Java的邮件开发

# 基本概念

## JavaMail

Java的邮件开发接口

## SMTP协议默认端口号25

Simple mail tansfer protocol（发送协议）

即简单的邮件传输协议，可以控制邮件的中转方式。

邮件服务器也称SMTP服务器

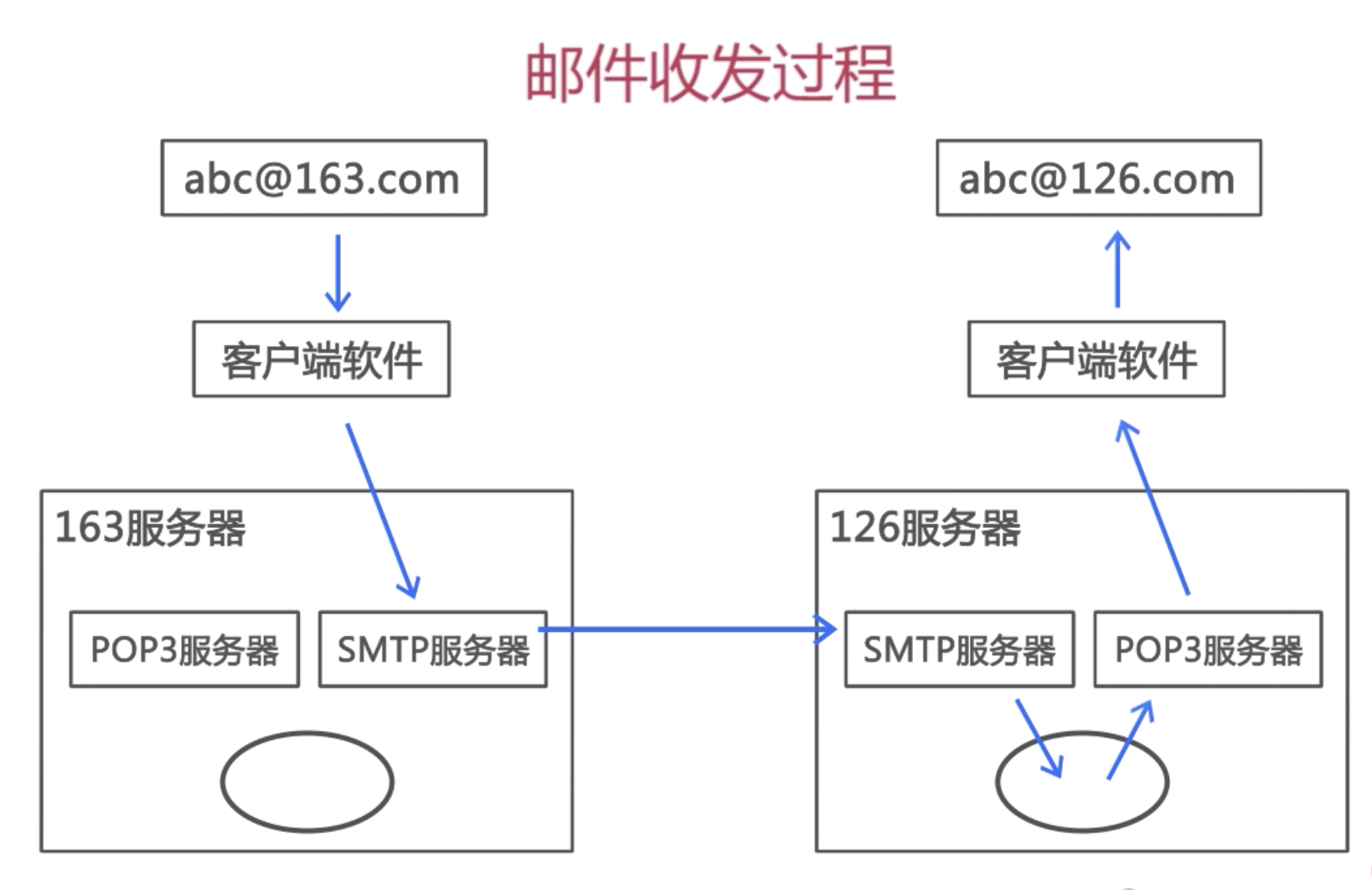
## POP3默认端口号110

Post office protocal -version3

接受协议

用于在客户端上远程管理邮件服务器上的邮件

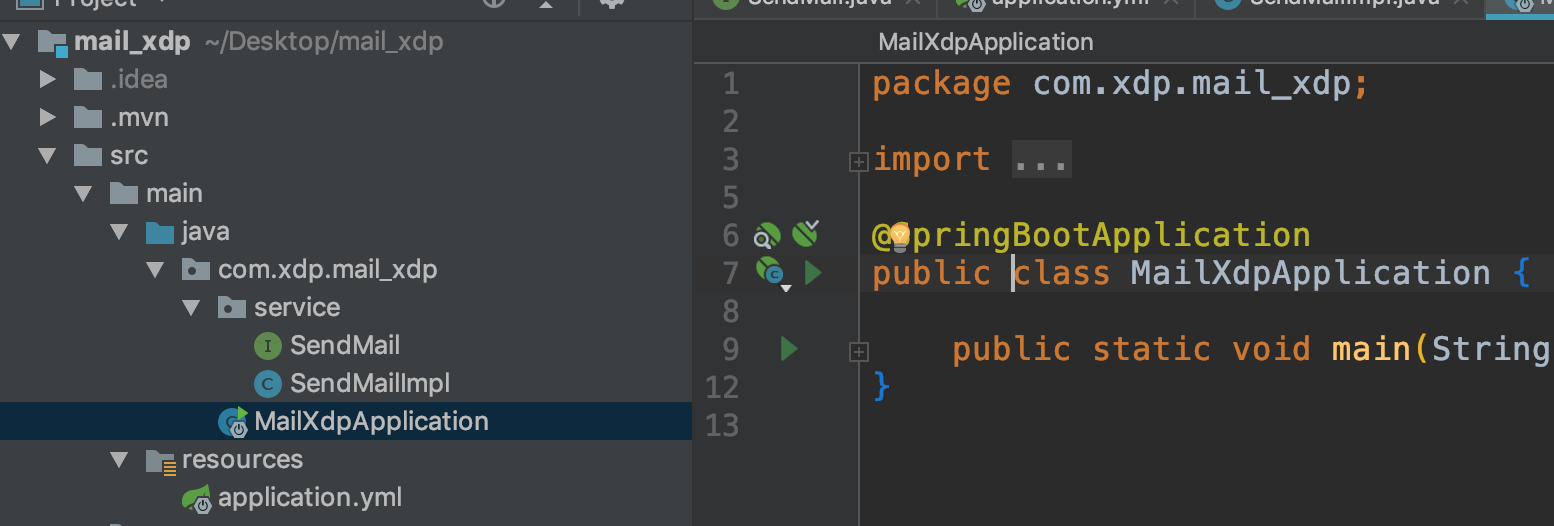
## 邮件收发过程



# SpringCloud实现发送邮件

## 创建项目

## 项目结构



## 项目依赖

*<*parent*>  
 <*groupId*>*org.springframework.boot*</*groupId*>  
 <*artifactId*>*spring-boot-starter-parent*</*artifactId*>  
 <*version*>*2.1.0.RELEASE*</*version*>  
 <*relativePath*/>* <!-- lookup parent from repository -->  
*</*parent*>  
  
<*properties*>  
 <*project.build.sourceEncoding*>*UTF-8*</*project.build.sourceEncoding*>  
 <*project.reporting.outputEncoding*>*UTF-8*</*project.reporting.outputEncoding*>  
 <*java.version*>*1.8*</*java.version*>  
</*properties*>  
  
<*dependencies*>  
 <*dependency*>  
 <*groupId*>*org.springframework.boot*</*groupId*>  
 <*artifactId*>*spring-boot-starter-mail*</*artifactId*>  
 </*dependency*>  
  
 <*dependency*>  
 <*groupId*>*org.springframework.boot*</*groupId*>  
 <*artifactId*>*spring-boot-starter-test*</*artifactId*>  
 <*scope*>*test*</*scope*>  
 </*dependency*>  
 <*dependency*>  
 <*groupId*>*org.springframework.boot*</*groupId*>  
 <*artifactId*>*spring-boot-starter-web*</*artifactId*>  
 </*dependency*>  
</*dependencies*>  
  
<*build*>  
 <*plugins*>  
 <*plugin*>  
 <*groupId*>*org.springframework.boot*</*groupId*>  
 <*artifactId*>*spring-boot-maven-plugin*</*artifactId*>  
 </*plugin*>  
 </*plugins*>  
</*build*>*

## 项目配置application.yml

**spring**:  
 *#配置邮箱服务器信息* **mail**:  
 **host**: smtp.163.com  
 **username**: m18703421242@163.com  
 **password**: xdp123456  
 **enable**: true  
 **smtp**:  
 **auth**: true  
 **starttls**:  
 **enable**: true  
 **required**: true  
*########################################***msg**:  
