**HIFES**

**I. 개요**

**1. 프로젝트 개요**

**코로나가 종식되면서 이제는 축제도 다시 활성화되는 추세인데, 축제에 참가하는데 불편한 점들이 많습니다. 예를 들면, 부스의 위치를 알기 힘들다거나, 같이 온 친구들과 소통이 잘 안된다거나, 실시간 정보를 받기가 힘들다는 점 등입니다. 그래서 이런 불편한 점들을 해결하기 위해서 HiFes앱을 기획했습니다.**

**2. 프로젝트 사용도구**

* 이슈 관리 : Jira
* 형상 관리 : Gitlab
* 커뮤니케이션 : Mattermost, Notion
* 디자인 : Figma
* UCC :
* CI/CD : Jenkins

**3. 개발환경**

* Java 11
* Spring Boot 2.7.14
* IntelliJ Ultimate
* MariaDB 10.11
* Server : AWS EC2 Ubuntu 20.04.3 LTS

**4. 외부 서비스**

* Naver Map API
* Kakao Login API
* Firebase Cloud Messaging V1

**5. Gitignore 처리한 핵심 키들**

# Gradle files

.gradle/

build/

### oauth ###

/src/main/resources/application-oauth.properties

# Local configuration file (sdk path, etc)

local.properties

# Log/OS Files

\*.log

# IntelliJ

\*.iml

.idea/

misc.xml

deploymentTargetDropDown.xml

render.experimental.xml

# Google Services (e.g. APIs or Firebase)

google-services.json

### Java ###

# Compiled class file

\*.class

# Log file

# BlueJ files

\*.ctxt

# Mobile Tools for Java (J2ME)

.mtj.tmp/

# Package Files #

\*.jar

\*.war

\*.nar

\*.ear

\*.zip

\*.tar.gz

\*.rar

### Gradle ###

.gradle

\*\*/build/

!src/\*\*/build/

# Ignore Gradle GUI config

gradle-app.setting

# Avoid ignoring Gradle wrapper jar file (.jar files are usually ignored)

!gradle-wrapper.jar

# Avoid ignore Gradle wrappper properties

!gradle-wrapper.properties

# Cache of project

.gradletasknamecache

# IntelliJ IDEA

\*.iws

/out/

# User-specific configurations

.idea/caches/

.idea/libraries/

.idea/shelf/

.idea/workspace.xml

.idea/tasks.xml

.idea/.name

.idea/compiler.xml

.idea/copyright/profiles\_settings.xml

.idea/encodings.xml

.idea/misc.xml

.idea/modules.xml

.idea/scopes/scope\_settings.xml

.idea/dictionaries

.idea/vcs.xml

.idea/jsLibraryMappings.xml

.idea/datasources.xml

.idea/dataSources.ids

.idea/sqlDataSources.xml

.idea/dynamic.xml

.idea/uiDesigner.xml

.idea/assetWizardSettings.xml

.idea/gradle.xml

.idea/jarRepositories.xml

.idea/navEditor.xml

# Legacy Eclipse project files

.cproject

.settings/

# virtual machine crash logs (Reference: <http://www.java.com/en/download/help/error\_hotspot.xml>)

## Plugin-specific files:

# mpeltonen/sbt-idea plugin

.idea\_modules/

!/gradle/wrapper/gradle-wrapper.jar

**II 빌드**

**1. 환경변수**

[application.properties](http://application.properties) :

