

EMail 보내기

안 화 수

Mail Server

❖ Mail Server

- Windows : Exchange Server
- Linux : SendMail, Qmail

Mail Server Protocol

❖ Mail Server Protocol

- Mail 송신 : SMTP(Simple Mail Transfer Protocol) - 25번
- Mail 수신 : POP3(Post Office Protocol 3) - 110번

Email SMTP Library

❖ Email 전송 라이브러리 종류

- Apache Commons Email 라이브러리
- Javax Mail 라이브러리

Naver Email 보내기

❖ Naver Email 보내기

1. Apache Commons Email 라이브러리 사용
2. Javax Mail 라이브러리 사용

Naver Mail Server 환경 설정

❖ Naver Mail Server 활용하기

Naver로 로그인후 [환경설정] - [POP3/IMAP 설정] - [POP3/SMTP 설정]탭 - [POP3/SMTP 사용함] - [확인]

NAVER 메일

환경 설정 | 메일로 돌아가기

기본 환경 설정 | 메일함 관리 | 메일 자동 분류 | 서명/빠른답장 | 부재 중 설정 | 새 메일 알림 설정

스팸 설정 | 외부 메일 가져오기 | **✓ POP3/IMAP 설정** | 단축키

POP3/SMTP 설정 | IMAP/SMTP 설정

휴대폰, 아웃룩 등에서 네이버 메일을 확인할 수 있도록 POP3/SMTP를 설정합니다.

giduck23 님은 현재 POP3/SMTP를 사용하고 있습니다.

POP3/SMTP 사용 | ☒ 사용함 | ☐ 사용 안 함

적용 범위 | ☐ 지금부터 새로 받는 메일만 받음 | ☐ 기존에 받은 메일을 포함하여 받음 | ☒ 이전에 설정한 시간 이후 수신한 메일만 받음(2016-02-29 14:05)

읽음 표시 | ☒ POP3로 읽어간 메일을 읽음 표시 | ☐ POP3로 읽어간 메일을 읽지 않음으로 표시

원본 저장 | ☒ 네이버 메일에 원본 저장 ? | ☐ 메일 프로그램 설정에 따라 저장 또는 삭제 ?

외부메일 처리 | ☒ POP3로 읽어갈 때 외부메일을 포함하지 않음 | ☐ POP3로 읽어갈 때 외부메일을 포함

기본 설정으로 | **확인** | 취소

용량 2.66GB / 5GB | **환경설정** | 스킨설정 | POP3 로그인 기록보기 | 모바일 메일 | ht © NAVER Corp. All Rights Reserved. | 스팸정책 | 공지사항 | 메일 고객센터

