



ZIP 포팅 메뉴얼

A407여섯싸둥이

목차

1. 개발환경 및 외부서비스	3
1.1. 프로젝트 개요	3
1.2. 협업 툴 / 환경	3
1.3. 개발환경	3
1.4. 외부 서비스	5
1.4.1. Amazon Web Service S3 (Simple Storage Service)	5
1.4.2. Google Gmail SMTP	8
1.4.3. Firebase Messaging Service	10
1.4.4. Photon	14
2) 환경변수 설정 및 빌드	17
2.1. 환경변수 설정	17
2.1.1. Spring – application.properties	17
2.1.2. Android	19
2.2. 빌드	19
2.2.1. Ubuntu Server- Backend 구축	19
2.2.2. Android & Unity 빌드	20

1. 개발환경 및 외부서비스

1.1. 프로젝트 개요

사랑하는 가족과 매일 주어지는 미션을 통해 소통하고, 메타버스를 결합해 소통을 즐겁게 만드는
게이미피케이션 + 메타버스 결합 가족 소통 애플리케이션

1.2. 협업 툴 / 환경

- GitLab
 - Git-Flow. develop, FE_develop, BE_develop 브랜치를 나누고
 - 각 feature별 브랜치를 따서 작업 진행
 - MR시 Front/Back Maintainer가 확인 후 합병
- Notion
 - 회의가 있을때마다 회의록을 기록하여 보관
 - 컨벤션 정리
 - api 문서 관리 등
- JIRA
 - 매주 월요일 목표량을 설정하여 Sprint 진행
 - 업무별 Story Point(1~4)를 설정, In-Progress -> Done 순으로 작업
- MatterMost
 - Gitlab, JIRA 봇 연동하여 실시간으로 협업
 - Server 연동하여 실시간 에러 처리
- Webex
 - 회의 : 평일 아침 Webex에서 데일리 스크럼 진행
 - 문제점이 생겼을 때 팀원들에게 직접 소통

1.3. 개발환경

- **Backend - Spring**
 - IntelliJ IDEA : 2021.3.1

- IntelliJ Runtime: 11.0.13+7-b1751.21 amd64
- JDK : 17-ea
- JRE : build 17-ea+14
- JVM : build 17-ea+14, mixed mode, sharing
- DB : 8.0.30-MySQL
- Springboot : 2.7.3
- Gradle : 7.5
- **Frontend – Android Studio**
 - Android Studio Dolphin | 2021.3.1 Patch 1
 - ART : 11.0.13+0-b1751.21-8125866 amd64
 - VM: OpenJDK 64-Bit Server VM by JetBrains s.r.o.
 - SDK version : 32 (min : 26, target : 32)
 - NDK version : 21.3.6528147
 - SDK Build Tools : 30.0.3
- **Unity : 2021.3.9.f1**
- **CI/CD**
 - Server : AWS EC2 Ubuntu 20.04 LTS
 - Docker : 20.10.20
 - nginx : 1.18.0
 - Jenkins : 2.346.2

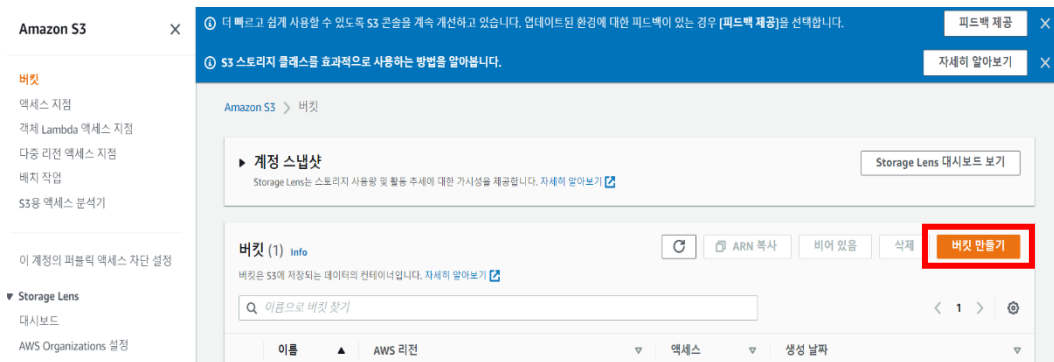
1.4. 외부 서비스

사용된 외부 서비스 : Amazon S3, Google Gmail SMTP, Firebase Service, Photon

1.4.1. Amazon Web Service S3 (Simple Storage Service)

- 1) AWS 가입하기
- 2) S3 버킷 생성

A. 버킷 만들기 선택



B. 버킷 이름 작성 및 AWS리전 서울 선택

버킷 이름

noning

버킷 이름은 전역에서 고유해야 하며 공백 또는 대문자를 포함할 수 없습니다. [버킷 이름 지정 규칙 참조](#)

AWS 리전

아시아 태평양(서울) ap-northeast-2

기존 버킷에서 설정 복사 - 선택 사항

다음 구성의 버킷 설정만 복사됩니다.

버킷 선택

C. 퍼블릭 액세스 차단 풀어주기

이 버킷의 퍼블릭 액세스 차단 설정

퍼블릭 액세스는 ACL(엑세스 제어 목록), 버킷 정책, 액세스 지점 정책 또는 모두를 통해 버킷 및 객체에 부여됩니다. 이 버킷 및 해당 객체에 대한 퍼블릭 액세스가 차단되었는지 확인하려면 모든 퍼블릭 액세스 차단을 활성화합니다. 이 설정은 이 버킷 및 해당 액세스 지점에만 적용됩니다. AWS에서는 모든 퍼블릭 액세스 차단을 활성화하도록 권장하지만, 이 설정을 적용하기 전에 퍼블릭 액세스가 없어도 애플리케이션이 올바르게 작동하는지 확인합니다. 이 버킷 또는 내부 객체에 대한 어느 정도 수준의 퍼블릭 액세스가 필요한 경우 특정 스토리지 사용 사례에 맞게 아래 개별 설정을 사용자 지정할 수 있습니다. [자세히 알아보기](#)

☐ 모든 퍼블릭 액세스 차단

이 설정을 활성화하면 아래 4개의 설정을 모두 활성화한 것과 같습니다. 다음 설정 각각은 서로 독립적입니다.

☐ 새 ACL(엑세스 제어 목록)을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단

S3은 새로 추가된 버킷 또는 객체에 적용되는 퍼블릭 액세스 권한을 차단하며, 기존 버킷 및 객체에 대한 새 퍼블릭 액세스 ACL 생성을 금지합니다. 이 설정은 ACL을 사용하여 S3 리소스에 대한 퍼블릭 액세스를 허용하는 기존 권한을 변경하지 않습니다.

☐ 임의의 ACL(엑세스 제어 목록)을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단

S3은 버킷 및 객체에 대한 퍼블릭 액세스를 부여하는 모든 ACL을 무시합니다.

☐ 새 퍼블릭 버킷 또는 액세스 지점 정책을 통해 부여된 버킷 및 객체에 대한 퍼블릭 액세스 차단

S3은 버킷 및 객체에 대한 퍼블릭 액세스를 부여하는 새 버킷 및 액세스 지점 정책을 차단합니다. 이 설정은 S3 리소스에 대한 퍼블릭 액세스를 허용하는 기존 정책을 변경하지 않습니다.

