



链滴

SpringBoot 入门教程 (六) | 用 JdbcTemplate 访问 Mysql

作者: [JavaFish](#)

原文链接: <https://ld246.com/article/1621174197347>

来源网站: [链滴](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

前言

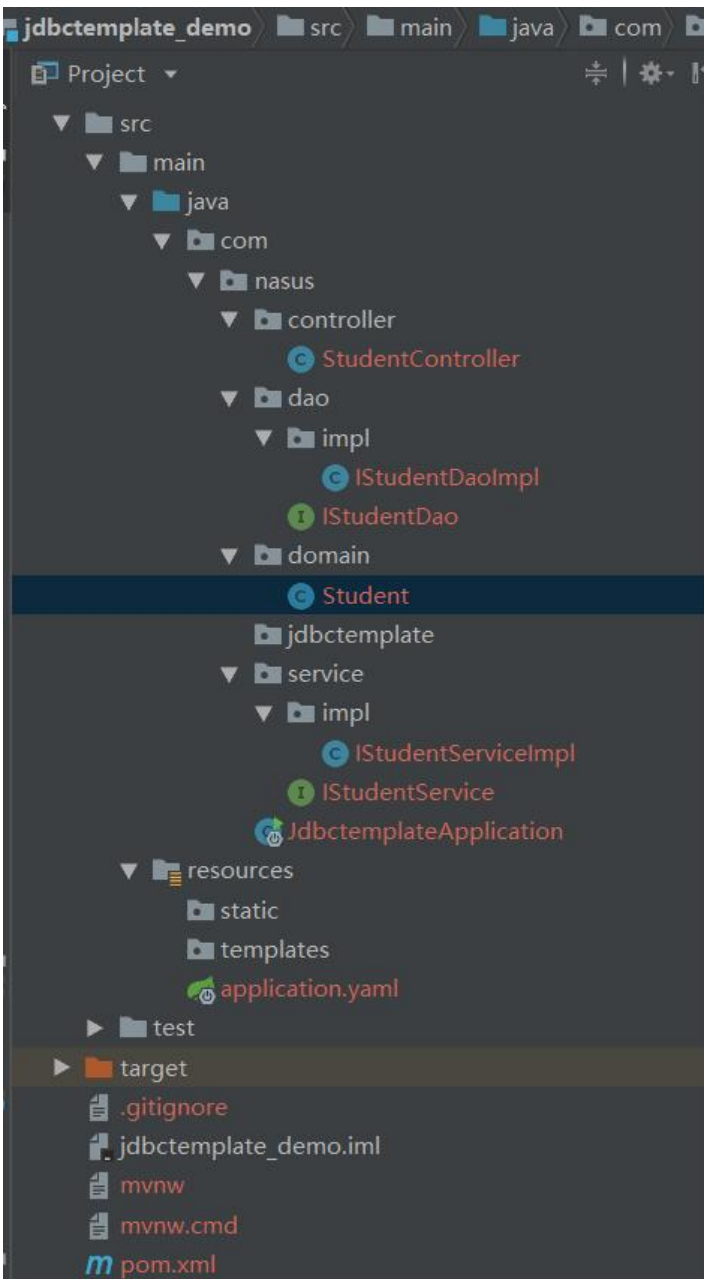
如题，今天介绍 springboot 通过jdbc访问关系型mysql,通过 spring 的 JdbcTemplate 去访问。

准备工作

- SpringBoot 2.x
- jdk 1.8
- maven 3.0
- idea
- mysql

构建 SpringBoot 项目，不会的朋友参考旧文章：[如何使用 IDEA 构建 Spring Boot 工程](#)

