

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
- Jakarta Mail 라이브러리

Naver Email 보내기

❖ Naver Email 보내기

1. Apache Commons Email 라이브러리 사용
2. Jakarta 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/SMTP 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

1.Naver Email 보내기

1. Apache Commons Email 라이브러리 사용

- 1) 의존 라이브러리(common-email) 추가
- 2) Controller 클래스에 email 전송용 code 작성

1.Naver Email 보내기

❖ mailTest 프로젝트 생성

의존 라이브러리 추가

❖ 의존 라이브러리 추가 : **build.gradle**

```
dependencies {  
    // jsp  
    implementation 'org.apache.tomcat.embed:tomcat-embed-jasper'  
    // jakarta jstl  
    implementation 'org.glassfish.web:jakarta.servlet.jsp.jstl:3.0.0'  
    // email  
    implementation 'org.apache.commons:commons-email:1.5'  
  
    implementation 'org.springframework.boot:spring-boot-starter-web'  
    providedRuntime 'org.springframework.boot:spring-boot-starter-tomcat'  
    testImplementation 'org.springframework.boot:spring-boot-starter-test'  
}
```

환경 설정파일 수정

❖ 환경 설정 파일 수정

main / resources / **application.properties**

port

server.port=80

prefix and suffix

spring.mvc.view.prefix=/WEB-INF/views/

spring.mvc.view.suffix=.jsp

❖ webapp / **WEB-INF** / **views** 폴더 생성

1.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>
```

1.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";

1.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";  
}  
}
```

View 생성

❖ View 생성

webapp / WEB-INF / views - result.jsp 생성

```
<%@ 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>
```

결과 페이지 : \${result}

```
</body>  
</html>
```

2.Naver Email 보내기

2. jakarta mail 라이브러리 사용

- 1) build.gradle 파일에 의존 라이브러리 추가
- 2) application.properties 파일에 네이버 이메일 서버 설정
- 3) Controller 클래스에 email 전송용 code 작성

2.Naver Email 보내기

❖ navermail 프로젝트 생성

의존 라이브러리 추가

❖ 의존 라이브러리 추가 : **build.gradle**

```
dependencies {  
    // jsp  
    implementation 'org.apache.tomcat.embed:tomcat-embed-jasper'  
    // jakarta jstl  
    implementation 'org.glassfish.web:jakarta.servlet.jsp.jstl:3.0.0'  
    // email  
    implementation 'org.springframework.boot:spring-boot-starter-mail:3.0.5'  
    implementation 'org.springframework:spring-context:6.0.7'  
    implementation 'org.springframework:spring-context-support:6.0.7'  
    implementation 'com.sun.mail:jakarta.mail:2.0.1'  
    implementation 'org.springframework.boot:spring-boot-starter-web'  
    compileOnly 'org.projectlombok:lombok'  
    annotationProcessor 'org.projectlombok:lombok'  
    providedRuntime 'org.springframework.boot:spring-boot-starter-tomcat'  
    testImplementation 'org.springframework.boot:spring-boot-starter-test'  
}
```

환경 설정파일 수정

❖ 환경 설정 파일 수정

main / resources / **application.properties** (1/2)

port

server.port=80

prefix and suffix

spring.mvc.view.prefix=/WEB-INF/views/

spring.mvc.view.suffix=.jsp

❖ webapp / **WEB-INF** / **views** 폴더 생성

환경 설정파일 수정

❖ 환경 설정 파일 수정

main / resources / **application.properties** (2/2)

```
# naver email server
spring.mail.host=smtp.naver.coma
spring.mail.port=465
spring.mail.username=giduck23@naver.com
spring.mail.password=000000
spring.mail.properties.mail.smtp.auth=true
spring.mail.properties.mail.smtp.ssl.enable=true
spring.mail.properties.mail.smtp.ssl.trust=smtp.naver.com
spring.mail.properties.mail.smtp.starttls.enable=true
```

2.Naver Email 보내기

❖ 메일 전송 : mailform.jsp

EMail 보내기

보내는사람	<input type="text" value="naver EMail주소"/>
받는사람	<input type="text" value="받는사람 EMail주소"/>
제목	<input type="text"/>
내용	<div><div></div></div>
<input type="button" value="전송"/>	

2.Naver Email 보내기

❖ DTO : Mail.java

```
package com.example.demo.model;
```

```
import lombok.Data;
```

```
@Data
```

```
public class Mail {
```

```
    private String sendmail;
```

```
    private String receivedmail;
```

```
    private String subject;
```

```
    private String content;
```

```
}
```