☐ 임의의 퍼블릭 버킷 또는 액세스 지점 정책을 통해 부여된 버킷 및 객체에 대한 퍼블릭 및 교차 계정 액세스 차단

S3은 버킷 및 객체에 대한 퍼블릭 액세스를 부여하는 정책을 사용하는 버킷 또는 액세스 지점에 대한 퍼블릭 및 교차 계정 액세스를 무시합니다.

D. 버킷 만들기

E. 버킷 – 권한에서 버킷 정책 설정

버킷 정책

JSON으로 작성된 버킷 정책은 버킷에 저장된 객체에 대한 액세스 권한을 제공합니다. 버킷 정책은 다른 계정이 소유한 객체에는 적용되지 않습니다. [자세히 알아보기](#)

[편집](#)[삭제](#)

```
{
  "Version": "2012-10-17",
  "Id": "Policy1659851181014",
  "Statement": [
    {
      "Sid": "Stmt1659851171151",
      "Effect": "Allow",
      "Principal": "*",
      "Action": [
        "s3:DeleteObject",
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::noning/*"
    }
  ]
}
```

[복사](#)

3) IAM 추가

사용자 추가

1 2 3 4 5

사용자 세부 정보 설정

동일한 액세스 유형 및 권한을 사용하여 한 번에 여러 사용자를 추가할 수 있습니다. [자세히 알아보기](#)

사용자 이름*

이름이 "noning"인 사용자가 이미 존재합니다.

[+ 다른 사용자 추가](#)

AWS 액세스 유형 선택

이러한 사용자가 주로 AWS에 액세스하는 방법을 선택합니다. 프로그래밍 방식의 액세스만 선택하면 사용자가 위임된 역할을 사용하여 콘솔에 액세스하는 것을 방지할 수 없습니다. 액세스 키와 자동 생성된 암호가 마지막 단계에서 제공됩니다. [자세히 알아보기](#)


- AWS 자격 증명 유형 선택***
- ☒ **액세스 키 – 프로그래밍 방식 액세스**
AWS API, CLI, SDK 및 기타 개발 도구에 대해 **액세스 키 ID** 및 **비밀 액세스 키** 을(를) 할
성화합니다.
 - ☐ **암호 – AWS 관리 콘솔 액세스**
사용자가 AWS Management Console에 로그인할 수 있도록 허용하는 **비밀번호** 을(를)
활성화합니다.


A. IAM 사용자 추가 – 액세스 키 체크


사용자 추가

1 2 3 4 5


▼ 권한 설정

 그룹에 사용자 추가

 기존 사용자에서 권한 복
사


 기존 정책 직접 연결

정책 생성



정책 필터 ▼

1 결과 표시

	정책 이름 ▼	유형	사용 용도
<input type="checkbox"/>	 AmazonS3FullAccess	AWS 관리형	Permissions policy (1)

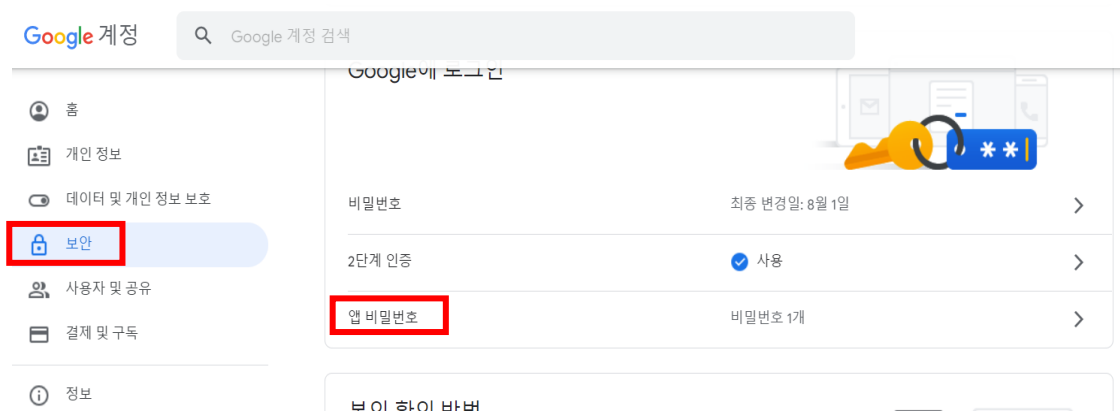
B. S3FullAccess 정책 연결

C. 생성된 사용자 csv파일 받은 후 Access Key, Secret key 확인

1.4.2. Google Gmail SMTP

1) Google 메일 앱 비밀번호 발급

A. Google 계정 설정 - 2단계 인증 설정



B. Google 계정 설정 - 보안 - 앱 비밀번호

← 앱 비밀번호

앱 비밀번호를 사용하면 2단계 인증을 지원하지 않는 기기의 앱에서 Google 계정에 로그인할 수 있습니다. 비밀번호를 한 번만 입력하면 기억할 필요가 없습니다. [자세히 알아보기](#)

앱 비밀번호

이름	생성됨	최종 사용일
Windows 컴퓨터의 메일	8월 2일	오후 1:42

앱 비밀번호를 생성할 앱 및 기기를 선택하세요.

메일 ▼ Windows 컴퓨터 ▼

생성

C. 메일 , Windows 컴퓨터 비밀번호 생성

D. Windows 컴퓨터용 앱 비밀번호 확인

2) POP/IMAP 설정

A. Gmail – 모든 설정

설정

기본설정 라벨 받은편지함 계정 및 가져오기 필터 및 차단된 주소 **전달 및 POP/IMAP** 부가기능 채팅 및 Meet 고급 오프라인 테마

전달:
[자세히 알아보기](#)

도움말: [필터를 만들면](#) 메일 중 일부만 전달할 수도 있습니다.

POP 다운로드:
[자세히 알아보기](#)

1. 상태: 모든 메일에 대해 **POP가 사용 설정되어 있습니다.**

- ☒ 이미 다운로드된 메일을 포함하여 **모든 메일에 POP를 활성화 하기**
- ☐ 지금부터 수신되는 메일에만 POP를 사용하기
- ☐ POP 사용 안함

2. POP로 메시지를 여는 경우 ▼

3. 이메일 클라이언트 구성 (예: Outlook, Eudora, Netscape Mail)
[설정 방법](#)

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

상태: IMAP를 사용할 수 있습니다.