项目目录结构



pom 文件引入依赖

```
<dependencies>
  <!-- jdbcTemplate 依赖 -->
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-jdbc</artifactId>
  </dependency>
  <!-- 开启web: -->
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
  </dependency>

  <!-- mysql连接类 -->
  <dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <scope>runtime</scope>
  </dependency>
  <!-- https://mvnrepository.com/artifact/com.alibaba/druid -->
  <!-- druid 连接池-->
  <dependency>
    <groupId>com.alibaba</groupId>
    <artifactId>druid</artifactId>
    <version>1.1.13</version>
  </dependency>

  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
  </dependency>
</dependencies>
```

application.yaml 配置数据库信息

```
spring:
  datasource:
    driver-class-name: com.mysql.jdbc.Driver
    url: jdbc:mysql://127.0.0.1:3306/test?useUnicode=true&characterEncoding=utf8&serverTi
ezone=UTC&useSSL=true
    username: 数据库用户名
    password: 数据库密码
```

实体类

```
package com.nasus.domain;

/**
 * Project Name:jdbctemplate_demo <br/>
 * Package Name:com.nasus.domain <br/>
 * Date:2019/2/3 10:55 <br/>
```

```

* <b>Description:</b> TODO: 描述该类的作用 <br/>
*
* @author <a href="turodog@foxmail.com">nasus</a> <br/>
* Copyright Notice =====
=====
* This file contains proprietary information of Eastcom Technologies Co. Ltd.
* Copying or reproduction without prior written approval is prohibited.
* Copyright (c) 2019 =====
=====
*/
public class Student {

    private Integer id;

    private String name;

    private Integer age;

    public Integer getId() {
        return id;
    }

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

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public Integer getAge() {
        return age;
    }

    public void setAge(Integer age) {
        this.age = age;
    }

    @Override
    public String toString() {
        return "Student{" +
            "id=" + id +
            ", name='" + name + '\'' +
            ", age=" + age +
            '}';
    }
}

```

dao 层

```

package com.nasus.dao;

import com.nasus.domain.Student;
import java.util.List;

/**
 * Project Name:jdbctemplate_demo <br/>
 * Package Name:com.nasus.dao <br/>
 * Date:2019/2/3 10:59 <br/>
 * <b>Description:</b> TODO: 描述该类的作用 <br/>
 * @author <a href="mailto:turodog@foxmail.com">turodog</a> <br/>
 */
public interface IStudentDao {

    int add(Student student);

    int update(Student student);

    int delete(int id);

    Student findStudentById(int id);

    List<Student> findStudentList();

}

```

具体实现类:

```

package com.nasus.dao.impl;

import com.nasus.dao.IStudentDao;
import com.nasus.domain.Student;
import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.BeanPropertyRowMapper;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.stereotype.Repository;

/**
 * Project Name:jdbctemplate_demo <br/>
 * Package Name:com.nasus.dao.impl <br/>
 * Date:2019/2/3 11:00 <br/>
 * <b>Description:</b> TODO: 描述该类的作用 <br/>
 *
 * @author <a href="mailto:turodog@foxmail.com">turodog</a> <br/>
 */
@Repository
public class IStudentDaoImpl implements IStudentDao{

    @Autowired
    private JdbcTemplate jdbcTemplate;

    @Override
    public int add(Student student) {

```

```

        return jdbcTemplate.update("insert into student(name, age) values(?, ?)",
            student.getName(),student.getAge());
    }

    @Override
    public int update(Student student) {
        return jdbcTemplate.update("UPDATE student SET NAME=? ,age=? WHERE id=?",
            student.getName(),student.getAge(),student.getId());
    }

    @Override
    public int delete(int id) {
        return jdbcTemplate.update("DELETE from TABLE student where id=?",id);
    }

    @Override
    public Student findStudentById(int id) {
        // BeanPropertyRowMapper 使获取的 List 结果列表的数据库字段和实体类自动对应
        List<Student> list = jdbcTemplate.query("select * from student where id = ?", new Objec
        []{id}, new BeanPropertyRowMapper(Student.class));
        if(list!=null && list.size()>0){
            Student student = list.get(0);
            return student;
        }else{
            return null;
        }
    }

    @Override
    public List<Student> findStudentList() {
        // 使用Spring的JdbcTemplate查询数据库, 获取List结果列表, 数据库表字段和实体类自动对
        , 可以使用BeanPropertyRowMapper
        List<Student> list = jdbcTemplate.query("select * from student", new Object[]{}, new Be
        nPropertyRowMapper(Student.class));
        if(list!=null && list.size()>0){
            return list;
        }else{
            return null;
        }
    }
}