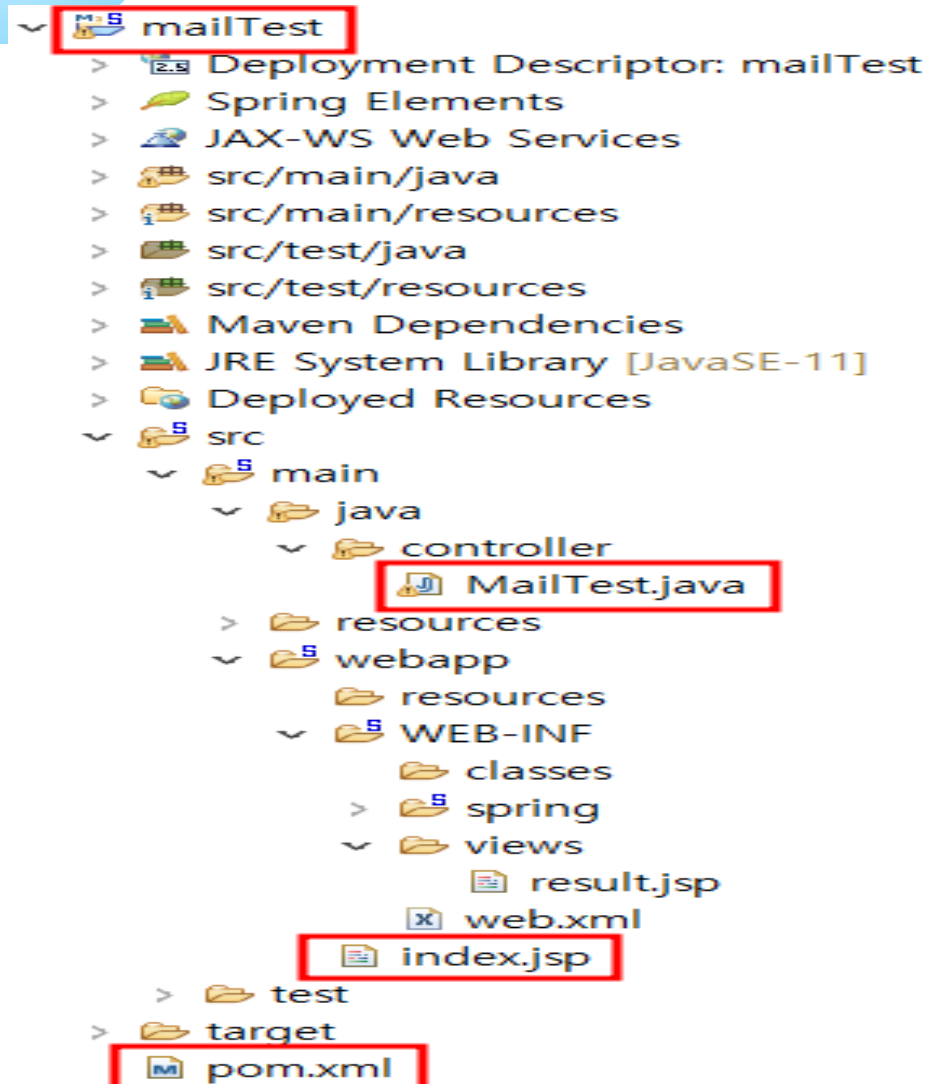
Naver Email 보내기

1. Apache Commons Email 라이브러리 사용

- 1) pom.xml 파일에 의존 라이브러리(common-email) 추가
- 2) controller 클래스에 email 전송용 code 작성

Naver Email 보내기

❖ mailTest 프로젝트 생성



Naver Email 보내기

❖ 의존 라이브러리 추가 : pom.xml

```
<dependencies>
```

```
    <!-- EMail -->
```

```
    <dependency>
```

```
        <groupId>org.apache.commons</groupId>
```

```
        <artifactId>commons-email</artifactId>
```

```
        <version>1.5</version>
```

```
    </dependency>
```

```
</dependencies>
```

Naver Email 보내기

❖ 메일 전송 : index.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"  
    pageEncoding="UTF-8"%>
```

```
<!DOCTYPE html">
```

```
<html>
```

```
<head>
```

```
<meta charset="UTF-8">
```

```
<title>Insert title here</title>
```

```
</head>
```

```
<body>
```

```
<a href="send.do">메일 전송</a> <br> <br>
```

```
</body>
```

```
</html>
```

Naver Email 보내기

❖ 컨트롤러 클래스 : MailTest.java (1/3)

```
@Controller
```

```
public class MailTest {
```

```
    @RequestMapping("/send.do")
```

```
    public String send(Model model) {
```

```
        Random random = new Random();
```

```
        int a = random.nextInt(100);
```

Naver Email 보내기

❖ 컨트롤러 클래스 : MailTest.java (2/3)

// Mail Server 설정

String charSet = "utf-8";

String hostSMTP = "smtp.naver.com";

String hostSMTPid = "giduck23@naver.com";

String hostSMTPpwd = "000000000"; // 비밀번호 입력

// 보내는 사람 EMail, 제목, 내용

String fromEmail = "giduck23@naver.com";

String fromName = "친절한 홍길동씨";

String subject = "Overflow인증메일입니다.";

// 받는 사람 E-Mail 주소

String mail = "giduck23@gmail.com";

Naver Email 보내기

❖ 컨트롤러 클래스 : MailTest.java (3/3)