- ☒ **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

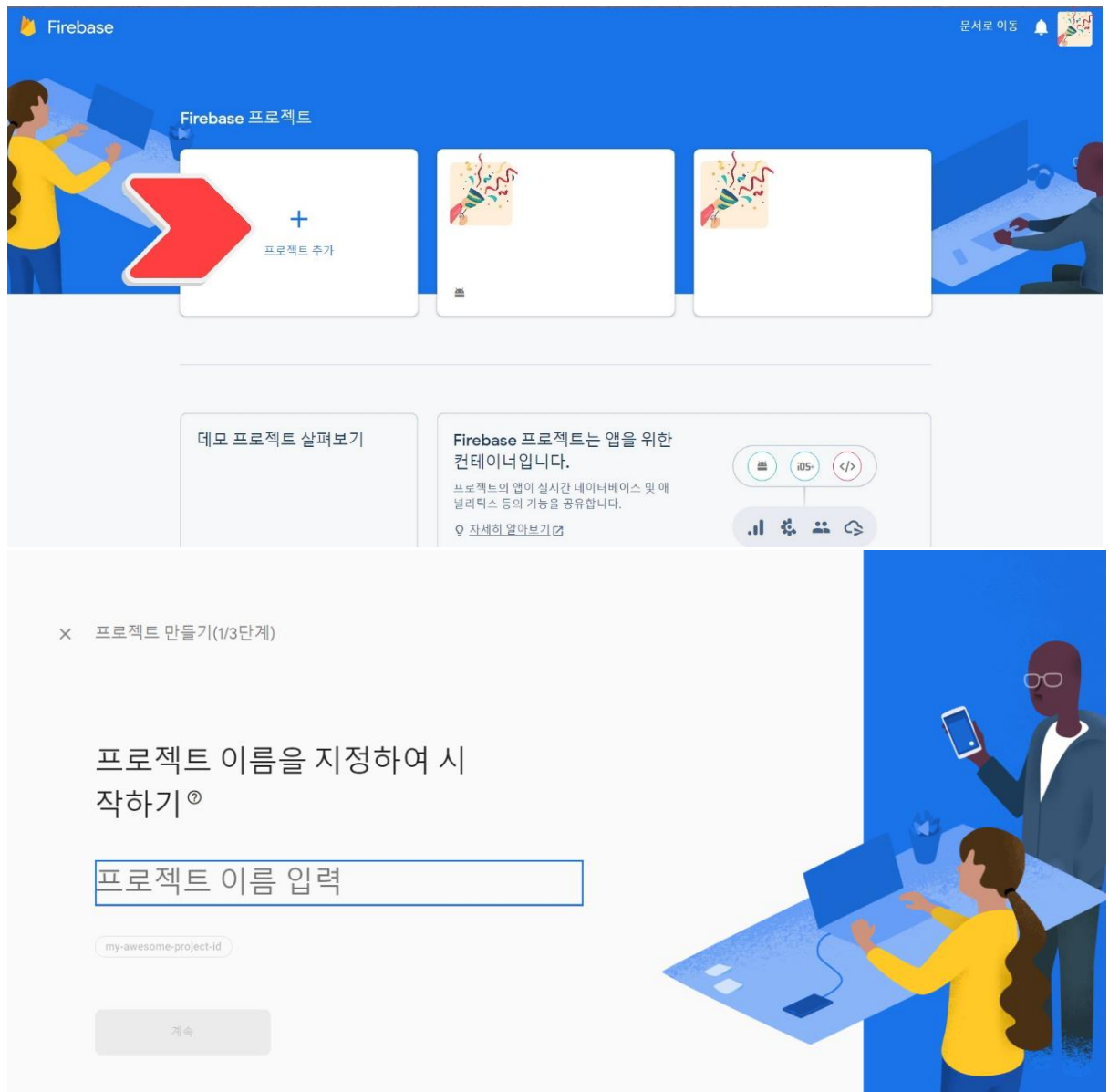
메일이 삭제된 것으로 표시되고 마지막으로 표시된 IMAP 폴더에서 삭제된 경우:

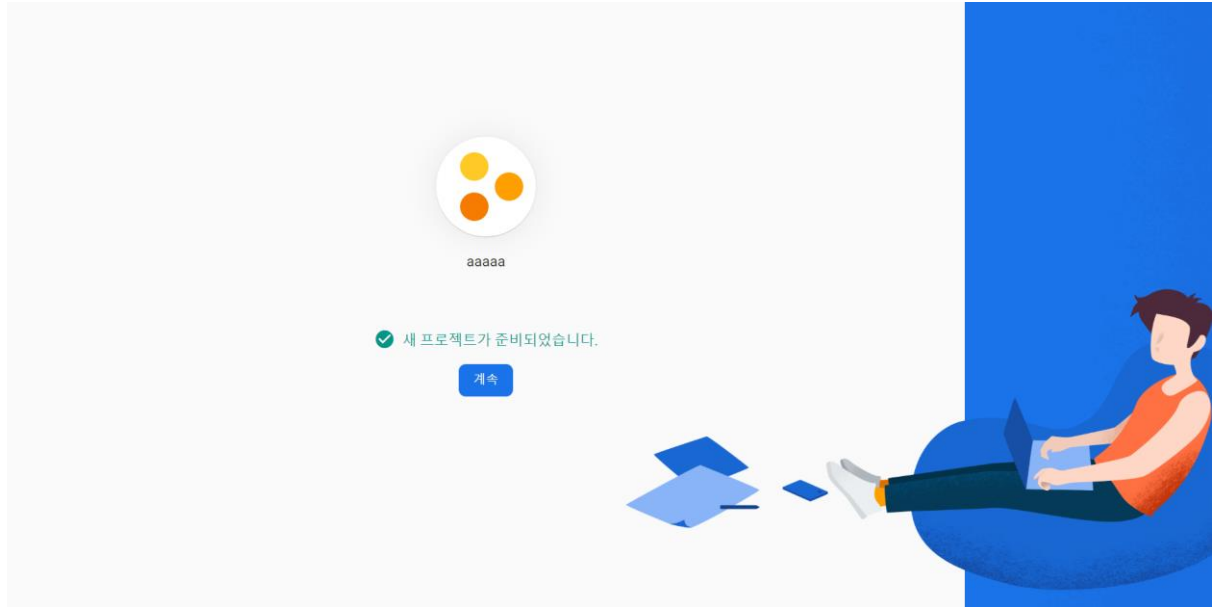
- ☒ 메일 보관(기본값)
- ☐ 메일을 휴지통으로 이동
- ☐ 메일을 즉시 완전삭제