```

service 层

```

package com.nasus.service;

import com.nasus.domain.Student;
import java.util.List;

/**
 * Project Name:jdbctemplate_demo <br/>
 * Package Name:com.nasus.service <br/>
 * Date:2019/2/3 11:17 <br/>

```

```

* <b>Description:</b> TODO: 描述该类的作用 <br/>
*
* @author <a href="turodog@foxmail.com">nasus</a> <br/>
*/
public interface IStudentService {

    int add(Student student);

    int update(Student student);

    int delete(int id);

    Student findStudentById(int id);

    List<Student> findStudentList();

}

```

实现类:

```

package com.nasus.service.impl;

import com.nasus.dao.IStudentDao;
import com.nasus.domain.Student;
import com.nasus.service.IStudentService;
import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;

/**
 * Project Name:jdbctemplate_demo <br/>
 * Package Name:com.nasus.service.impl <br/>
 * Date:2019/2/3 11:18 <br/>
 * <b>Description:</b> TODO: 描述该类的作用 <br/>
 *
 * @author <a href="turodog@foxmail.com">nasus</a> <br/>
 * Copyright Notice =====
=====
 * This file contains proprietary information of Eastcom Technologies Co. Ltd.
 * Copying or reproduction without prior written approval is prohibited.
 * Copyright (c) 2019 =====
=====
 */
@Repository
public class IStudentServiceImpl implements IStudentService {

    @Autowired
    private IStudentDao iStudentDao;

    @Override
    public int add(Student student) {
        return iStudentDao.add(student);
    }
}

```

```

@Override
public int update(Student student) {
    return iStudentDao.update(student);
}

@Override
public int delete(int id) {
    return iStudentDao.delete(id);
}

@Override
public Student findStudentById(int id) {
    return iStudentDao.findStudentById(id);
}

@Override
public List<Student> findStudentList() {
    return iStudentDao.findStudentList();
}
}

```

controller 构建 restful api

```

package com.nasus.controller;

import com.nasus.domain.Student;
import com.nasus.service.IStudentService;
import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

/**
 * Project Name:jdbctemplate_demo <br/>
 * Package Name:com.nasus.controller <br/>
 * Date:2019/2/3 11:21 <br/>
 * <b>Description:</b> TODO: 描述该类的作用 <br/>
 *
 * @author <a href="mailto:turodog@foxmail.com">turodog</a> <br/>
 */
@RestController
@RequestMapping("/student")
public class StudentController {

    @Autowired
    private IStudentService iStudentService;
}

```



```

@PostMapping("")
public int addStudent(@RequestBody Student student){
    return iStudentService.add(student);
}

@PutMapping("/{id}")
public String updateStudent(@PathVariable Integer id, @RequestBody Student student){
    Student oldStudent = new Student();
    oldStudent.setId(id);
    oldStudent.setName(student.getName());
    oldStudent.setAge(student.getAge());
    int t = iStudentService.update(oldStudent);
    if (t == 1){
        return student.toString();
    }else {
        return "更新学生信息错误";
    }
}

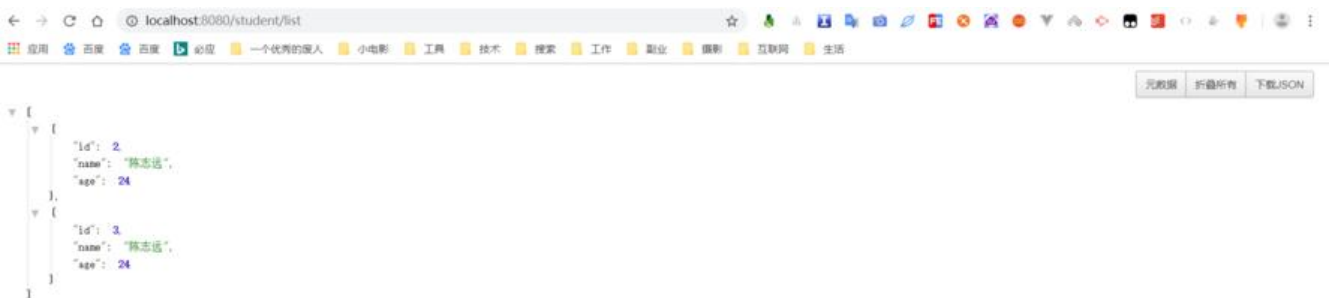
@GetMapping("/{id}")
public Student findStudentById(@PathVariable Integer id){
    return iStudentService.findStudentById(id);
}

@GetMapping("/list")
public List<Student> findStudentList(){
    return iStudentService.findStudentList();
}

@DeleteMapping("/{id}")
public int deleteStudentById(@PathVariable Integer id){
    return iStudentService.delete(id);
}
}

