

Quiz后端-2、测试Spring contrller接收网络请求

日期：2025.07.19 作者：tfzhang

后端主要提供用户与题目的注册、查询、删除等操作。使用到的开发工具：

- IDEA2024; 关注公众号“青椒工具”，发送“IDEA”，获取windows下的IDEA安装包
- mysql 5.7; 关注公众号“青椒工具”，发送“mysql”，获取windows下的mysql5.7安装包；

需求1：无参数请求

1 | 1、接收浏览器发起的/hello请求，给浏览器返回字符串"hello world"

步骤1：创建controller类

在com.tfzhang.quiz目录下创建controller这个目录，然后再创建HelloController这个类：

```
1 package com.tfzhang.quiz.controller;
2
3 import org.springframework.web.bind.annotation.RequestMapping;
4 import org.springframework.web.bind.annotation.RestController;
5
6 @RestController //表示当前类为请求处理类;
7 public class HelloController {
8
9     @RequestMapping("/hello") //表示如果接收到浏览器的/hello请求，就执行下面的
    hello()方法;
10     public String hello() {
11         System.out.println("hello");
12         return "Hello world";
13     }
14 }
```

步骤2：启动函数，实现浏览器访问

我们的用户数据库表如下：

```

1 DROP TABLE IF EXISTS user;
2
3 create table user
4 (
5     id            bigint auto_increment primary key,
6     userName      varchar(256)                null comment '用户名',
7     userPassword  varchar(512)                null comment '密码',
8     updateTime    datetime default CURRENT_TIMESTAMP null,
9     createTime    datetime default CURRENT_TIMESTAMP null,
10    isDelete      tinyint                    null,
11    userRole      int          default 0      null comment '表示用户角
色, 0 普通用户, 1 管理员'
12 )
13 comment '用户表';

```

userRole表示用户的身份, 1表示管理员, 表示普通用户;

isDelete用于逻辑删除, 也就是现实生产环境中, 删除往往只是逻辑删除, 不会真正从数据库中将数据删除。

1、MyBatis的通流程与之前类似:

步骤1: 创建数据库

步骤2: 创建数据库对应的实体类

对应上述的user数据表, 如下是对应的user实体类:

```

1 @Data
2 public class User {
3     /** 用户ID */
4     private Long id;
5
6     /** 用户名 */
7     private String userName;
8
9     /** 密码*/
10    private String userPassword;
11
12    /** 是否删除 */
13    private Integer isDelete;
14
15    /** 用户角色: 0-普通用户, 1-管理员 */
16    private Integer userRole;
17
18    /** 创建时间 */
19    private Date createTime;
20
21    /**更新时间 */
22    private Date updateTime;
23 }

```

步骤3: 创建MyBatis配置文件

步骤4: 编写SQL语句

之前的MyBatis测试案例的代码:

```

1 package com.tfzhang.quiz.mapper;
2
3 import com.tfzhang.quiz.model.User;
4 import org.apache.ibatis.annotations.Mapper;
5 import org.apache.ibatis.annotations.Select;
6
7 import java.util.List;
8
9 @Mapper //运行时，框架会自动实现实现类对象，并将对象实例交由IoC容器管理；
10 public interface UserMapper{
11
12     //查询全部用户；
13     @Select("select * from user")
14     public List<User> list();
15 }

```

现在要新增用户，对应的sql语句如下：

```

1 insert into user(userName,
2   userPassword,isDelete,userRole,createTime,updateTime)
   values('tom', '123456', 0, 0, now(), now(),);

```

对应的java代码：

```

1 @Insert("insert into user(username, userPassword,updateTime, createTime,
2   isDelete,userRole)" +
3   "values(#{userName}, #{userPassword}, #{isDelete}, #{userRole}, #
4   {createTime}, #{updateTime})")
5 @Options(useGeneratedKeys = true, keyProperty = "id")
6 public int insert(User user);

```

步骤5：编写测试类

将我们的测试list()的代码写在test.com.tfzhang.quiz.QuizApplicationTests.java中；

```

1 package com.tfzhang.quiz;
2
3 import com.tfzhang.quiz.mapper.UserMapper;
4 import com.tfzhang.quiz.model.User;
5 import org.junit.jupiter.api.Test;
6 import org.springframework.beans.factory.annotation.Autowired;
7 import org.springframework.boot.test.context.SpringBootTest;
8
9 import java.util.List;
10
11 @SpringBootTest
12 class QuizApplicationTests {
13
14     @Autowired //依赖注入；
15     private UserMapper userMapper;
16
17     @Test
18     public void testInsert(){
19         User user = new User();
20         user.set...
21

```

```
22         userMapper.insert(user);
23     }
24 }
```