B. 전달 및 POP/IMAP 설정

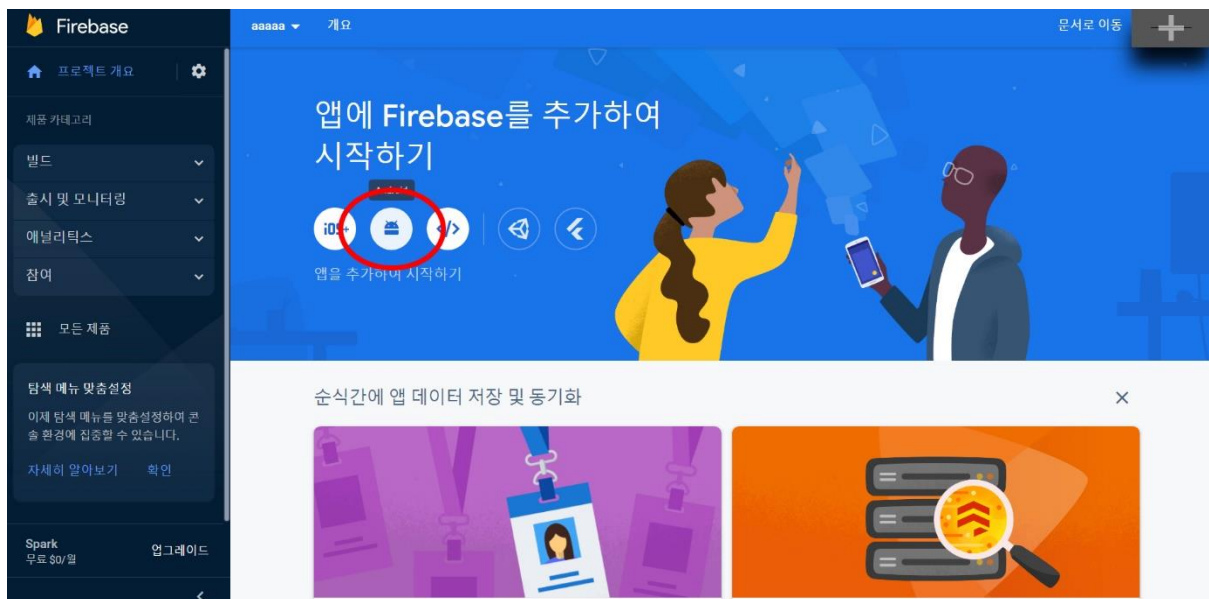
1.4.3. Firebase Messaging Service

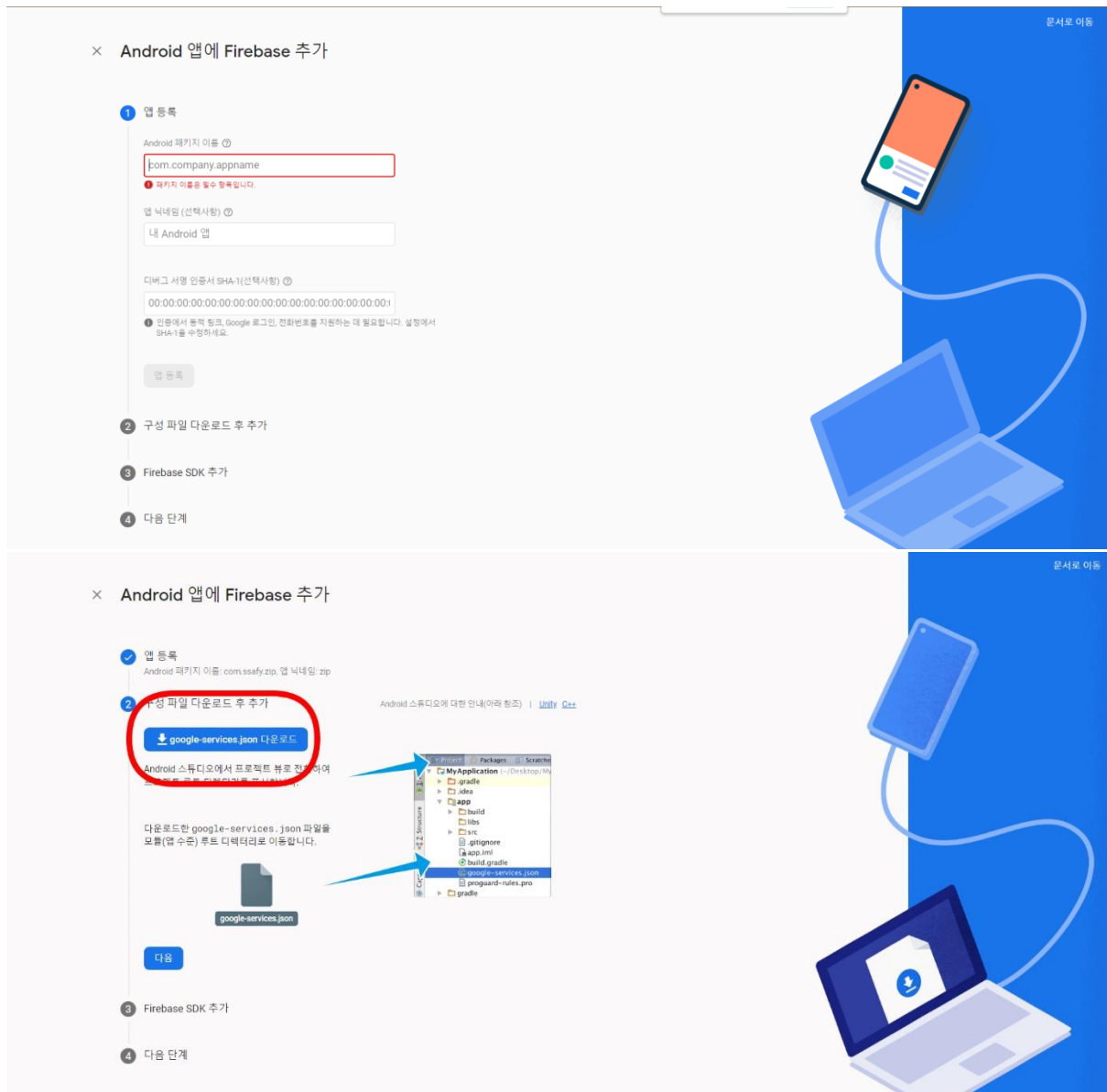
- 1) <https://console.firebase.google.com/>에서 프로젝트 추가





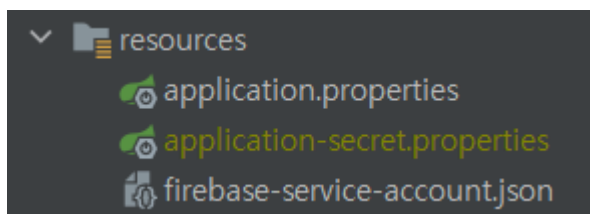
2) 프로젝트에 앱 추가 및 설정파일 다운로드



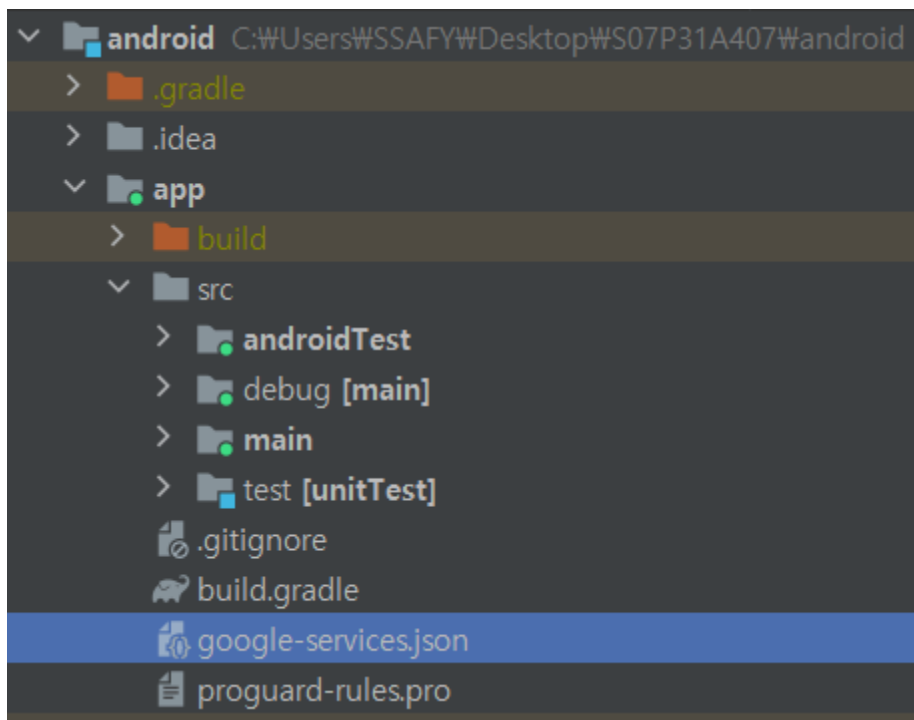


3) 다운로드 받은 google-service.json 파일 복사 및 붙여넣기

- Spring resources 폴더



- Andriod 프로젝트 내 위치



1.4.4. Photon

1) Photon(<https://www.photonengine.com>) 가입 및 Voice App 만들기

The image displays two screenshots of the Photon Engine website. The top screenshot shows the 'Your Photon Cloud Apps' dashboard. The 'CREATE A NEW APP' button is circled in red, with a red arrow pointing to it from the text '2. 새로운 앱 만들기 클릭' (Click to create a new app). The 'Dashboard' link in the top navigation bar is also circled in red, with a red arrow pointing to it from the text '1. 대쉬보드 탭으로 이동' (Move to the dashboard tab). The bottom screenshot shows the 'Create a New Application' form. The 'Photon Type' dropdown is set to 'Voice' (labeled '사용할 Photon Type(Voice로)'). The 'Name' field contains '어플리케이션 이름' (Application name). The 'CREATE' button at the bottom is circled in red, with a red arrow pointing to it from the text '상단의 정보 기입 후 CREATE 클릭' (Click CREATE after entering the information above).

