포팅 매뉴얼

생성자	㉠ 최승준
생성 일시	@2024년 8월 15일 오후 3:17
상태	시작 전
최종 편집자	이재영
태그	

환경 설정

Android

1.1 JDK

- 제품: OpenJDK
- 버전: 21

확인 방법:

1. Android Studio에서 File > Project Structure > SDK Location
2. JDK 위치 확인
3. 터미널에서 해당 JDK의 bin 디렉토리로 이동 후 `java -version` 명령어 실행

1.2 JVM (앱 빌드용)

- Java 버전: 1.8 (Java 8)

확인 방법: build.gradle 파일의 compileOptions 섹션 참조

1.3 안드로이드 개발 환경

- Android Studio 버전: Android Studio Koala | 2024.1.1 Patch 1
- compileSdk: 35 (Android 15 Preview - VanillaIceCream)

- minSdk: 24
- targetSdk: 34
- Gradle 플러그인: libs.plugins.android.application
- Kotlin 플러그인: libs.plugins.jetbrains.kotlin.android
- Google Services 플러그인: com.google.gms.google-services
- Hilt 플러그인: com.google.dagger.hilt.android
- Navigation SafeArgs 플러그인: androidx.navigation.safeargs.kotlin

확인 방법: build.gradle 파일 참조, Android Studio의 About 메뉴에서 버전 확인

1.4 AWS S3

- 버킷 이름: local.properties 파일의 aws.bucket.name에 정의됨

확인 방법: local.properties 파일 참조

2. 빌드 환경 변수

- aws.bucket.name
- aws.access.key
- aws.secret.key

3. 배포 시 특이사항

1. Release 빌드 설정:
    - Minify: 비활성화
    - Proguard 규칙: proguard-rules.pro 파일 적용
2. BuildConfig 필드:
    - AWS_BUCKET_NAME
    - AWS_ACCESS_KEY
    - AWS_SECRET_KEY

4. 주요 계정 및 프로퍼티 파일 목록

1. local.properties: AWS 관련 설정 (버킷 이름, 액세스 키, 시크릿 키)
2. google-services.json: Firebase 설정 파일 (프로젝트 루트에 위치)

5. 외부 서비스 정보

5.1 소셜 인증

- Google Sign-In: com.google.android.gms:play-services-auth 라이브러리 사용
- Firebase Authentication: com.google.firebase:firebase-auth-ktx 라이브러리 사용

5.2 코드 컴파일

- Android SDK 버전: 35 (Android 15 Preview - VanillaIceCream)
- Kotlin 버전: 프로젝트의 libs.versions.toml 파일에서 확인 가능

5.3 기타 주요 라이브러리

- Retrofit: 네트워크 통신
- Hilt: 의존성 주입
- Navigation Component: 화면 이동
- Glide: 이미지 로딩
- Coroutines: 비동기 프로그래밍
- DataStore: 데이터 저장
- AWS Android SDK S3: S3 연동
- ZXing: QR 코드 스캔
- Lottie: 애니메이션

Backend

1.1 JDK

- 제품: OpenJDK
- 버전: 21

1.2 Spring Boot

- 버전: 3.3.1

1.3 Gradle

- 버전: gradle 8.8

1.4 데이터베이스

- 제품: MySQL
- 버전: 5.7
- JDBC URL: jdbc:mysql://i11d102.p.ssafy.io:3307/dosunsangdb

2. 빌드 환경 변수

application.properties 파일에 다음 변수들이 정의되어 있습니다:

- spring.datasource.driver=com.mysql.cj.jdbc.Driver
- spring.datasource.jdbc-url=[데이터베이스 URL]
- spring.datasource.username=root
- spring.datasource.password=ss...
- spring.datasource.mapper-locations=classpath:/mapper/**/*.xml
- openai.api.key=[OpenAI API 키]

3. 배포 시 특이사항

1. 프로파일 설정:
   - application.properties 파일에 spring.profiles.include=private 설정이 있음
2. jar 파일 생성:
   - `./gradlew bootJar` 명령어로 실행 가능한 jar 파일 생성
3. 실행 명령어:
   - `java -jar [생성된 jar 파일명].jar`
4. 파일 업로드 제한:
   - 최대 파일 크기: 10MB
   - 최대 요청 크기: 10MB