2.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";
```

```
    }
```

2.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";
}
```


3.gmail 보내기

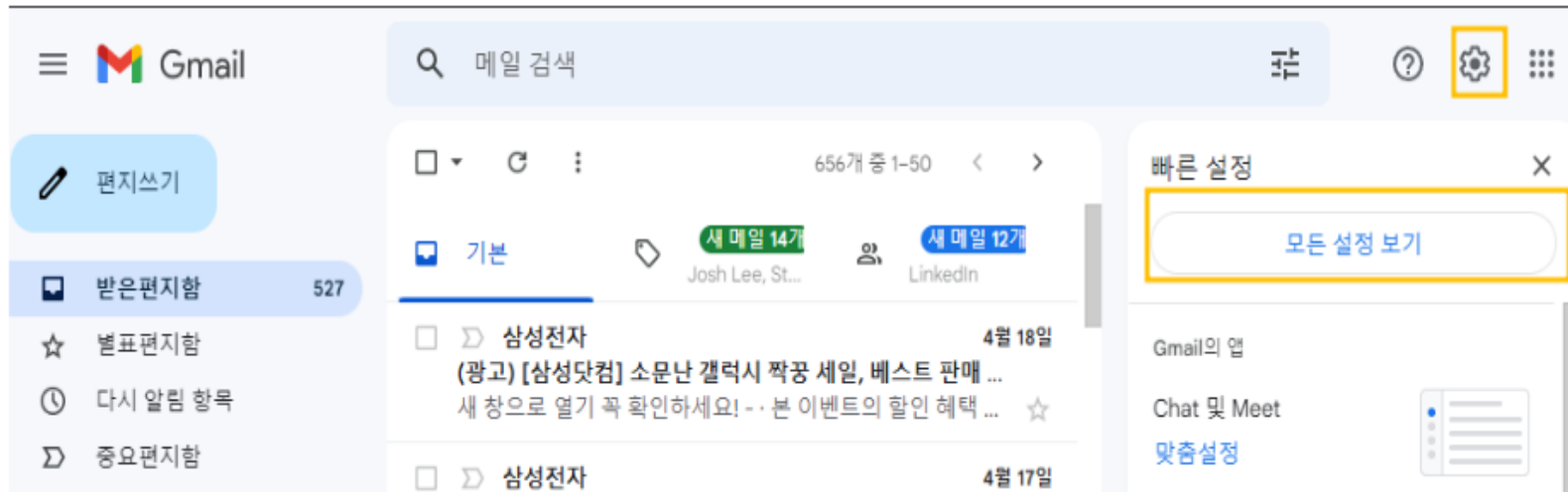
❖ Gmail SMTP Server 사용하기

1. gmail 설정에서 imap을 사용할 수 있도록 활성화 시킨다.
2. Gmail의 SMTP 서버에서는 2022년 5월 30일부터 사용자 이름과 비밀번호만으로 SMTP 서버를 사용할 수 없도록 정책이 변경 되었다.
3. SMTP 서버를 사용하려면 2단계 인증을 지원하지 않는 앱에서도 로그인 할 수 있는 앱 비밀번호(16자리)를 생성해야 한다.

3.gmail 보내기

❖ imap 활성화 (1/2)

gmail에서 우측상단 톱니바퀴를 누르고, [모든 설정 보기] 버튼을 클릭한다.



3.gmail 보내기

❖ imap 활성화 (2/2)

아래 그림처럼 imap을 사용할 수 있도록 체크하고, 변경사항을 저장한다.

설정

기본설정라벨받은편지함계정 및 가져오기필터 및 차단된 주소전달 및 POP/IMAP부가기능채팅 및 Meet고급오프라인테마

IMAP 액세스:
(IMAP를 사용하여 다른 클라이언트에서 Gmail에 액세스)
[자세히 알아보기](#)

상태: IMAP 사용 안함
☒ IMAP 사용
☐ IMAP 사용 안함

IMAP에서 메일을 삭제된 것으로 표시하는 경우:

☒ 자동 삭제 사용 - 서버를 즉시 업데이트(기본값)
☐ 자동 삭제 사용 안함 - 클라이언트가 서버를 업데이트할 때까지 대기

메일이 삭제된 것으로 표시되고 마지막으로 표시된 IMAP 폴더에서 삭제된 경우:

☒ 메일 보관(기본값)
☐ 메일을 휴지통으로 이동
☐ 메일을 즉시 완전삭제

폴더 크기 제한

☒ IMAP 폴더에서 메일 수를 제한하지 않습니다(기본값).
☐ 이만큼의 메일만 포함하도록 IMAP 폴더를 제한합니다. 1,000

이메일 클라이언트 구성(예: Outlook, Thunderbird, iPhone)

[설정 방법](#)

변경사항 저장취소

3.gmail 보내기

❖ 2단계 인증 활성화

구글의 [계정]을 클릭하고 보안 메뉴를 선택한다.

2단계 인증이 사용중지 상태라면 안내에 따라 2단계인증을 진행하도록 한다.

