springboot(十)：邮件服务

springboot仍然在狂速发展，才五个多月没有关注，现在看[官网](http://projects.spring.io/spring-boot/)已经到1.5.3.RELEASE版本了。准备慢慢在写写springboot相关的文章，本篇文章使用springboot最新版本1.5.3进行开发。

发送邮件应该是网站的必备功能之一，什么注册验证，忘记密码或者是给用户发送营销信息。最早期的时候我们会使用JavaMail相关api来写发送邮件的相关代码，后来spring推出了JavaMailSender更加简化了邮件发送的过程，在之后springboot对此进行了封装就有了现在的spring-boot-starter-mail,本章文章的介绍主要来自于此包。

**简单使用**

**1、pom包配置**

pom包里面添加spring-boot-starter-mail包引用

<dependencies>

<dependency>

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

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

</dependency>

</dependencies>

**2、在application.properties中添加邮箱配置**

spring.mail.host=smtp.qiye.163.com //邮箱服务器地址

spring.mail.username=xxx@oo.com //用户名

spring.mail.password=xxyyooo //密码

spring.mail.default-encoding=UTF-8

mail.fromMail.addr=xxx@oo.com //以谁来发送邮件

**3、编写mailService,这里只提出实现类。**

**@Component**

**public** **class** **MailServiceImpl** **implements** MailService**{**

**private** **final** Logger logger **=** LoggerFactory**.**getLogger**(this.**getClass**());**

**@Autowired**

**private** JavaMailSender mailSender**;**

**@Value(**"${mail.fromMail.addr}"**)**

**private** String from**;**

**@Override**

**public** **void** **sendSimpleMail(**String to**,** String subject**,** String content**)** **{**

SimpleMailMessage message **=** **new** SimpleMailMessage**();**

message**.**setFrom**(**from**);**

message**.**setTo**(**to**);**

message**.**setSubject**(**subject**);**

message**.**setText**(**content**);**

**try** **{**

mailSender**.**send**(**message**);**

logger**.**info**(**"简单邮件已经发送。"**);**

**}** **catch** **(**Exception e**)** **{**

logger**.**error**(**"发送简单邮件时发生异常！"**,** e**);**

**}**

**}**

**}**

**4、编写test类进行测试**

**@RunWith(**SpringRunner**.**class**)**

**@SpringBootTest**

**public** **class** **MailServiceTest** **{**

**@Autowired**

**private** MailService MailService**;**

**@Test**

**public** **void** **testSimpleMail()** **throws** Exception **{**

MailService**.**sendSimpleMail**(**"ityouknow@126.com"**,**"test simple mail"**,**" hello this is simple mail"**);**

**}**

**}**

至此一个简单的文本发送就完成了。

**加点料**

但是在正常使用的过程中，我们通常在邮件中加入图片或者附件来丰富邮件的内容，下面讲介绍如何使用springboot来发送丰富的邮件。

**发送html格式邮件**

其它都不变在MailService添加sendHtmlMail方法.

**public** **void** **sendHtmlMail(**String to**,** String subject**,** String content**)** **{**

MimeMessage message **=** mailSender**.**createMimeMessage**();**

**try** **{**

*//true表示需要创建一个multipart message*

MimeMessageHelper helper **=** **new** MimeMessageHelper**(**message**,** **true);**

helper**.**setFrom**(**from**);**

helper**.**setTo**(**to**);**

helper**.**setSubject**(**subject**);**

helper**.**setText**(**content**,** **true);**

mailSender**.**send**(**message**);**

logger**.**info**(**"html邮件发送成功"**);**

**}** **catch** **(**MessagingException e**)** **{**

logger**.**error**(**"发送html邮件时发生异常！"**,** e**);**

**}**

**}**

在测试类中构建html内容，测试发送

**@Test**

**public** **void** **testHtmlMail()** **throws** Exception **{**

String content**=**"<html>\n" **+**

"<body>\n" **+**

" <h3>hello world ! 这是一封Html邮件!</h3>\n" **+**

"</body>\n" **+**

"</html>"**;**

MailService**.**sendHtmlMail**(**"ityouknow@126.com"**,**"test simple mail"**,**content**);**

**}**

**发送带附件的邮件**

在MailService添加sendAttachmentsMail方法.

**public** **void** **sendAttachmentsMail(**String to**,** String subject**,** String content**,** String filePath**){**

MimeMessage message **=** mailSender**.**createMimeMessage**();**

**try** **{**

MimeMessageHelper helper **=** **new** MimeMessageHelper**(**message**,** **true);**

helper**.**setFrom**(**from**);**

helper**.**setTo**(**to**);**

helper**.**setSubject**(**subject**);**

helper**.**setText**(**content**,** **true);**

FileSystemResource file **=** **new** FileSystemResource**(new** File**(**filePath**));**

String fileName **=** filePath**.**substring**(**filePath**.**lastIndexOf**(**File**.**separator**));**

helper**.**addAttachment**(**fileName**,** file**);**

mailSender**.**send**(**message**);**

logger**.**info**(**"带附件的邮件已经发送。"**);**

**}** **catch** **(**MessagingException e**)** **{**

logger**.**error**(**"发送带附件的邮件时发生异常！"**,** e**);**

**}**

**}**

添加多个附件可以使用多条 helper.addAttachment(fileName, file)

在测试类中添加测试方法

**@Test**

**public** **void** **sendAttachmentsMail()** **{**

String filePath**=**"e:\\tmp\\application.log"**;**

mailService**.**sendAttachmentsMail**(**"ityouknow@126.com"**,** "主题：带附件的邮件"**,** "有附件，请查收！"**,** filePath**);**

**}**

**发送带静态资源的邮件**

邮件中的静态资源一般就是指图片，在MailService添加sendAttachmentsMail方法.