# DATABASE

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

#spring.datasource.url=jdbc:mariadb://localhost:3306/board

#spring.datasource.username=admin

#spring.datasource.password=dodo1234

spring.jpa.show-sql=true

spring.jpa.hibernate.ddl-auto=update

spring.mvc.converters.preferred-json-mapper=gson

spring.mvc.pathmatch.matching-strategy=ant\_path\_matcher

# jwt key

jwt.secretKey=dksfajdk38jdfjf920fj920f2j9f02j0f3j39jfidkjf932j4kjdfd902sfdsijdif9

jwt.access.expiration=21600000

jwt.access.header=Authorization

jwt.refresh.expiration=1209600000

jwt.refresh.header=Authorization

springdoc.version=1.6.9

spring.jpa.hibernate.naming.implicit-strategy=org.hibernate.boot.model.naming.ImplicitNamingStrategyLegacyJpaImpl

spring.jpa.hibernate.naming.physical-strategy=org.hibernate.boot.model.naming.PhysicalNamingStrategyStandardImpl

spring.main.allow-bean-definition-overriding=true

# fcm key

fcm.key.path=gjgs-fcm.json

fcm.key.scope=https://www.googleapis.com/auth/cloud-platform

build.gradle :

plugins {

id 'java'

id 'org.springframework.boot' version '2.7.14'

id 'io.spring.dependency-management' version '1.0.15.RELEASE'

}

group = 'hiFes'

version = '0.0.1-SNAPSHOT'

java {

sourceCompatibility = '11'

}

configurations {

compileOnly {

extendsFrom annotationProcessor

}

}

repositories {

mavenCentral()

}

dependencies {

implementation 'org.springframework.boot:spring-boot-starter-validation'

implementation 'org.springframework.boot:spring-boot-starter-data-jpa'

implementation 'org.springframework.boot:spring-boot-starter-oauth2-client'

implementation 'org.springframework.boot:spring-boot-starter-security'

implementation 'org.springframework.boot:spring-boot-starter-thymeleaf'

implementation 'org.springframework.boot:spring-boot-starter-web'

implementation 'commons-io:commons-io:2.11.0'

implementation 'org.projectlombok:lombok'

developmentOnly 'org.springframework.boot:spring-boot-devtools'

runtimeOnly 'org.mariadb.jdbc:mariadb-java-client'

compileOnly 'org.projectlombok:lombok'

annotationProcessor 'org.projectlombok:lombok'

developmentOnly 'org.projectlombok:lombok'

testImplementation 'org.springframework.boot:spring-boot-starter-test'

testImplementation 'org.springframework.security:spring-security-test'

implementation 'io.jsonwebtoken:jjwt-api:0.11.5'

implementation 'io.jsonwebtoken:jjwt-impl:0.11.5'

implementation group: 'com.google.code.gson', name: 'gson', version: '2.8.5'

implementation 'com.auth0:java-jwt:4.2.1'

implementation group: 'org.apache.poi', name: 'poi', version: '4.1.2'

implementation group: 'org.apache.poi', name: 'poi-ooxml', version: '4.1.2'

implementation group: 'org.apache.tika', name: 'tika-core', version: '2.3.0'

implementation group: 'org.springframework.boot', name: 'spring-boot-starter-webflux'

implementation 'commons-fileupload:commons-fileupload:1.4'

implementation 'org.springdoc:springdoc-openapi-ui:1.6.9'

implementation 'com.google.firebase:firebase-admin:6.8.1'

implementation group: 'com.squareup.okhttp3', name: 'okhttp', version: '4.2.2'

implementation 'org.modelmapper:modelmapper:2.4.4'

}

tasks.named('test') {

useJUnitPlatform()

}

**2. 빌드하기**

Back-Spring Boot

* Gradle 실행
* Bootjar 실

**3. 배포하기**

pipeline {

agent {

docker {

// 젠킨스에서 사용할 Docker 이미지 설정

// 여기서는 Gradle과 Java가 설치된 이미지를 사용합니다.

image 'gradle:jdk11'

args '--network=host' // Docker 컨테이너를 호스트의 네트워크와 공유

}

}

environment {

// 빌드 결과물이 저장될 경로

BUILD\_DIR = "/var/www/backend"

DOCKER\_CONTAINER\_NAME = "hifes\_develop.app"

}

stages {

stage('Git Clone') {

steps {

git branch: 'develop', credentialsId: '45c98653-ac5e-4ec6-ac6f-9fb104c5dbc9',

url: '<https://lab.ssafy.com/s09-webmobile4-sub2/S09P12D104>'

