1. 개요

이메일을 발송하기 위해서는 메일 서버가 필요하다. 메일 서버는 아래와 같은 두 가지 방법으로 사용이 가능한데, 이번에는 두 번째 방법인 **Gmail의 SMTP 서버를 활용하여 이메일을 발송하는 방법**에 대해 알아보도록 하자.

- 메일 서버 직접 구축
- 구글, 네이버 등 포털 사이트에서 제공하는 SMTP 서버 이용

2. 구글 계정 설정

Gmail의 SMTP 서버를 활용하기 위해서는 계정 설정이 필요하다.

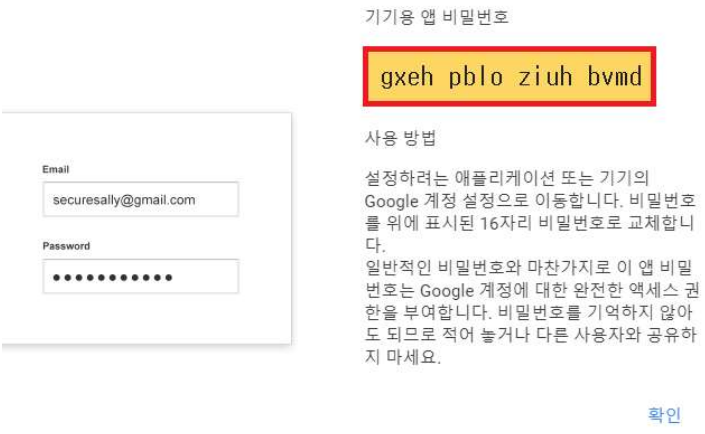
1) 구글 계정 관리 - 보안 탭 - 앱 비밀번호 클릭

(보안 탭의 2단계 인증까지 완료된 상태에서 진행해야 한다.)

2) 앱 : 메일 / 기기 : 기타 (맞춤 이름) 선택하고 기기 이름ex) `Mail for test`을 입력 후 생성 클릭

3) 아래와 같이 16자리의 앱 비밀번호가 생성된 것을 확인할 수 있다.

(앱 비밀번호가 유출되면 다른 사용자가 본인의 계정으로 메일에 접근할 수 있으므로 주의해야 한다.)



3. 개발 환경

- Java 11

- Spring Boot 2.5.3

4. 공통설정

4-1. build.gradle

```
dependencies {
    ...
    implementation 'org.springframework.boot:spring-boot-starter-mail'
    ...
}
```

4-2. application.yml

```
spring:
  mail:
    host: smtp.gmail.com # SMTP 서버 호스트
    port: 587 # SMTP 서버 포트
    username: tyjk32 # SMTP 서버 로그인 아이디 (발신자)
    password: vrfoenmcfqhqchvl # SMTP 서버 로그인 패스워드 (앱 비밀번호)
    properties:
      mail:
        smtp:
          auth: true # 사용자 인증 시도 여부 (기본값 : false)
          timeout: 5000 # Socket Read Timeout 시간(ms) (기본값 : 무한대)
          starttls:
            enable: true # StartTLS 활성화 여부 (기본값 : false)
```

5. 텍스트로 구성된 이메일 발송

[EmailMessage.java]

```
@Data
@Builder
public class EmailMessage {

    private String to;

    private String subject;

    private String message;
}
```

[EmailService.java]

```
@Slf4j
@Service
@RequiredArgsConstructor
public class EmailService {

    private final JavaMailSender javaMailSender;

    public void sendMail(EmailMessage emailMessage) {
        MimeMessage mimeMessage = javaMailSender.createMimeMessage();
        try {
            MimeMessageHelper mimeMessageHelper = new MimeMessageHelper(mimeMessage, false, "UTF-8");
            mimeMessageHelper.setTo(emailMessage.getTo()); // 메일 수신자
            mimeMessageHelper.setSubject(emailMessage.getSubject()); // 메일 제목
            mimeMessageHelper.setText(emailMessage.getMessage(), false); // 메일 본문 내용, HTML 여부
            javaMailSender.send(mimeMessage);
            log.info("Success!!");
        } catch (MessagingException e) {
            log.info("fail!!");
            throw new RuntimeException(e);
        }
    }
}
```

MimeMessageHelper(MimeMessage mimeMessage, boolean multipart, @Nullable String encoding)	
이메일을 위한 포맷인 Mime 타입의 메시지 생성을 도와주는 객체	
MimeMessage mimeMessage	Mime 메시지
boolean multipart	첨부파일 전송을 위한 multipart 여부
@Nullable String encoding	인코딩 방식

[EmailController.java]

```
@RestController
@RequiredArgsConstructor
public class EmailController {

    private final EmailService emailService;

    @PostMapping("/send-mail")
    public ResponseEntity sendMail() {
        EmailMessage emailMessage = EmailMessage.builder()
                .to("tyjk32@gmail.com")
                .subject("테스트 메일 제목")
                .message("테스트 메일 본문")
                .build();
        emailService.sendMail(emailMessage);
        return new ResponseEntity(HttpStatus.OK);
    }
}
```

[테스트 결과]

6. HTML로 구성된 이메일 발송

[EmailMessage.java]

```java
@Data
@Builder
public class EmailMessage {

    private String to;

    private String subject;

    private String message;
}
```

