

springboot邮件服务

本项目使用邮件服务的原因是想要提供找回密码的功能。下面将分三部分来讲解。

- springboot -email配置
- springboot- email服务类的编写
- 找回密码的功能

基础配置主要有两部分

1. pom.xml

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

1. .properties文件

```
spring.mail.host: smtp.163.com
spring.mail.username: 你的邮箱
spring.mail.password: 你的邮箱代理密码
spring.mail.properties.mail.smtp.auth: true
spring.mail.properties.mail.smtp.starttls.enable: true
spring.mail.properties.mail.smtp.starttls.required: true
```

我这里用的是163邮箱来实现发邮件的，如果你想使用别的邮箱发邮件，那你可能需要百度一下。

对了，你还需要登录网页版的163邮箱，去允许使用代理。

编写邮箱服务类

```
package com.example.demo.service;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.mail.SimpleMailMessage;
import org.springframework.mail.javamail.JavaMailSender;
import org.springframework.scheduling.annotation.Async;
import org.springframework.stereotype.Service;

/**
 * @ClassName MailService
 * @Description TODO
 * @Author ydc
 * @Date 2019/2/10 8:43
 * @Version 1.0
 */
```

```

/**
@Service
public class MailService {
    private JavaMailSender mailSender; //框架自带的

    @Autowired
    public void setMailSender(JavaMailSender mailSender) {
        this.mailSender = mailSender;
    }

    @Value("${spring.mail.username}") //发送人的邮箱 比如155156641XX@163.c
om,这里是获取了你properties里的信息
    private String from;

    @Async //意思是异步调用这个方法
    public void sendMail(String title, String url, String email) {
        SimpleMailMessage message = new SimpleMailMessage();
        message.setFrom(from); // 发送人的邮箱
        message.setSubject(title); //标题
        message.setTo(email); //发给谁 对方邮箱
        message.setText(url); //内容
        mailSender.send(message); //发送
    }
}

```

有了上个这个类，你就可以使用发送邮件的功能了。

找回密码功能

在login页面有个“忘记密码”，点击以后即可进入下面的页面：



The screenshot shows a web page for password reset. At the top, there is a dark blue header with the word "SHUE" in white. Below the header, the page title "重置密码" (Reset Password) is centered. A prompt "请填写您的登录邮箱来收取密码重置邮件" (Please fill in your login email to receive the password reset email) is displayed. Below the prompt is a text input field with the placeholder "输入邮箱" (Enter email) and an envelope icon on the right. Underneath the input field is a large green button with the text "发送重置邮件" (Send reset email). At the bottom of the page, the copyright information "© 2019 - 2020 ydc LiuwenLi Johnson" is shown.

这个页面对应于项目里的forgetPassword.html页面。

前端页面代码不在分析，可以直接到项目里看，因为逻辑非常简单

后端代码：

```
@RequestMapping("user/sendForgotPasswordEmail")
public ResponseData forgetEmail(String email) {
    //接收到前端发来的邮箱地址
    try {
        User registerUser = userServiceImp.getUserByEmail(email);
        if (registerUser == null) {
            //先检测这个邮箱在不在数据库里面
            return new ResponseData(ExceptionMsg.EmailNotRegister);
        }
        Random random = new Random();
        StringBuffer secretKey = new StringBuffer();
        for (int i = 0; i < 4; i++) {
            //利用随机数生成4位验证码
            int tmp = random.nextInt(9);
            secretKey.append(tmp);
        }
        String ans = secretKey.toString();
        registerUser.setValidcode(ans);
        //将验证码存到数据库里，为后面的验证做准备
        userServiceImp.updateUserValidCode(ans, registerUser.getUserId
    ());

        //发送邮件
        mailService.sendMail("SHUE密码找回服务", ans, email);
        return new ResponseData(ExceptionMsg.SUCCESS);
    } catch (Exception e) {
        e.printStackTrace();
        return new ResponseData(ExceptionMsg.FAILED);
    }
}
```

当验证成功以后，我不可能把原密码发给用户，因为真正的项目里，我是不能知道你的密码的，所以，我直接给你一个修改密码的页面，你修改完密码以后再去登录即可。

修改密码的页面为newPassword.html。这里不再分析这个页面的逻辑实现。