}

}

stage('Build') {

steps {

sh 'pwd'

sh 'ls -al'

// 스프링 프로젝트 빌드

sh 'chmod +x backend/hiFes/gradlew'

sh 'cd backend/hiFes && ./gradlew clean build'

}

}

stage('Deploy') {

steps {

// 기존 서버 중지

sh "docker stop $DOCKER\_CONTAINER\_NAME || true" // 기존 컨테이너가 없을 경우를 위해 무시

sh "docker rm $DOCKER\_CONTAINER\_NAME || true" // 기존 컨테이너가 없을 경우를 위해 무시

// 빌드된 파일을 원하는 위치로 배포

// 예: 서버에 배포하는 경우

sh "mkdir -p $BUILD\_DIR"

sh "cp backend/hiFes/build/libs/hiFes-0.0.1-SNAPSHOT.jar $BUILD\_DIR"

sh "cd $BUILD\_DIR"

sh "pwd"

sh "ls -al"

// 새로운 서버 실행

sh "docker run -d -p 8001:8001 --name $DOCKER\_CONTAINER\_NAME --network=host -v $BUILD\_DIR:/app -v /home/ubuntu/images:/home/ubuntu/images gradle:jdk11 java -jar /app/hiFes-0.0.1-SNAPSHOT.jar"

}

}

}

post {

//success {

//echo '빌드 성공! 서버가 구동되었습니다.'

// 추가 작업 수행 가능

//}

//failure {

//echo '빌드 실패...'

// 실패 시 추가 작업 수행 가능

//}

//mattermost 연동 하고 싶을 시 mattermost plugin 설치 후 작성

success {

script {

def Author\_ID = sh(script: "git show -s --pretty=%an", returnStdout: true).trim()

def Author\_Name = sh(script: "git show -s --pretty=%ae", returnStdout: true).trim()

mattermostSend (color: 'good',

message: "배포 성공!!!: ${env.JOB\_NAME} #${env.BUILD\_NUMBER} by ${Author\_ID}(${Author\_Name})\\n(<${env.BUILD\_URL}|Details>)",

endpoint: '<https://meeting.ssafy.com/hooks/dokezizx7pfw7d89iftxd6mqww>',

channel: 'legend'

)

}

}

failure {

script {

def Author\_ID = sh(script: "git show -s --pretty=%an", returnStdout: true).trim()

def Author\_Name = sh(script: "git show -s --pretty=%ae", returnStdout: true).trim()

mattermostSend (color: 'danger',

message: "배포 실패!!!: ${env.JOB\_NAME} #${env.BUILD\_NUMBER} by ${Author\_ID}(${Author\_Name})\\n(<${env.BUILD\_URL}|Details>)",

endpoint: '<https://meeting.ssafy.com/hooks/dokezizx7pfw7d89iftxd6mqww>',

channel: 'legend'

)