The screenshot shows the Google Account management interface. On the left, the '보안' (Security) menu item is highlighted with a red box. In the main content area, the '2단계 인증' (2-step verification) option is highlighted with a red box, showing a green checkmark and the text '사용 설정일: 3월 3일'. On the right, a dropdown menu is open, showing the '계정' (Account) option at the top, which is also highlighted with a red box and has a red arrow pointing to it from the main menu.

Google 계정

Google 계정 검색

- 홈
- 개인 정보
- 데이터 및 개인 정보 보호
- 보안**
- 사용자 및 공유
- 결제 및 구독
- 정보

Google에 로그인하는 방법

항상 Google 계정에 액세스할 수 있도록 최신 정보가 반영되었는지 확인하세요.

2단계 인증	✓ 사용 설정일: 3월 3일	>
패스키 및 보안 키	패스키 사용 시작	>
비밀번호	최종 변경일: 2020. 1. 11.	>
가능한 경우 비밀번호 건너뛰기	✓ 사용	>









- 계정
- 검색
- 지도
- YouTube
- Gemini
- 뉴스
- Gmail
- Meet
- 채팅

3.gmail 보내기

❖ 앱 비밀번호 생성 (1/2)

앱 비밀번호가 없으면 아래 그림의 화살표를 눌러서 생성한다.

← 2단계 인증

	Google 메시지	 기기 2대	>
	OTP	 OTP 앱 추가	>
	전화번호	 010-9809-4320	>
	복구 코드	 백업 코드 받기	>

앱 비밀번호

앱 비밀번호는 권장되지 않으며 대부분의 경우 필요하지 않습니다. 계정을 안전하게 보호하려면 'Google 계정으로 로그인'을 사용하여 앱을 Google 계정에 연결하세요.

앱 비밀번호

앱 비밀번호 1개

>

3.gmail 보내기

❖ 앱 비밀번호 생성 (2/2)

앱이름을 입력하고, 앱 비밀번호(16자리)를 생성한다.

앱 비밀번호가 없습니다.

To create a new app specific password, type a name for it below...

App name

My-SMTP

만들기

3.gmail 보내기

❖ gmail 프로젝트 생성

의존 라이브러리 추가

❖ 의존 라이브러리 추가 : **build.gradle**

```
dependencies {  
    // jsp  
    implementation 'org.apache.tomcat.embed:tomcat-embed-jasper'  
    // jakarta jstl  
    implementation 'org.glassfish.web:jakarta.servlet.jsp.jstl:3.0.0'  
    // email  
    implementation 'org.springframework.boot:spring-boot-starter-mail:3.0.5'  
    implementation 'org.springframework:spring-context:6.0.7'  
    implementation 'org.springframework:spring-context-support:6.0.7'  
    implementation 'com.sun.mail:jakarta.mail:2.0.1'  
    implementation 'org.springframework.boot:spring-boot-starter-web'  
    compileOnly 'org.projectlombok:lombok'  
    annotationProcessor 'org.projectlombok:lombok'  
    providedRuntime 'org.springframework.boot:spring-boot-starter-tomcat'  
    testImplementation 'org.springframework.boot:spring-boot-starter-test'  
}
```


환경 설정파일 수정

❖ 환경 설정 파일 수정

main / resources / **application.properties** (1/2)

port

server.port=80

prefix and suffix

spring.mvc.view.prefix=/WEB-INF/views/

spring.mvc.view.suffix=.jsp

❖ webapp / **WEB-INF** / **views** 폴더 생성

환경 설정파일 수정

❖ 환경 설정 파일 수정

main / resources / **application.properties** (2/2)

```
# gmail server
spring.mail.host=smtp.gmail.com
spring.mail.port=587
spring.mail.username=giduck23@gmail.com
spring.mail.password=2단계 인증비번(16자리)
spring.mail.properties.mail.debug=true
spring.mail.properties.mail.smtp.connectiontimeout=5000
spring.mail.properties.mail.smtp.starttls.enable=true
spring.mail.properties.mail.smtp.starttls.required=true
spring.mail.properties.mail.smtp.auth=true
```

3.gmail 보내기

❖ 메일 전송 : mailform.jsp

E-Mail 보내기

보내는사람	<input type="text" value="google EMail주소"/>
받는사람	<input type="text" value="받는사람 EMail주소"/>
제목	<input type="text"/>
내용	<div><div></div></div>
<div>전송</div>	

3.gmail 보내기

❖ DTO : Mail.java

```
package com.example.demo.model;
```

```
import lombok.Data;
```

```
@Data
```

```
public class Mail {
```

```
    private String sendmail;
```

```
    private String receivedmail;
```

```
    private String subject;
```

```
    private String content;
```

```
}
```

3.gmail 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";
```

```
    }
```

3.gmail 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";
}
```