4. 주요 계정 및 프로퍼티 파일 목록

1. application.properties: 주요 애플리케이션 설정
2. application-private.properties: private 프로파일 설정

5. 외부 서비스 정보

5.1 인증

- JWT (JSON Web Token) 사용

5.2 주요 의존성

- Spring Boot Starter Web
- Spring Boot Starter Thymeleaf
- MyBatis Spring Boot Starter
- MySQL Connector
- Lombok
- Logback & SLF4J
- Springdoc OpenAPI (Swagger)
- OkHttp
- Spring Cloud AWS
- Spring Boot Starter WebSocket
- Spring Boot Starter Security
- JJWT (JSON Web Token for Java)
- Metadata Extractor

5.3 기타 서비스

- OpenAI API 사용

6. 데이터베이스 정보

- 데이터베이스명: dosunsangdb
- 호스트: i11d102.p.ssafy.io
- 포트: 3307
- 사용자명: root

데이터베이스 덤프 생성 방법:

```
mysqldump -h i11d102.p.ssafy.io -P 3307 -u root -p dosunsan
gdb > dump.sql
```

데이터베이스 복원 방법:

```
mysql -h i11d102.p.ssafy.io -P 3307 -u root -p dosunsangdb
< dump.sql
```

7. API 문서

- Swagger UI: /swagger-ui.html
- OpenAPI 문서: /v3/api-docs

ROS

▼ 1. 개발 도구

- VS Code
- colcon
- cmake
- Terminator
- SSH

▼ 2. 개발 환경

- OS : Ubuntu 22.04
- ROS2 Humble
- ros2_control
  - Hardware Abstraction(carlikebot)
- slam_toolbox
  - SLAM(Mapping, Localization)

- Nav2
  - Planner(**Smac Hybrid-A* Planner**)

▼ 3. 환경 변수

- `.bashrc`

```
...
export ROS_DOMAIN_ID=102
export RMW_IMPLEMENTATION=rmw_cyclonedds_cpp
...
```

▼ 4. 설치

▼ ROS2 Humble

- Set locale

```
locale  # check for UTF-8

sudo apt update && sudo apt install locales
sudo locale-gen en_US en_US.UTF-8
sudo update-locale LC_ALL=en_US.UTF-8 LANG=en_US.UTF-
export LANG=en_US.UTF-8

locale  # verify settings
```

- Setup Sources

```
sudo apt install software-properties-common
sudo add-apt-repository universe

sudo apt update && sudo apt install curl -y
sudo curl -sSL https://raw.githubusercontent.com/ros/

echo "deb [arch=$(dpkg --print-architecture) signed-b
```

- Install ROS2 Packages
  - Update

```
sudo apt update
sudo apt upgrade
```

- Deskop - RViz, demos, tutorials

```
sudo apt install ros-humble-desktop
```

- Base - No GUI Tools

```
sudo apt install ros-humble-ros-base
```

- Dev Tools

```
sudo apt install ros-dev-tools
```

▼ ROS2 Packages

- ros2 control

```
sudo apt install ros-humble-ros2-control
sudo apt install ros-humble-ros2-controllers
```

- slam_toolbox

```
sudo apt install ros-humble-slam-toolbox
```

- nav2

```
sudo apt install ros-humble-nav2-bringup
sudo apt install ros-humble-navigation2
```

- etc

```
sudo apt install ros-humble-xacro # for urdf
```

▼ Git repository

- `ros2_ws` 디렉토리 아래 `src` 에 ros source 를 위치

- `colcon build` 로 빌드

```
colcon build --symlink-install
```

▼ 5. 기타

- 환경 변수 체크

```
printenv | grep -i ROS
```

- `ROS_DOMAIN_ID` 변수 설정
  - 도메인 ID를 설정해야 같은 환경으로 설정

```
echo "export ROS_DOMAIN_ID=<your_domain_id>" >> ~/.bashr
```

- ROS에서 Terminal을 새로 생성할 때마다 source 필요
  - `.bashrc` 에서 추가하면 Terminal 실행 시에 자동으로 실행

```
echo "source /opt/ros/humble/setup.bash" >> ~/.bashrc
```

- `colcon build` 자동 완성
  - `.bashrc` 에서 추가하면 `tab`으로 자동 완성

```
echo "source /usr/share/colcon_argcomplete/hook/colcon-a
```