}

}

}

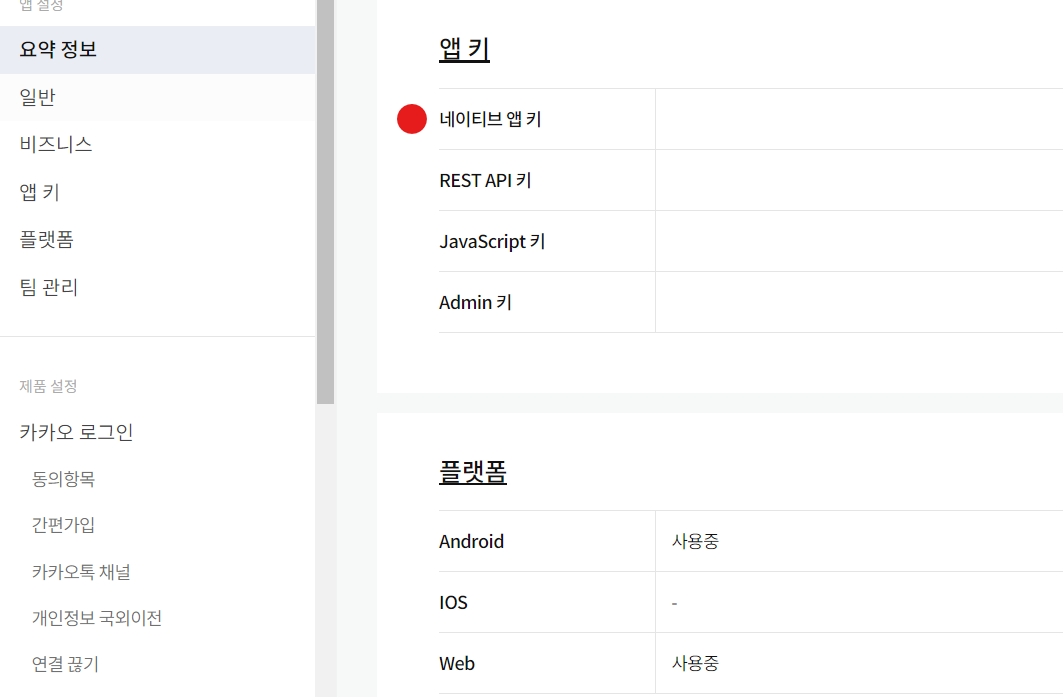
}

**하이페스 안드로이드 & 웹 포팅 메뉴얼**

카카오 소셜 로그인

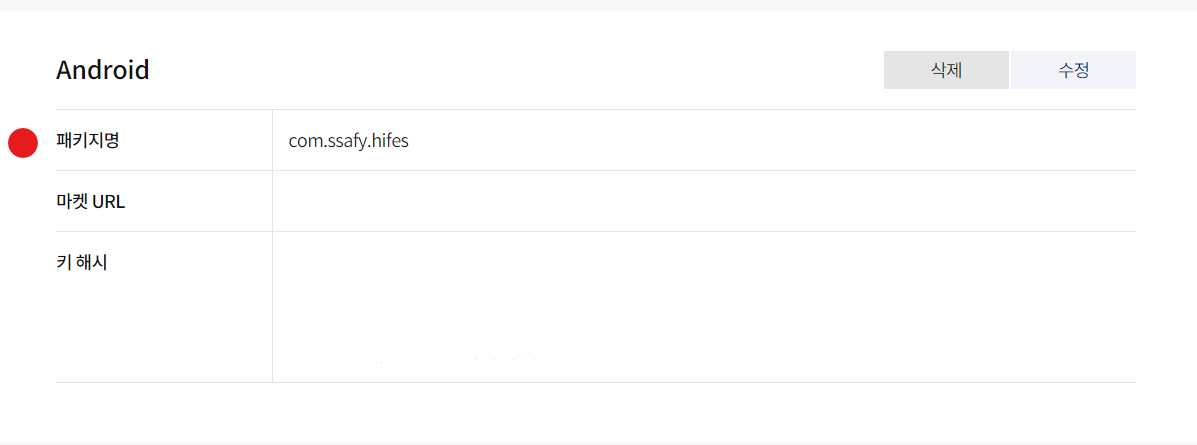






애플리케이션을 추가하여 안드로이드는 네이티브 앱 키, 웹은 JavaScript키를 활용하여 카카오 서버와 통신합니다.

원하는 플랫폼을 선택하여 필요한 정보를 입력합니다.



안드로이드는 사용할 패키지명을 등록합니다.

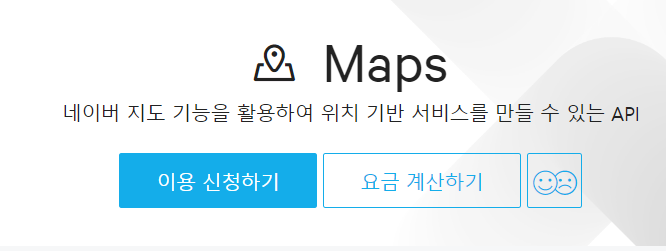


웹에서는 카카오 로그인 이후 리다이렉트할 도메인을 등록합니다.