 **subject**: 注册成功邮件。  
 **text**: 恭喜您的账户{},注册成功!

## 编写代码

@SpringBootApplication  
public class MailXdpApplication *{* public static void main*(*String*[]* args*) {* SpringApplication.*run(*MailXdpApplication.class, args*)*;  
 *}  
}*

package com.xdp.mail\_xdp.service;  
  
import org.springframework.web.bind.annotation.RequestMapping;  
import org.springframework.web.bind.annotation.RequestParam;  
  
  
public interface SendMail *{* @RequestMapping*(*"/sendMail"*)* String sendMail*(* @RequestParam String toMail*)*;  
*}*

package com.xdp.mail\_xdp.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.util.StringUtils;  
import org.springframework.web.bind.annotation.RequestParam;  
import org.springframework.web.bind.annotation.RestController;  
  
@RestController  
public class SendMailImpl implements SendMail *{* public static final String *ERROR* = "error";  
 public static final String *SUCCESS* = "success";  
 @Value*(*"${msg.subject}"*)* private String subject;  
 @Value*(*"${msg.text}"*)* private String text;  
 @Value*(*"${spring.mail.username}"*)* private String fromEmail;  
 @Autowired  
 private JavaMailSender javaMailSender;  
 @Override  
 public String sendMail*(*@RequestParam String toMail*) {* if *(*StringUtils.*isEmpty(*toMail*)){* return *ERROR*;  
 *}* SimpleMailMessage simpleMailMessage = new SimpleMailMessage*()*;  
 simpleMailMessage.setFrom*(*fromEmail*)*;  
 simpleMailMessage.setTo*(*toMail*)*;  
 simpleMailMessage.setSubject*(*subject*)*;  
 simpleMailMessage.setText*(*text.replace*(*"{}",toMail*))*;  
 javaMailSender.send*(*simpleMailMessage*)*;  
 return *SUCCESS*;  
 *}  
}*

## 测试运行

