MongoDB学习小记-文章评论小案例

一、需求分析

实现

- 基本的增删改查API
- 根据文章id查询评论
- 评论点赞
- 主要是数据的操作，UI层不再详细展示

二、表结构分析

2021/2/3 16:40:50

数据库：articledb

专栏文章评论	comment		
字段名称	字段含义	字段类型	备注
_id	ID	ObjectId或String	Mongo的主键的字段
articleid	文章ID	String	
content	评论内容	String	
userid	评论人ID	String	
nickname	评论人昵称	String	
createdatetime	评论的日期时间	Date	
likenum	点赞数	Int32	
replynum	回复数	Int32	
state	状态	String	0：不可见；1：可见；
parentid	上级ID	String	如果为0表示文章的顶级评论

三、技术选型

- mongodb-driver：mongoDB连接Java的驱动包相当于JDBC的驱动包
- SpringDataMongoDB：SpringData的家族成员之一，用于操作MongoDB的持久层框架，封装了底层Mongodb-driver
- 微服务模块搭建

四、实现

4.1查询Comment表的所有评论、新增评论

1. 依赖

```
<dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
```

```
            <artifactId>spring-boot-starter-data-mongodb</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>

        <dependency>
            <groupId>org.projectlombok</groupId>
            <artifactId>lombok</artifactId>
            <optional>true</optional>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>
        <dependency>
            <groupId>junit</groupId>
            <artifactId>junit</artifactId>
            <scope>test</scope>
        </dependency>
    </dependencies>
```

2. pojo

```
package henu.soft.xiaosi.pojo;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;
import org.springframework.data.annotation.Id;
import org.springframework.data.mongodb.core.index.Indexed;
import org.springframework.data.mongodb.core.mapping.Document;
import org.springframework.data.mongodb.core.mapping.Field;
import java.io.Serializable;
import java.time.LocalDateTime;
import java.util.Date;
/**
 * 文章评论实体类
 */
//把一个java类声明为mongodb的文档，可以通过collection参数指定这个类对应的文档。
//@Document(collection="mongodb 对应 collection 名")
// 若未加 @Document ，该 bean save 到 mongo 的 comment collection
// 若添加 @Document ，则 save 到 comment collection
@Document(collection="comment")//可以省略，如果省略，则默认使用类名小写映射集合
//复合索引
// @CompoundIndex( def = "{'userid': 1, 'nickname': -1}
@Data
@AllArgsConstructor
@NoArgsConstructor
public class Comment implements Serializable {
```

```
    //主键标识，该属性的值会自动对应mongodb的主键字段"_id"，如果该属性名就叫“id”,则该注解可以省略，否则必须写
    @Id
    private String id;//主键
//该属性对应mongodb的字段的名字，如果一致，则无需该注解
    @Field("content")
    private String content;//吐槽内容
    private Date publishtime;//发布日期
    //添加了一个单字段的索引
    @Indexed
    private String userid;//发布人ID
    private String nickname;//昵称
    private LocalDateTime createdatetime;//评论的日期时间
    private Integer likenum;//点赞数
    private Integer replynum;//回复数
    private String state;//状态
    private String parentid;//上级ID
    private String articleid;


}
```

3. CommentDao

```
package henu.soft.xiaosi.dao;

import henu.soft.xiaosi.pojo.Comment;
import org.springframework.data.mongodb.repository.MongoRepository;

public interface CommentDao extends MongoRepository<Comment,String> {
}
```