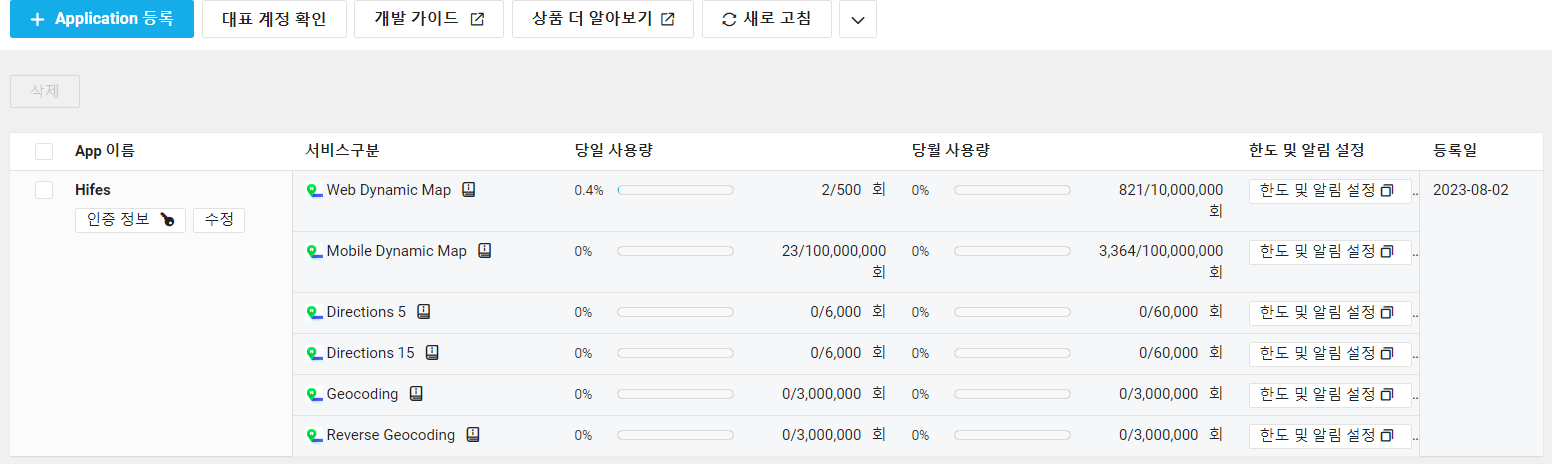
로그인 시도 후 리다이렉트 관련 오류가 발생하면, 해당 url을 추가하면 문제를 해결할 수 있습니다.

네이버 지도 API

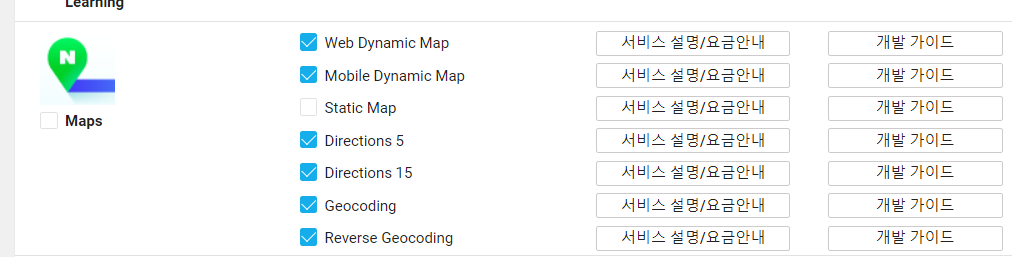
<https://www.ncloud.com/product/applicationService/maps>



