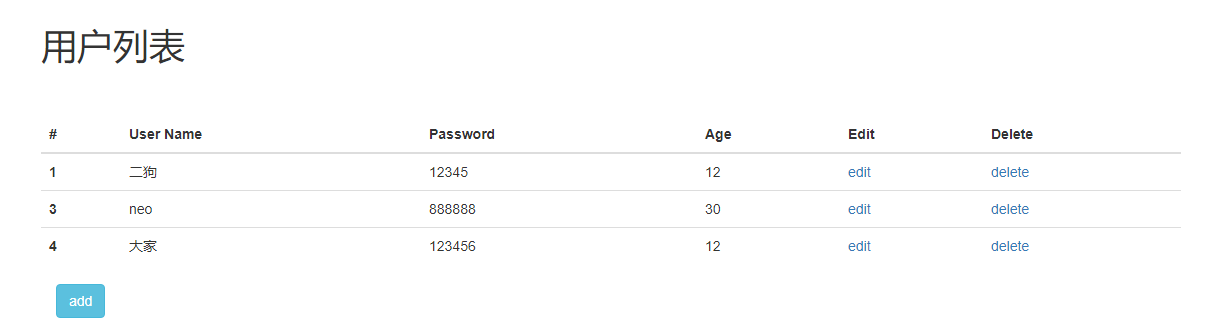
</div>

</div>

</body>

</html>

效果图：



<tr th:each="user : ${users}"> 这里会从controler层model set的对象去获取相关的内容，th:each表示会循环遍历对象内容。

其实还有其它的写法，具体的语法内容可以参考这篇文章：[springboot(四)：thymeleaf使用详解](http://www.ityouknow.com/springboot/2016/05/01/springboot(%E5%9B%9B)-thymeleaf%E4%BD%BF%E7%94%A8%E8%AF%A6%E8%A7%A3.html)

修改页面：

**<!DOCTYPE html>**

<html lang="en" xmlns:th="http://www.thymeleaf.org">

<head>

<meta charset="UTF-8"/>

<title>user</title>

<link rel="stylesheet" th:href="@{/css/bootstrap.css}"></link>

</head>

<body class="container">

<br/>

<h1>修改用户</h1>

<br/><br/>

<div class="with:80%">

<form class="form-horizontal" th:action="@{/edit}" th:object="${user}" method="post">

<input type="hidden" name="id" th:value="\*{id}" />

<div class="form-group">

<label for="userName" class="col-sm-2 control-label">userName</label>

<div class="col-sm-10">

<input type="text" class="form-control" name="userName" id="userName" th:value="\*{userName}" placeholder="userName"/>

</div>

</div>

<div class="form-group">

<label for="password" class="col-sm-2 control-label" >Password</label>

<div class="col-sm-10">

<input type="password" class="form-control" name="password" id="password" th:value="\*{password}" placeholder="Password"/>

</div>

</div>

<div class="form-group">

<label for="age" class="col-sm-2 control-label">age</label>

<div class="col-sm-10">

<input type="text" class="form-control" name="age" id="age" th:value="\*{age}" placeholder="age"/>

</div>

</div>

<div class="form-group">

<div class="col-sm-offset-2 col-sm-10">

<input type="submit" value="Submit" class="btn btn-info" />

&nbsp; &nbsp; &nbsp;

<a href="/toAdd" th:href="@{/list}" class="btn btn-info">Back</a>

</div>

</div>

</form>

</div>

</body>

</html>

添加页面和修改类似就不在贴代码了。

效果图：



这样一个使用jpa和thymeleaf的增删改查示例就完成了。

当然所以的示例代码都在这里：

[**示例代码-github**](https://github.com/ityouknow/spring-boot-examples)

[**示例代码-码云**](https://gitee.com/ityouknow/spring-boot-examples)