[EmailService.java]

```java
@Slf4j
@Service
@RequiredArgsConstructor
public class EmailService {

    private final JavaMailSender javaMailSender;

    public void sendMail(EmailMessage emailMessage) {
        MimeMessage mimeMessage = javaMailSender.createMimeMessage();
        try {
            MimeMessageHelper mimeMessageHelper = new MimeMessageHelper(mimeMessage, false, "UTF-8");
            mimeMessageHelper.setTo(emailMessage.getTo()); // 메일 수신자
            mimeMessageHelper.setSubject(emailMessage.getSubject()); // 메일 제목
            mimeMessageHelper.setText(emailMessage.getMessage(), true); // 메일 본문 내용, HTML 여부
            javaMailSender.send(mimeMessage);
            log.info("Success!!");
        } catch (MessagingException e) {
            log.info("fail!!");
            throw new RuntimeException(e);
        }
    }
}
```

Line 14 : 두 번째 인자인 HTML 여부를 false로 주면 HTML을 인식하지 못하고 텍스트로 발송

MimeMessageHelper(MimeMessage mimeMessage, boolean multipart, @Nullable String encoding)	
이메일을 위한 포맷인 Mime 타입의 메시지 생성을 도와주는 객체	
MimeMessage mimeMessage	Mime 메시지
boolean multipart	첨부파일 전송을 위한 multipart 여부
@Nullable String encoding	인코딩 방식

[EmailController.java]

```java
@RestController
@RequiredArgsConstructor
public class EmailController {

    private final EmailService emailService;

    @PostMapping("/send-mail")
    public ResponseEntity sendMail() {
        EmailMessage emailMessage = EmailMessage.builder()
                .to("tyjk32@gmail.com")
                .subject("테스트 메일 제목")
                .message("<html><head></head><body><div style=\"background-color: gray;\">테스트 메일 본문<div></body></html>")
                .build();
        emailService.sendMail(emailMessage);
        return new ResponseEntity(HttpStatus.OK);
    }
}
```

[테스트 결과 - HTML여부를 true로 설정]

HTML이 정상 반영된 것을 확인할 수 있다.

[테스트 결과 - HTML여부를 false로 설정]

HTML 태그를 인식하지 못하고 텍스트로 표시된 것을 확인할 수 있다.

7. Template Engine을 이용한 이메일 발송

Template Engine(템플릿 엔진)이란 템플릿과 데이터를 이용하여 동적으로 컨텐츠를 생성하는 방법을 말한다. 스프링 부트가 자동설정을 지원하는 Template Engine은 `FreeMarker, Groovy, Thymeleaf, Mustache` 4가지가 있는데, 이 중에서 **Thymeleaf를 이용한 이메일 발송 방법**을 알아보도록 하자.

[build.gradle]

```
dependencies {
    ...
    implementation 'org.springframework.boot:spring-boot-starter-thymeleaf'
    ...
}
```

[mail-sample.html]

```
<!DOCTYPE html>
<html lang="en" xmlns:th="http://www.thymeleaf.org">
<head>
  <meta charset="UTF-8">
  <title>메일 테스트</title>
</head>
<body>
<div>
  <p>안녕하세요, <span th:text="${nickname}"></span>님</p>
  <p>메일 테스트입니다.</p>
</div>
</body>
</html>
```

[EmailMessage.java]

```
@Data
@Builder
public class EmailMessage {

    private String to;

    private String subject;

    private String message;
}
```

[EmailService.java]

```
@Slf4j
@Service
@RequiredArgsConstructor
public class EmailService {

    private final JavaMailSender javaMailSender;

    public void sendMail(EmailMessage emailMessage) {
        MimeMessage mimeMessage = javaMailSender.createMimeMessage();
        try {
            MimeMessageHelper mimeMessageHelper = new MimeMessageHelper(mimeMessage, false, "UTF-8");
            mimeMessageHelper.setTo(emailMessage.getTo()); // 메일 수신자
            mimeMessageHelper.setSubject(emailMessage.getSubject()); // 메일 제목
            mimeMessageHelper.setText(emailMessage.getMessage(), true); // 메일 본문 내용, HTML 여부
            javaMailSender.send(mimeMessage);
            log.info("Success!!");
        } catch (MessagingException e) {
            log.info("fail!!");
            throw new RuntimeException(e);
        }
    }
}
```

Line 14 : 두 번째 인자인 HTML 여부를 false로 주면 HTML을 인식하지 못하고 텍스트로 발송

MimeMessageHelper(MimeMessage mimeMessage, boolean multipart, @Nullable String encoding)	
이메일을 위한 포맷인 Mime 타입의 메시지 생성을 도와주는 객체	
MimeMessage mimeMessage	Mime 메시지
boolean multipart	첨부파일 전송을 위한 multipart 여부
@Nullable String encoding	인코딩 방식

[EmailController.java]

```
@RestController
@RequiredArgsConstructor
public class EmailController {

    private final EmailService emailService;
    private final TemplateEngine templateEngine;

    @PostMapping("/send-mail")
```

```
9       public ResponseEntity sendMail() {
10          Context context = new Context();
11          context.setVariable("nickname", "Aiden");
12          String message = templateEngine.process("mail/mail-sample", context);
13
14          EmailMessage emailMessage = EmailMessage.builder()
15                  .to("tyjk32@gmail.com")
16                  .subject("테스트 메일 제목")
17                  .message(message)
18                  .build();
19          emailService.sendMail(emailMessage);
20          return new ResponseEntity(HttpStatus.OK);
21      }
22  }
```

Line 11 : mail-sample.html의 nickname으로 넘겨줄 값 세팅

[테스트 결과 - HTML여부를 true로 설정]

[테스트 결과 - HTML여부를 false로 설정]