1. 대쉬보드 탭으로 이동

2. 새로운 앱 만들기 클릭

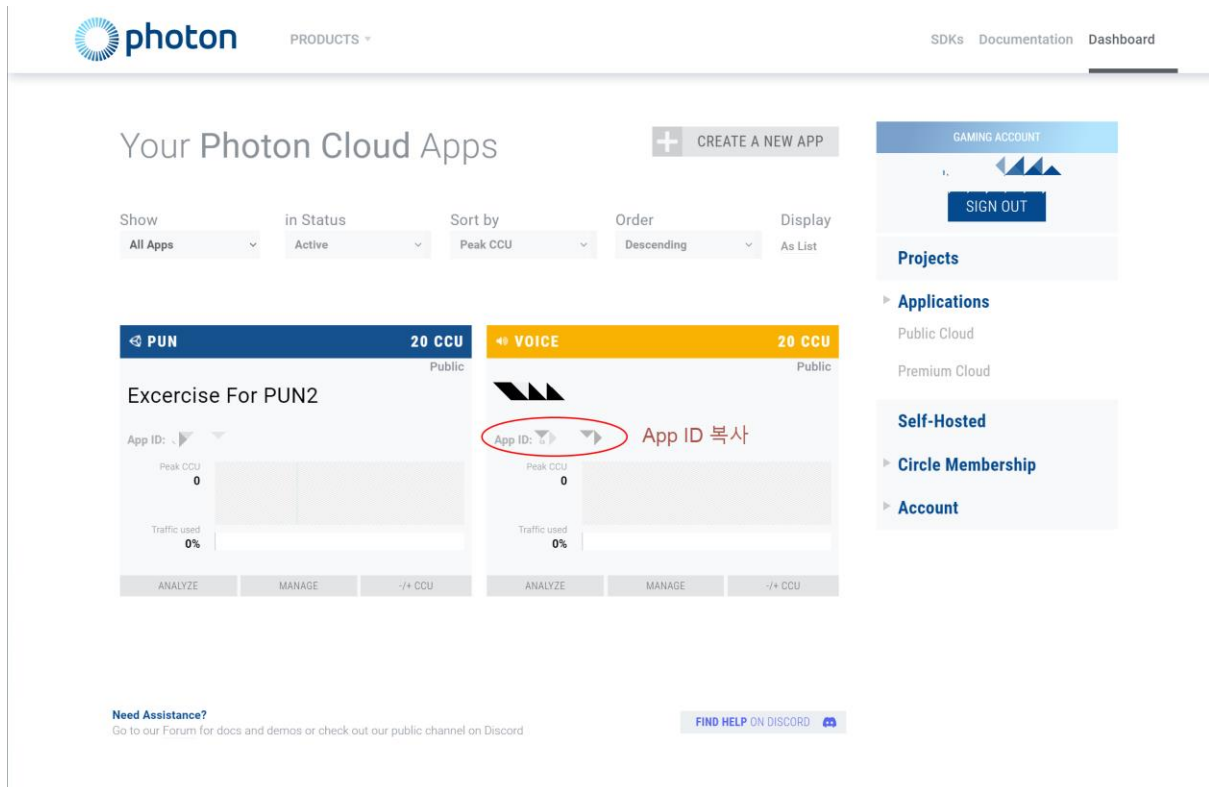
CREATE A NEW APP

Photon Type * 사용할 Photon Type(Voice로)

Name * 어플리케이션 이름

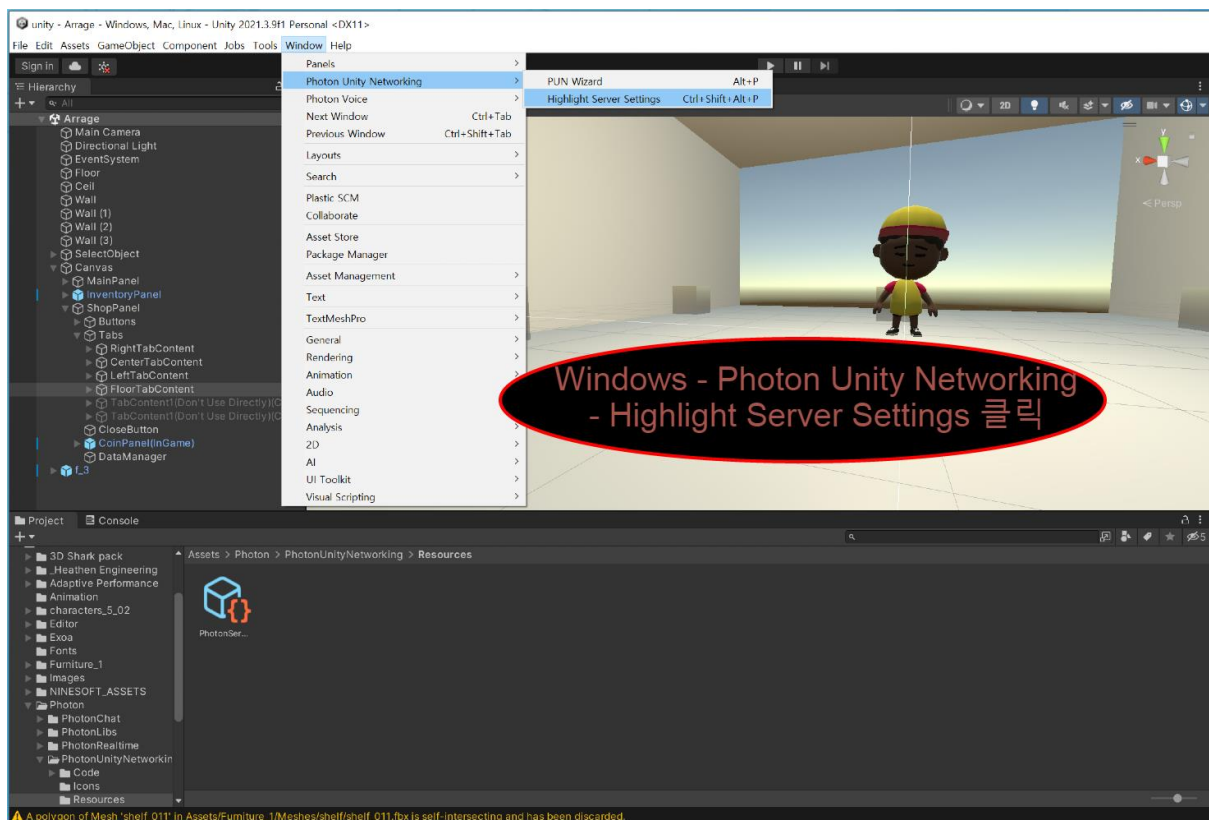
CREATE

상단의 정보 기입 후 CREATE 클릭



2) 같은 방식으로 PUN APP 만들기

3) Photon Voice 및 PUN APP ID를 유니티 설정에 삽입



Inspector

Photon Server Settings (Server Settings)