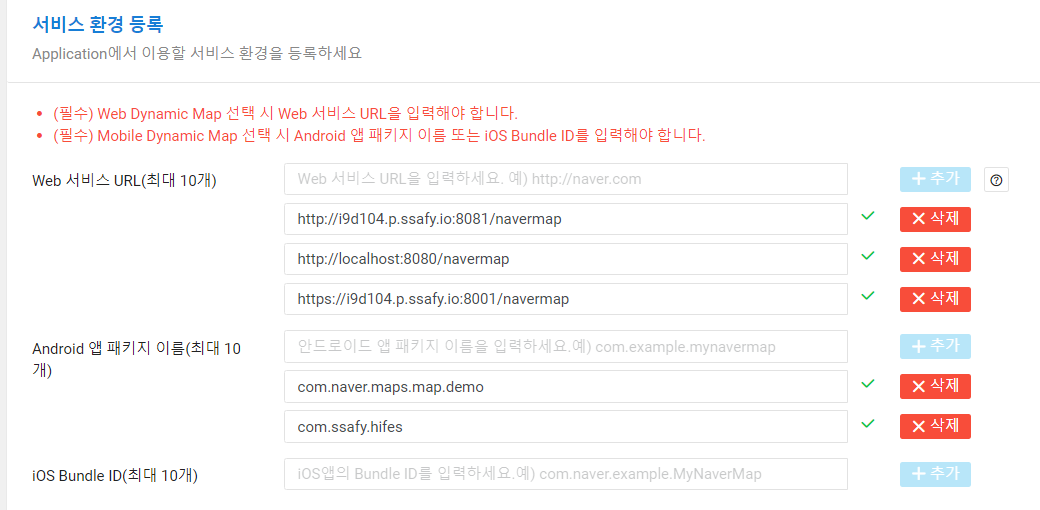
이용 신청하기 버튼을 누릅니다.



Application을 등록합니다.

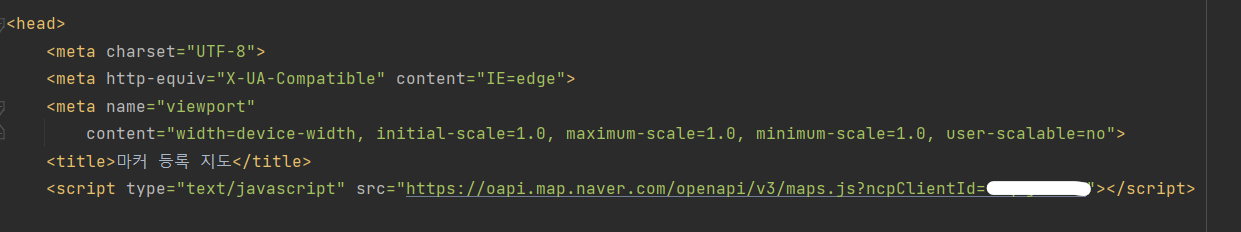


지도를 사용할 것이기에 해당하는 부분을 체크합니다.



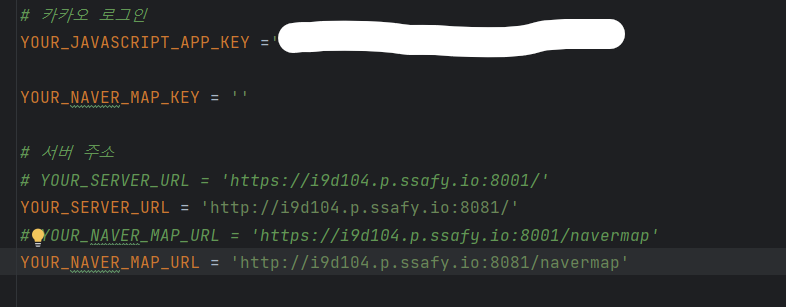
카카오와 마찬가지로 웹 URL과 안드로이드 패키지명을 사용하여 해당 웹 및 모바일에서 네이버 지도 API를 사용할 수 있습니다.