4.CommentService

```
package henu.soft.xiaosi.service;

import henu.soft.xiaosi.dao.CommentDao;
import henu.soft.xiaosi.pojo.Comment;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.List;

@Service
public class CommentService {

    @Autowired
    private CommentDao commentDao;
```

```
    /**
     * 保存一个评论
     *
     * @param comment
     */
    public void saveComment(Comment comment) {
        //如果需要自定义主键，可以在这里指定主键；如果不指定主键，MongoDB会自动生成主键
        //设置一些默认初始值。。。
        //调用dao
        commentDao.save(comment);
    }

    /**
     * 更新评论
     *
     * @param comment
     */
    public void updateComment(Comment comment) {
        //调用dao
        commentDao.save(comment);
    }

    /**
     * 根据id删除评论
     *
     * @param id
     */
    public void deleteCommentById(String id) {
        //调用dao
        commentDao.deleteById(id);
    }

    /**
     * 查询所有评论
     *
     * @return
     */
    public List<Comment> findCommentList() {
        //调用dao
        return commentDao.findAll();
    }

    /**
     * 根据id查询评论
     *
     * @param id
     * @return
     */
    public Comment findCommentById(String id) {
        //调用dao
        return commentDao.findById(id).get();
    }


}
```

5. Test测试类

```
package henu.soft.xiaosi;

import henu.soft.xiaosi.pojo.Comment;
import henu.soft.xiaosi.service.CommentService;
import org.junit.jupiter.api.Test;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.test.context.SpringBootTest;

import java.time.LocalDateTime;
import java.util.List;

@SpringBootTest
class DemoApplicationTests {
    @Autowired
    CommentService commentService;

    @Test
    void contextLoads() {
    }

    @Test
    void testFindAll(){
        List<Comment> commentList = commentService.findCommentList();
        System.out.println(commentList);
    }

    @Test
    void testSaveComment(){
        Comment comment=new Comment();
        //comment.setArticleid("100000");
        comment.setContent("我是马儿扎哈");
        comment.setCreatedatetime(LocalDateTime.now());
        comment.setUserid("1003");
        comment.setNickname("mezh");
        comment.setState("1");
        comment.setLikenum(0);
        comment.setReplynum(0);
        commentService.saveComment(comment);
    }


}
```

4.2 根据上级ID查询文章子评论的分页列表

分页查询

1.编写CommentDao中的方法

```
package henu.soft.xiaosi.dao;

import henu.soft.xiaosi.pojo.Comment;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.Pageable;
import org.springframework.data.mongodb.repository.MongoRepository;

public interface CommentDao extends MongoRepository<Comment,String> {

    //自定义方法,方法名不能乱写,parentid为pojo内的上级ID 根据上级ID查询文章评论的分页列表
    Page<Comment> findByParentid(String parentid, Pageable pageable);
}
```

2.编写CommentService中的方法

```
public Page<Comment> findCommentListByParentid(String parentid,int page,int
size){
        return commentDao.findByParentid(parentid, PageRequest.of(page -
1,size));
    }
```

3.测试

```
 @Test
    void testFindCommentListByParentid(){
        Page<Comment> page = commentService.findCommentListByParentid("3", 1,
2);
        System.out.println(page.getTotalElements());
        System.out.println(page.getContent());
    }

//输出结果:
//1
//[Comment(id=601a764932d0de2c78849218, content=learn MongoDB!,
publishtime=null, userid=null, nickname=xiaosi, createdatetime=null,
likenum=null, replynum=null, state=null, parentid=3, articleid=null)]
```

4.3MongoDBTemplate实现评论点赞

1.传统思路

- 根据点击的评论id获取Comment，然后将 set 点赞数 + 1，最后保存这个更新后的Comment
- 该方法虽然看起来比较简单，但是执行效率不高，涉及太多操作

2.使用MongoTemplate类实现对某列的操作

- 在Service中直接注入MongoTemplate

- 在Service中定义对点赞数列的修改方法`updateCommentLikenum()`
- 测试

代码

```
/**
     * 查询、更新点赞数
     */
    public void updateCommentLikenum(String id){
        //查询条件
        Query query = Query.query(Criteria.where("_id").is(id));

        //更新条件
        Update update = new Update();
        //可以指定更新的步长，默认为1，指定更显的列
        update.inc("likenum");

        mongoTemplate.updateFirst(query,update,Comment.class);
    }
```

```
@Test
    void testUpdateCommentLikenum(){
        commentService.updateCommentLikenum("192");
    }
```

效果图