Open

Version:

Pun: 2.41 Photon lib: 4.1.6.19

▼ Server/Cloud Settings

App Id PUN

PUN App ID 입력

Dashboard

App Id Chat

Dashboard

App Id Voice

Voice App ID 입력

Dashboard

App Version

Use Name Server

☒

Fixed Region

KR

Server

Port

0

Proxy Server

Protocol

TCP로 설정

Protocol Fallback

☒

Lobby Statistics

☐

Network Logging

ERROR

PUN Logging

Errors Only

Support Logger

☐

Run In Background

☒

Start In Offline Mode

☐

Dev Region

Best Region Preference: n/a

Reset

Edit WhiteList

▶ RPCs

2) 환경변수 설정 및 빌드

2.1. 환경변수 설정

2.1.1. Spring – application.properties

#MySQL Settings

spring.datasource.url=jdbc:mysql://\${ DB source URL }

spring.datasource.username=\${ DB username }

spring.datasource.password=\${ DB password }

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

spring.jpa.database-platform=org.hibernate.dialect.MySQL5Dialect

#Redis Settings

spring.redis.host = 호스트 주소

spring.redis.port = 포트 번호

spring.redis.password= 레디스 암호

JWT

JWT.SECRET : 시크릿 키

#SMTP

spring.mail.host : 주소 (구글 SMTP : smtp.gmail.com)

spring.mail.port : 포트 (구글 SMTP : 587)

spring.mail.username : 구글 ID

spring.mail.password : 메일 접속 비밀번호

spring.mail.properties.mail.smtp.auth : auth 명령 사용여부 (true)

spring.mail.properties.mail.smtp.starttls.enable : TLS-protection 사용여부 (true)

#WEBHOOK

notification.mattermost.enabled : 사용여부 (true)

notification.mattermost.webhook-url : Webhook 주소

notification.mattermost.pretext : attachment의 상단에 나오는 텍스트

#S3 Bucket

cloud.aws.credentials.accessKey : 버킷 접근키

cloud.aws.credentials.secretKey : 버킷 비밀키

cloud.aws.stack.auto : 스택이름 자동 감지여부 (false)

cloud.aws.s3.bucket : 버킷 이름

cloud.aws.region.static : 버킷 지역 (ap-northeast-2)

#multipart 사이즈 설정

spring.servlet.multipart.max-file-size=20MB

spring.servlet.multipart.max-request-size=20MB

#기타 설정

spring.mvc.pathmatch.matching-strategy=*ant_path_matcher*

spring.jpa.hibernate.ddl-auto=\${스프링 설정에 맞는 DB-schema 관리, default=none, 테스트 용으로 자동 생성 위해 create}

2.1.2. Android

#Server URL 설정(com.ssafy.zip.android.ApiService)

```
private const val BASE_URL = "${Server URL}"
```

사용자 변수 편집 ✕

변수 이름(N):	<input type="text" value="ANDROID_HOME"/>
변수 값(V):	<input type="text" value="C:\Users\사용자명\AppData\Local\AndroidSdk"/>

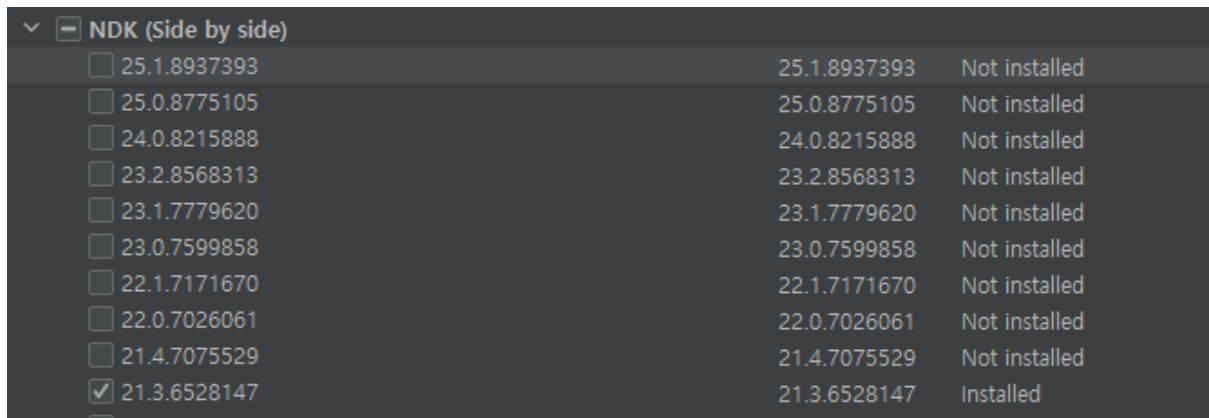
2.2. 빌드

2.2.1. Ubuntu Server- Backend 구축

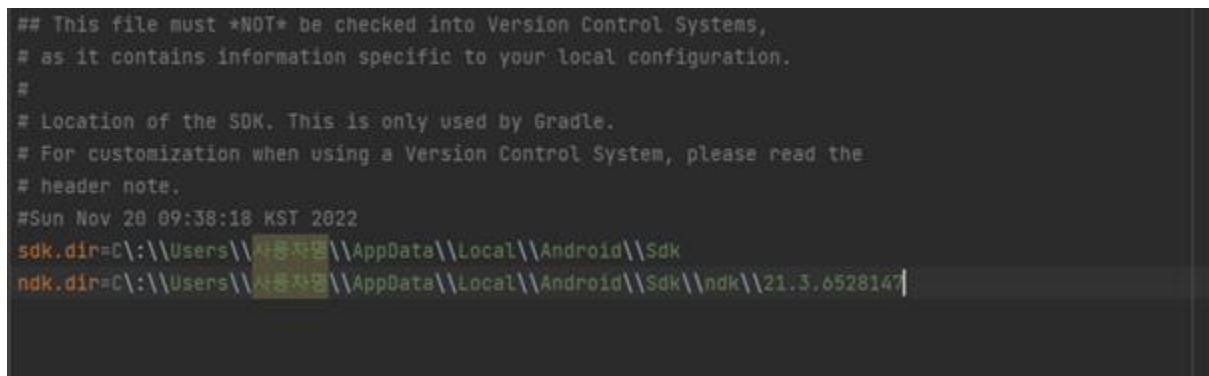
```
// MySQL DB 구축
sudo apt-get update
sudo apt-get install mysql-server
// SQL 접속해 스키마 만들기
// Docker 설치
sudo apt-get update
sudo apt-get install \
    ca-certificates \
    curl \
    gnupg \
    lsb-release
sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/etc/apt/keyrings/docker.gpg
echo \
    "deb [arch=$(dpkg --print-architecture) signed-
by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \
    $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list >
/dev/null
sudo apt-get update
apt-cache madison docker-ce | awk '5:20.10.20~3-0~ubuntu-jammy'
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-
plugin
// Docker 에 레디스 컨테이너 구축
docker search redis
docker pull redis
docker run -p 6379:${output_port_num} --name redis -d redis:latest --
requirepass "${password}"
// Docker 에 Spring Server 컨테이너 구축
sudo apt-get install git
git clone https://lab.ssafy.com/s07-final/S07P31A407.git
cd ./backend
docker build -t zip:latest .
docker run --name zip_back -d -p 8888:8888 zip:latest
```