```

演示结果



其他的 api 测试可以通过 postman 测试。我这里已经全部测试通过，请放心使用。

源码下载: https://github.com/turoDog/Demo/tree/master/jdbctemplate_demo

后语

以上SpringBoot 用 JdbcTemplate 访问Mysql 的教程。

初次见面，也不知道送你们啥。干脆就送**几百本电子书**和**2021最新面试资料**吧。微信搜索**JavaFish** 复**电子书**送你 1000+ 本编程电子书；回复**面试**获取 50 套大厂面试题；回复**1024**送你一套完整的 jav 视频教程。

面试题都是有答案的，如下所示：有需要的就来拿吧，**绝对免费，无套路获取**。

The image shows a file explorer window with a sidebar on the left containing various folders such as '大数据', 'C 语言', 'C++', 'Java', 'Git', 'Python', 'Go 语言', 'Linux', '经典必读', '面试相关', '前端', '人工智能', and '设计模式'. The main pane displays a folder named '面试题' (Interview Questions) containing a list of PDF files. The files are listed with their names and modification dates (all 2021). The list includes:

名称	修改E
7道消息队列ActiveMQ面试题! .pdf	2021,
10道Java高级必备的Netty面试题! .pdf	2021,
10道Java面试必备的设计模式面试题!	2021,
10个Java经典的List面试题! .pdf	2021,
10个Java经典的Main方法面试题! .pdf	2021,
10个Java经典的String面试题! .pdf	2021,
15道经典的Tomcat面试题! .pdf	2021,
15道面试常问的Java多线程面试题! .pdf	2021,
17道消息队列Kafka面试题! .pdf	2021,
18道非常牛逼的Nginx面试题! .pdf	2021,
20道顶尖的Spring Boot面试题! .pdf	2021,
20道面试官常问的JVM面试题! .pdf	2021,
22道面试常问的SpringMVC面试题! .pdf	2021,
24道经典的英语面试题! .pdf	2021,
24道消息队列RabbitMQ面试题! .pdf	2021,
27道顶尖的Java多线程、锁、内存模型...	2021,
29道常见的Spring面试题! .pdf	2021,
30个Java经典的集合面试题! .pdf	2021,
36道面试常问的MyBatis面试题! .pdf	2021,
40道常问的Java多线程面试题! .pdf	2021,
55道BAT精选的Mysql面试题! .pdf	2021,
60道必备的Java核心技术面试题! .pdf	2021,
70道阿里巴巴高级Java面试题! .pdf	2021,
Java 面试题经典 77 问! .pdf	2021,
分布式缓存 Redis + Memcached 经典...	2021,
搞定 HR 面试的 40 个必备问题! .pdf	2021,
精选7道Elastic Search面试题! .pdf	2021,
精选8道Dubbo面试题! .pdf	2021,
精选17道海量数量处理面试题! .pdf	2021,
史上最全40道Dubbo面试题! .pdf	2021,
史上最全50道Redis面试题! .pdf	2021,



JavaFish

微信扫描二维码，关注我的公众号