웹의 경우 해당 url에서 지도를 보여줄 html을 반환해야 합니다.



인증정보를 클릭해서 ClientId를 조회하고, 해당 Id를 사용하여 지도 기능을 활성화 할 수 있습니다.

웹 환경 변수



앱을 빌드할 때 사용하는 .env파일 내용입니다.

개인마다 달라질 수 있는 정보값, 공개해서는 안 되는 값(JS KEY)을 저장하기 위해 사용합니다.

루트 디렉토리에 .env파일을 작성하고, 위의 내용을 필수적으로 입력해야합니다.

Flutter 의존성

Flutter version: 3.13.0

kakao\_flutter\_sdk: ^1.3.1

flutter\_svg: ^2.0.7

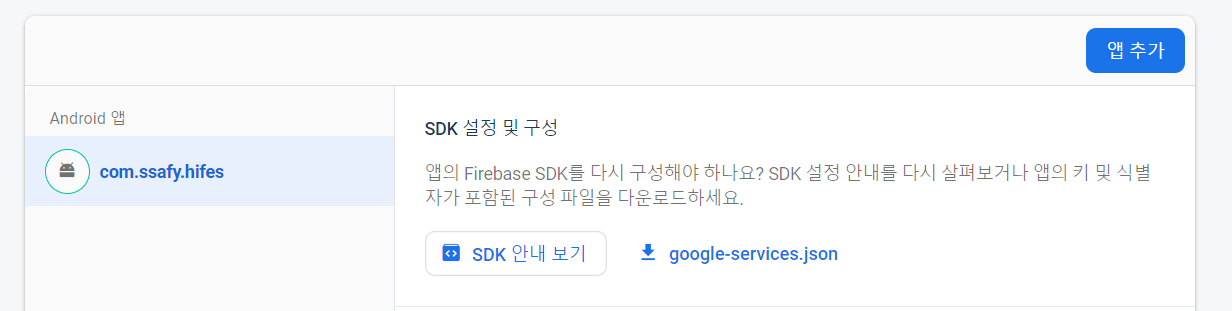
get: ^4.6.5

file\_picker: ^5.3.3

http: ^0.13.3

dio: ^5.3.2

FCM (FirebaseCloudMessaging)