2.2.2. Android & Unity 빌드

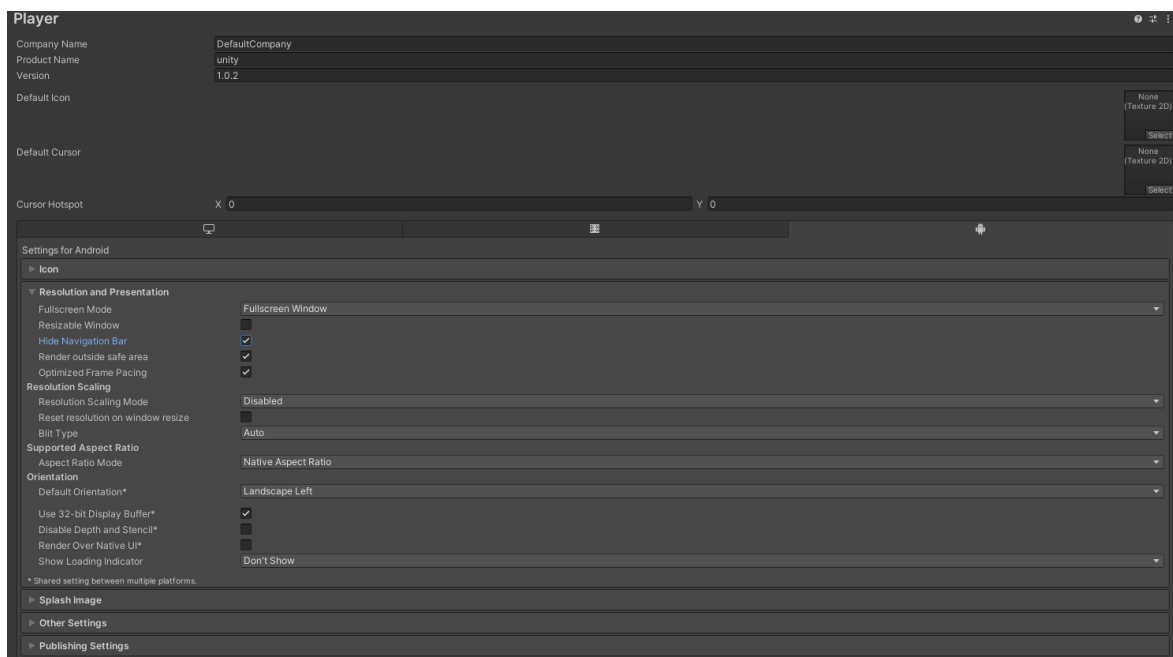
■ SDK MANAGER에서 NDK 21.3.6528147 설치



■ local.properties에서 ndk 경로 설정(git에서 pull했을 경우 여기까지만 진행)



■ Unity Project Settings(Player)



Other Settings

Rendering

Color Space*

Linear

Auto Graphics API

✓

Require ES3.1

✓

Require ES3.1+AEP

✓

Require ES3.2

✓

Color Gamut*

SRGB

Multithreaded Rendering*

✓

Static Batching

✓

Dynamic Batching

✓

Compute Skinning*

✓

Graphics Jobs (Experimental)

✓

Texture compression format

ASTC

Normal Map Encoding

XYZ

Lightmap Encoding

Normal Quality

Lightmap Streaming

✓

Streaming Priority

0

Frame Timing Stats

✓

OpenGL: Profiler GPU Recorders

✓

On OpenGL, Profiler GPU Recorders may disable the GPU Profiler.

Virtual Texturing*

✓

Shader precision model*

Use platform defaults for sampler precision

360 Stereo Capture*

✓

Vulkan Settings

SRGB Write Mode*

✓

Number of swapchain buffers*

3

Acquire swapchain image late as possible*

✓

Recycle command buffers*

✓

Apply display rotation during rendering

✓

Identification

Override Default Package Name

com.DefaultCompany.unity

Package Name

1.0.2

Version*

1

Bundle Version Code

Android 8.0 'Oreo' (API level 26)

Minimum API Level

API level 32

Target API Level

Configuration

Scripting Backend

IL2CPP

Api Compatibility Level*

.NET Standard 2.1

C++ Compiler Configuration

Release

Use incremental GC

✓

Assembly Version Validation (editor only)

✓

Mute Other Audio Sources*

✓

Target Architectures

ARMv7

✓

ARM64

✓

x86 (Chrome OS)

✓

x86-64 (Chrome OS)

✓

Split APKs is disabled when building Android AppBundle

Split APKs by target architecture (Experimental)

✓

Target Devices

All Devices

Install Location

Prefer External

Internet Access

Auto

Write Permission

Internal

Filter Touches When Obscured

✓

Sustained Performance Mode

✓

Low Accuracy Location

✓

Chrome OS Input Emulation

✓

Android TV Compatibility

✓

Warn about App Bundle size

✓

App Bundle size threshold

150

Active Input Handling*

Input Manager (Old)