```
try {  
    HtmlEmail email = new HtmlEmail();  
    email.setDebug(true);  
    email.setCharset(charSet);  
    email.setSSL(true);  
    email.setHostName(hostSMTP);  
    email.setSmtpport(587);  
    email.setAuthentication(hostSMTPid, hostSMTPpwd);  
    email.setTLS(true);  
    email.addTo(mail, charSet);  
    email.setFrom(fromEmail, fromName, charSet);  
    email.setSubject(subject);  
    email.setHtmlMsg("<p align = 'center'>Overflow에 오신것을 환영합니다.</p><br>"  
        + "<div align='center'> 인증번호 : " + a + "</div>");  
    email.send();  
} catch (Exception e) {  
    System.out.println(e);  
}  
  
model.addAttribute("result", "good~!!\n 등록된 E-Mail 확인");  
return "result";  
}
```

Naver Email 보내기

2. Javax Mail 라이브러리 사용

- 1) pom.xml 파일에 의존 라이브러리(mail, spring-context-support) 추가
- 2) root-context.xml 파일에 이메일 서버 설정을 위한 bean 생성
- 3) controller 클래스에 email 전송용 code 작성

Naver Email 보내기

❖ 의존 라이브러리 추가 : pom.xml

```
<dependencies>
```

```
    <!-- javax.mail -->
```

```
    <dependency>
```

```
        <groupId>javax.mail</groupId>
```

```
        <artifactId>mail</artifactId>
```

```
        <version>1.4.7</version>
```

```
    </dependency>
```

```
    <dependency>
```

```
        <groupId>org.springframework</groupId>
```

```
        <artifactId>spring-context-support</artifactId>
```

```
        <version>${org.springframework-version}</version>
```

```
    </dependency>
```

```
</dependencies>
```

Naver Email 보내기

❖ 이메일 서버 설정을 위한 bean 생성 : root-context.xml

```
<bean id="jMailSender" class="org.springframework.mail.javamail.JavaMailSenderImpl">
    <property name="host" value="smtp.naver.com" />
    <property name="port" value="465" />
    <property name="username" value="giduck23" />
    <property name="password" value="0000000" />
    <property value="smtps" name="protocol" />
    <property value="utf-8" name="defaultEncoding" />
    <property name="javaMailProperties">
        <props>
            <prop key="mail.smtp.auth">true</prop>
            <prop key="mail.smtp.starttls.enable">true</prop>
            <prop key="mail.smtps.ssl.checkserveridentity">true</prop>
            <prop key="mail.smtps.ssl.trust">*</prop>
            <prop key="mail.debug">true</prop>
        </props>
    </property>
</bean>
```


Naver Email 보내기

❖ 메일 전송 : mailform.jsp

E-Mail 보내기

보낸사람	<input type="text"/>
받는사람	<input type="text"/>
제목	<input type="text"/>
내용	<div><div></div></div>
<div>전송</div>	

Naver Email 보내기

❖ email 전송용 code 작성 : MailController.java (1/2)

```
@Controller
```

```
public class MailController {
```

```
    @Autowired
```

```
    private JavaMailSender jMailSender;
```

```
    @RequestMapping(value="mailform.do", method=RequestMethod.GET)
```

```
    public String mailform() {
```

```
        return "mailform";
```

```
    }
```

Naver Email 보내기

❖ email 전송용 code 작성 : MailController.java (2/2)

```
@RequestMapping(value="mailsend.do", method=RequestMethod.POST)
public String mailsend(Mail mail, Model model) {
    MimeMessage mms = jMailSender.createMimeMessage();
    try {
        MimeMessageHelper messageHelper = new MimeMessageHelper(mms, true, "utf-8");
        messageHelper.setSubject(mail.getSubject());
        messageHelper.setText(mail.getContent(), true);
        messageHelper.setFrom(mail.getSendmail());
        messageHelper.setTo(mail.getReceivemail());
        jMailSender.send(mms);
        model.addAttribute("result", 1);
        model.addAttribute("message", "입력하신 이메일로 발송");
    } catch (Exception e) {
        System.out.println(e.getMessage());
        model.addAttribute("result", -1);
        model.addAttribute("message", "메일 보내기 실패");
    }
    return "mailresult";
}
```