**public** **void** **sendInlineResourceMail(**String to**,** String subject**,** String content**,** String rscPath**,** String rscId**){**

MimeMessage message **=** mailSender**.**createMimeMessage**();**

**try** **{**

MimeMessageHelper helper **=** **new** MimeMessageHelper**(**message**,** **true);**

helper**.**setFrom**(**from**);**

helper**.**setTo**(**to**);**

helper**.**setSubject**(**subject**);**

helper**.**setText**(**content**,** **true);**

FileSystemResource res **=** **new** FileSystemResource**(new** File**(**rscPath**));**

helper**.**addInline**(**rscId**,** res**);**

mailSender**.**send**(**message**);**

logger**.**info**(**"嵌入静态资源的邮件已经发送。"**);**

**}** **catch** **(**MessagingException e**)** **{**

logger**.**error**(**"发送嵌入静态资源的邮件时发生异常！"**,** e**);**

**}**

**}**

在测试类中添加测试方法

**@Test**

**public** **void** **sendInlineResourceMail()** **{**

String rscId **=** "neo006"**;**

String content**=**"<html><body>这是有图片的邮件：<img src=\'cid:" **+** rscId **+** "\' ></body></html>"**;**

String imgPath **=** "C:\\Users\\summer\\Pictures\\favicon.png"**;**

mailService**.**sendInlineResourceMail**(**"ityouknow@126.com"**,** "主题：这是有图片的邮件"**,** content**,** imgPath**,** rscId**);**

**}**

添加多个图片可以使用多条 <img src='cid:" + rscId + "' > 和helper.addInline(rscId, res) 来实现

到此所有的邮件发送服务已经完成了。

**邮件系统**

上面发送邮件的基础服务就这些了，但是如果我们要做成一个邮件系统的话还需要考虑以下几个问题：

**邮件模板**

我们会经常收到这样的邮件：

尊敬的neo用户：

恭喜您注册成为xxx网的用户,，同时感谢您对xxx的关注与支持并欢迎您使用xx的产品与服务。

...

其中只有neo这个用户名在变化，其它邮件内容均不变，如果每次发送邮件都需要手动拼接的话会不够优雅，并且每次模板的修改都需要改动代码的话也很不方便，因此对于这类邮件需求，都建议做成邮件模板来处理。模板的本质很简单，就是在模板中替换变化的参数，转换为html字符串即可，这里以thymeleaf为例来演示。

**1、pom中导入thymeleaf的包**

<dependency>

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

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

</dependency>

**2、在resorces/templates下创建emailTemplate.html**

**<!DOCTYPE html>**

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

<head>

<meta charset="UTF-8"/>

<title>Title</title>

</head>

<body>

您好,这是验证邮件,请点击下面的链接完成验证,<br/>

<a href="#" th:href="@{ http://www.ityouknow.com/neo/{id}(id=${id}) }">激活账号</a>

</body>

</html>

**3、解析模板并发送**

**@Test**

**public** **void** **sendTemplateMail()** **{**

*//创建邮件正文*

Context context **=** **new** Context**();**

context**.**setVariable**(**"id"**,** "006"**);**

String emailContent **=** templateEngine**.**process**(**"emailTemplate"**,** context**);**

mailService**.**sendHtmlMail**(**"ityouknow@126.com"**,**"主题：这是模板邮件"**,**emailContent**);**

**}**

**发送失败**

因为各种原因，总会有邮件发送失败的情况，比如：邮件发送过于频繁、网络异常等。在出现这种情况的时候，我们一般会考虑重新重试发送邮件，会分为以下几个步骤来实现：

* 1、接收到发送邮件请求，首先记录请求并且入库。
* 2、调用邮件发送接口发送邮件，并且将发送结果记录入库。
* 3、启动定时系统扫描时间段内，未发送成功并且重试次数小于3次的邮件，进行再次发送

**异步发送**

很多时候邮件发送并不是我们主业务必须关注的结果，比如通知类、提醒类的业务可以允许延时或者失败。这个时候可以采用异步的方式来发送邮件，加快主交易执行速度，在实际项目中可以采用MQ发送邮件相关参数，监听到消息队列之后启动发送邮件。

可以参考前期文章：[springboot(八)：RabbitMQ详解](http://www.ityouknow.com/springboot/2016/11/30/springboot(%E5%85%AB)-RabbitMQ%E8%AF%A6%E8%A7%A3.html) 来实现。

[**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)

**参考:** [spring boot 发送邮件](http://blog.csdn.net/clementad/article/details/51833416)

**作者：纯洁的微笑**  
**出处：**[**http://www.ityouknow.com/**](http://www.ityouknow.com/springboot/2017/05/06/springboot-mail.html)   
**版权归作者所有，转载请注明出处**