Script Compilation

Scripting Define Symbols

BUILDING_MODULE

FLOORMAP_MODULE

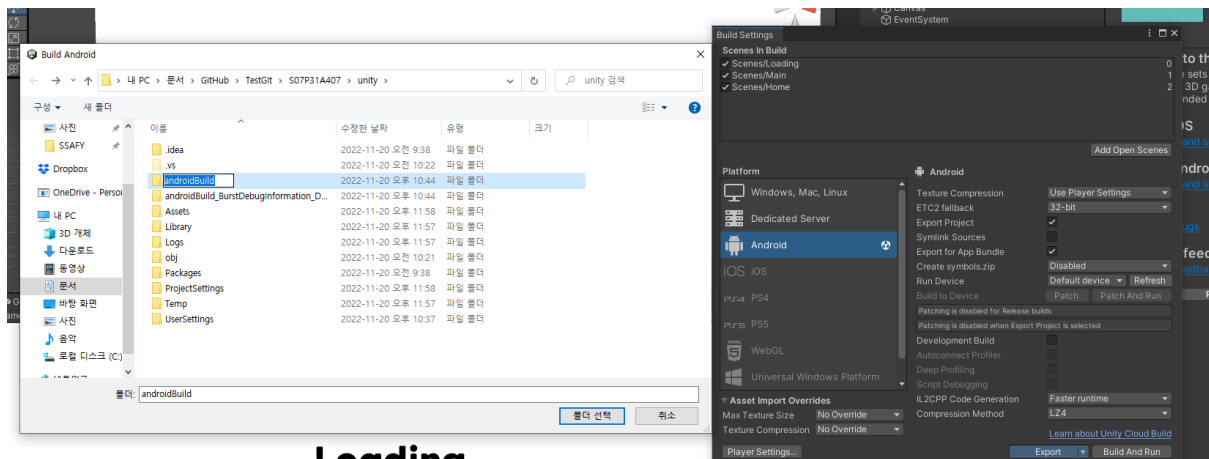
PHOTON_UNITY_NETWORKING

PUN_2_OR_NEWER

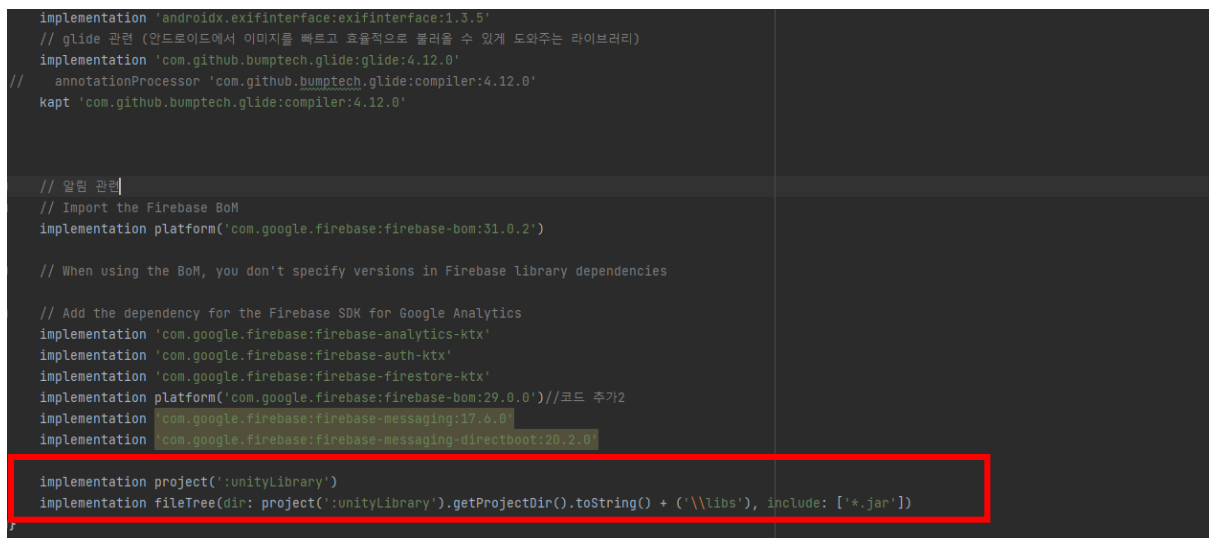
PUN_2_OR_NEWER

PUN_2_19_OR_NEWER

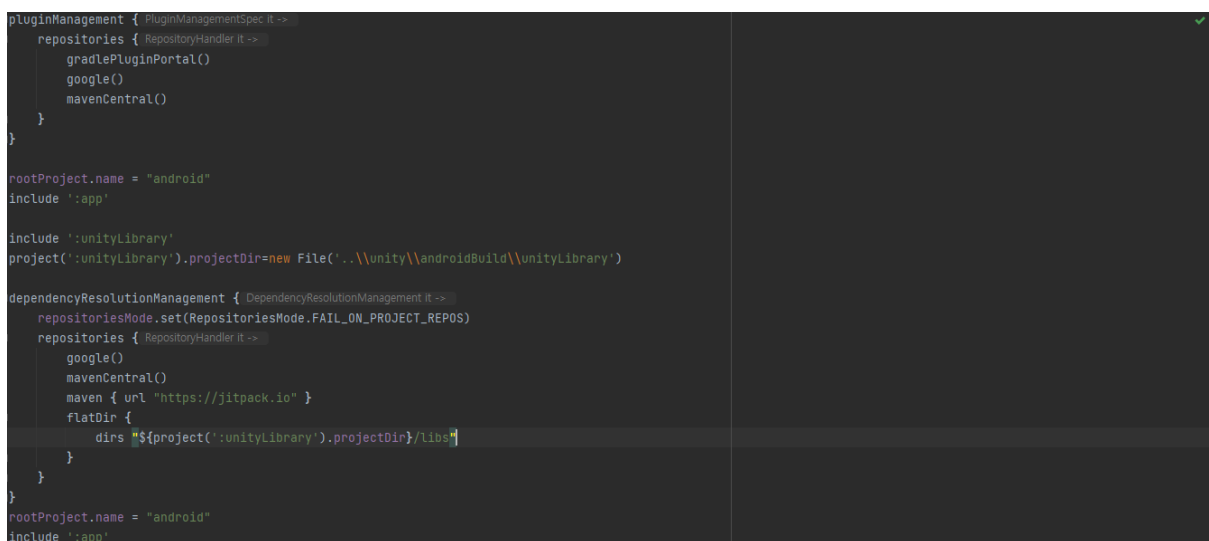
■ WunityWandroidBuild 폴더를 생성 후 export



■ build.gradle(android.app)을 dependencies 에 다음을 추가



■ settings.gradle 을 다음과 같이 수정.



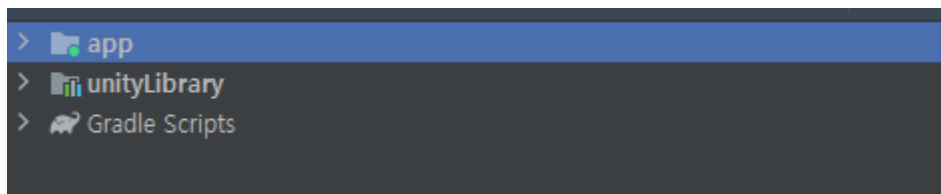
- gradle.properties 을 다음과 같이 수정.

```
org.gradle.jvmargs=-Xmx4096M -Dfile.encoding=UTF-8
org.gradle.parallel=true
unityStreamingAssets=.json
unityTemplateVersion=3
android.useAndroidX=true
android.enableJetifier=true
android.bundle.enableUncompressedNativeLibs=true
```

- AppWresWvaluesWstring.xml 에 다음을 추가한 후 gradle 빌드 한 번 진행

```
<string name="game_view_content_description">Game View</string>
```

- 프로젝트 구조가 다음과 같이 되었다면 아래 과정 수행.



- build.gradle(android.unityLibrary)의 buildToolsVersion 을 30.0.3으로 수정

```
android {
    compileSdkVersion 32
    buildToolsVersion '30.0.3'
```

- unityLibrary 의 manifestsWAndroidManifest.xml 의 해당 부분 삭제(앱 두개 생성 방지)

```
<intent-filter>
    <action android:name="android.intent.action.MAIN" />
    <category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
```

- unityLibrary 의 javaWcomWunity3dWplayerWUnityPlayerActivity 의 해당 부분 추가

```
{
    requestWindowFeature(Window.FEATURE_NO_TITLE);
    super.onCreate(savedInstanceState);

    String cmdLine = updateUnityCommandLineArguments(getIntent().getStringExtra( name: "unity"));
    getIntent().putExtra( name: "unity", cmdLine);
    mUnityPlayer = new UnityPlayer( context: this, iUnityPlayerLifecycleEvents: this);
    String tok = getIntent().getStringExtra( name: "token");
    mUnityPlayer.UnitySendMessage( s: "Panel", st: "Init", tok);
    setContentView(mUnityPlayer);
    mUnityPlayer.requestFocus();
}
```

- app 의 manifests\AndroidManifest.xml 의 해당 부분 추가

```
<activity
    android:name="com.unity3d.player.UnityPlayerActivity"
    android:configChanges="mcc|mnc|locale|touchscreen|keyboard|keyboardHidden|navigation|orientation|screenLayout
    |uiMode|screenSize|smallestScreenSize|fontScale|layoutDirection|density"
    android:exported="true"
    android:hardwareAccelerated="false"
    android:launchMode="singleTask"
    android:resizeableActivity="false"
    android:screenOrientation="fullUser"
    android:theme="@style/UnityThemeSelector"
    android:process=":UnityKillsMe"
    tools:replace="android:screenOrientation"
>
</activity>
```

- Wandroid\app\build\outputs\apk\release 폴더에 생성 ->

안드로이드 폰 다운로드 후 설치