프로젝트 APP에 google-services.json 추가

RealtimeDB



규칙 수정

Android Studio

IDE

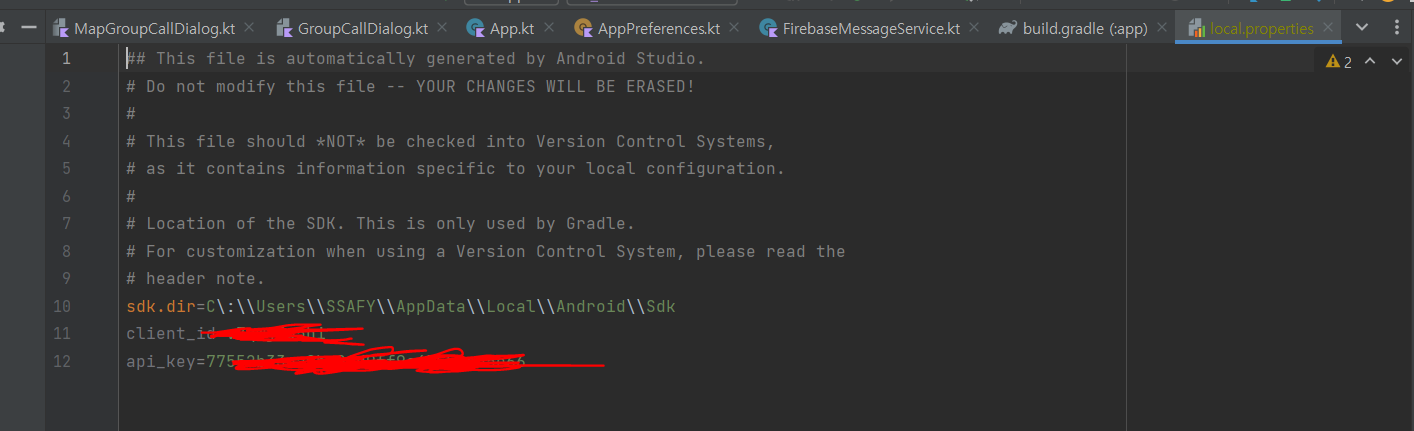
Giraffe 2022.3.1 Version

Project

Android Gradle Plugin Version 8.0.2

Gradle Version 8.0

local.Properties



client\_id : 네이버 클라이언트 아이디 추가

api\_key : 카카오 네이티브키 추가

Libraries

// ViewPager

implementation "com.google.accompanist:accompanist-pager:0.20.1"

implementation "com.google.accompanist:accompanist-pager-indicators:0.20.1"

// 애니메이션

implementation "androidx.compose.animation:animation:1.4.3"

// 서버와 이미지 처리를 위한 gilde

implementation 'com.github.bumptech.glide:glide:4.12.0'

annotationProcessor 'com.github.bumptech.glide:compiler:4.12.0'

// 서버통신을 위한 Retrofit

implementation 'com.squareup.retrofit2:retrofit:2.9.0'

implementation 'com.squareup.okhttp3:logging-interceptor:3.11.0'

// 느슨한 통신규칙을 위해서 gson

implementation 'com.squareup.retrofit2:retrofit:2.9.0'

implementation 'com.squareup.retrofit2:converter-gson:2.9.0'

implementation 'com.google.code.gson:gson:2.8.5'

implementation "com.squareup.retrofit2:converter-scalars:2.1.0"

// Hilt

implementation "com.google.dagger:hilt-android:2.43.2"

kapt "com.google.dagger:hilt-android-compiler:2.43.2"

implementation "androidx.hilt:hilt-navigation-compose:1.0.0-alpha03"

// coroutines

implementation "org.jetbrains.kotlinx:kotlinx-coroutines-android:1.5.2"

// viewModel

implementation 'androidx.lifecycle:lifecycle-viewmodel-ktx:2.4.1'

implementation "androidx.lifecycle:lifecycle-runtime-ktx:2.3.1"

implementation "androidx.lifecycle:lifecycle-viewmodel-ktx:2.3.1"

implementation 'androidx.activity:activity-ktx:1.1.0'

implementation 'androidx.fragment:fragment-ktx:1.6.0'

// SSA를 위한 navigation

implementation "androidx.navigation:navigation-fragment-ktx:2.3.5"

implementation "androidx.navigation:navigation-ui-ktx:2.3.5"

// Gson

implementation 'com.google.code.gson:gson:2.10.1'

// coil

implementation("io.coil-kt:coil-compose:2.4.0")

// compose nav controller

implementation("androidx.navigation:navigation-compose:2.5.3")

// naver map

implementation("com.naver.maps:map-sdk:3.17.0")

implementation 'io.github.fornewid:naver-map-compose:1.3.3'

// liveData for compose

implementation("androidx.compose.runtime:runtime-livedata:1.3.2")

// rating bar for compose

implementation 'com.github.a914-gowtham:compose-ratingbar:1.3.4'

// firebase fcm

implementation(platform("com.google.firebase:firebase-bom:32.2.2"))

implementation("com.google.firebase:firebase-analytics-ktx")

implementation 'com.google.firebase:firebase-messaging-ktx'

implementation("com.google.firebase:firebase-firestore-ktx")

// firebase realtime database

implementation 'com.google.firebase:firebase-database-ktx'

// lottie

implementation "com.airbnb.android:lottie-compose:6.1.0"

// permission for compose

implementation "com.google.accompanist:accompanist-permissions:0.30"

// kakao login

implementation "com.kakao.sdk:v2-user:2.15.0"

// kakao link

implementation "com.kakao.sdk:v2-share:2.11.2"