CI/CD

1. 사용 기술

- Jenkins: CI/CD 파이프라인 관리
- Docker: Spring, MySQL 컨테이너화
- Amazon EC2: Jenkins 호스팅

2. Jenkins 구성

- 버전: **2.452.3**

3. Docker 구성

- 버전: [Docker 버전 명시 필요]
- Dockerfile 위치: dosunsang-server root 디렉토리에 위치

4. EC2 인스턴스 정보

- 보안 그룹:
  - Jenkins 접근용 포트 (8080)
  - Spring Boot 접근용 포트 (8081)
  - SSH 접근용 포트 (22)

5. CI/CD 파이프라인 단계

1. 소스 코드 체크아웃 (Git)
   - 브랜치: seungjun
   - 저장소 URL: https://lab.ssafy.com/s11-webmobile3-sub2/S11P12D102.git
   - 인증: Jenkins Credentials (credentialsId: dosunsang)
2. Gradle을 사용한 빌드
   - 작업 디렉토리: dosunsang_server/dosunsang_server
   - gradlew 실행 권한 부여
   - clean build 실행 (테스트 제외)
3. Docker 이미지 빌드
   - 작업 디렉토리: dosunsang_server/dosunsang_server
   - 이미지 이름: dosunsang-server
4. 배포
   - 기존 Docker 컨테이너 중지 및 제거 (있는 경우)
   - 새 Docker 컨테이너 실행
     - 컨테이너 이름: dosunsang-server
     - 포트 매핑: 8081:8080

6. Jenkins 파이프라인 스크립트 (Jenkinsfile)

```
pipeline {
    agent any
    stages {
        stage('Checkout') {
            steps {
                git branch: 'seungjun',
                    url: '<https://lab.ssafy.com/s11-webmob
ile3-sub2/S11P12D102.git>',
                    credentialsId: 'dosunsang'
            }
        }
        stage('Build') {
            steps {
                dir('dosunsang_server/dosunsang_server') {
                    sh 'chmod +x ./gradlew'
                    sh './gradlew clean build -x test'
                }
            }
        }
        stage('Docker Build') {
            steps {
                dir('dosunsang_server/dosunsang_server') {
                    sh 'docker build -t dosunsang-server .'
                }
            }
        }
        stage('Deploy') {
            steps {
                sh '''
                    docker ps -a | grep -q dosunsang-server
&& docker stop dosunsang-server || true
                    docker ps -a | grep -q dosunsang-server
&& docker rm dosunsang-server || true
                    docker run -d --name dosunsang-server -
p 8081:8080 dosunsang-server
                '''
            }
        }
```

```
    }
}
```

음성

1. 환경설정

RPi 4

USB speaker : ABKO SLP20 Mini Stereo Speaker

USB Microphone : ABKO MP16 USB-C Pin Microphone

2. 설치(가상환경)

`pip install python-dotenv`

`pip install paho-mqtt`

`pip install google-cloud-speech google-cloud-texttospeech`

`pip install openai pyaudio pygame`

3. 오디오 모듈 로드

```
sudo modprobe snd_bcm2835
sudo lsmod | grep snd_bcm2835
```

4. API 사용

- open ai를 사용하기 위한 api키를 .env파일 사용
- google cloud speech 를 사용하기 위한 api 서비스키 json파일 사용
- 환경변수 설정

`export GOOGLE_APPLICATION_CREDENTIALS="home/jeongjae/Aichatbot/LLM.json"`

5. MQTT 브로커 설정

```
BROKER = "broker.hivemq.com"  # MQTT 브로커의 IP 주소
PORT = 1883
TOPIC = "raspberry_pi/start_explanation"  #토픽
JETSON_TOPIC = "ros2/nav"  # 젯슨 나노에 보낼 토픽
```

영상

환경설정

- Jetson nano orin
- yolov8
- CUDA 및 Pytorch 버전

설치

- `sudo apt-get install python3-pip python3-venv python3-dev`
- `sudo apt-get install libopencv-dev python3-opencv`

가상환경 설치

- `pip install ultralytics opencv-python-headless numpy`

필요파일

- 커스텀 데이터 학습 모델(best.pt)
- 기본 yolov8 모델(yolov8n.pt)

협업 툴

- GitLab : 형상관리
- Jira : 이슈관리
- Mattermost, notion : 소통
- Figma : 디자인