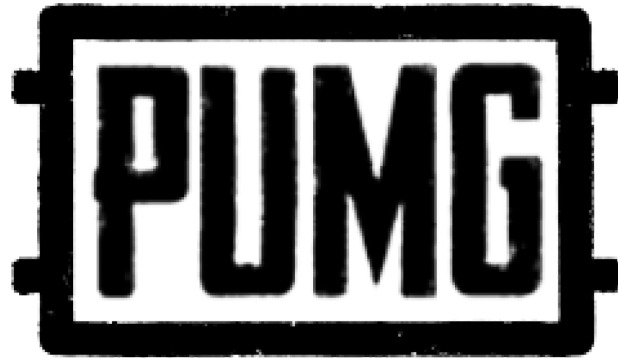


포팅메뉴얼

1. 개요



**PLAYERUNKNOWN'S
META GROUND**

1-1. 프로젝트 사용 도구

- 이슈 관리 : JIRA
- 형상 관리 : GitLab, Plastic SCM
- 커뮤니케이션 : Notion, Mattermost
- 디자인 : Figma
- CI/CD : Jenkins

1-2. 개발환경

- IntelliJ : 2021.2.4 IU-212.5712.43
- JDK : opneJDK 11
- Visual Studio Code : 1.75.0
- Unity : 2021.3.22f1
- SERVER : AWS EC2 Ubuntu 20.04.5 LTS
- DB : MySQL 8.0.33

1-3. 제외된 정보

Spring Boot

키정보

datasource:

driver-class-name: com.mysql.cj.jdbc.Driver

url: [데이터베이스 서버 URL]

username: [데이터베이스 관리자 아이디]

password: [데이터베이스 관리자 비밀번호]

Photon Cloud (게임 소켓 서버 설정)

<https://www.photonengine.com/ko-kr> (Photon 홈페이지 접속) 로그인

- 우측 상단의 [관리 화면으로 이동] 버튼 클릭
- 새 어플리케이션 만들기 클릭
- 어플리케이션 유형 선택 : 멀티플레이어 게임
- Photon 종류 : Reatime
- 어플리케이션 명, 설명, URL 자유기재
- 위 설정대로 어플리케이션 생성 후 메인페이지에서 생성된 어플리케이션에 나타난 어플리케이션 ID 획득



<https://assetstore.unity.com/packages/tools/network/pun-2-free-119922> (유니티 에셋스토어) PUN2 에셋 추가 후 유니티에서 설치

- 유니티 내의 Windows → Photon Unity Networking → PUN Wizard → Setup Project를 선택하여 발급받은 ID 등록

Vivox (음성 채팅 서버 설정)

Unity Gaming Service -> Project 생성 -> Multiplayer -> Voice and Text Chat 설정

Unity Package Manager에서 Add package from git URL로 com.unity.services.vivox 추가

Edit -> Project Settings -> Services에서 Unity Gaming Service에서 생성한 Project 연결

VivoxManager.cs 에서 server, issuer, domain, tokenKey 설정

2. 빌드

Spring Boot 배포

1. JAR 파일 빌드

```
chmod +x gradlew
./gradlew clean build
```

- gradlew 권한 변경
- 이전 기록 지우고 빌드 시작

Dockerizing

2. 도커 빌드

```
docker build -t spring-build .
```

- 스프링을 도커 이미지 생성

4. 동작중인 도커 실행 중지

```
docker ps -f name=springserver-q | xargs --no-run-if-empty docker container stop
docker container ls -a -f name=springserver -q | xargs -r docker container rm
```

- 해당 이름으로 동작 중인 도커 컨테이너 실행 중지
- 컨테이너 삭제

5. 새로 빌드한 도커로 실행

```
docker run -d --name springserver -p 8080:8080 pumg/springserver
```

- 8080 포트로 스프링 시작

Jenkins CI/CD Pipeline

- 젠킨스 새로운 Item → Pipeline으로 Item 생성
- Pipeline 작성

```
pipeline {
    agent any

    stages {
        stage('Pull') {
            steps {
                git branch: '브랜치 명', credentialsId: '인증용 ID', url: '깃의 URL '
            }
        }

        stage('springboot build'){
            steps {
                dir('/var/jenkins_home/workspace/spring/Backend/pumg'){
                    sh 'ls -l'
                    sh 'chmod +x gradlew'
                    sh './gradlew clean build -x test'
                }
            }
        }

        stage('Build') {
```

```

        steps {
            dir('/var/jenkins_home/workspace/spring/Backend/pumg'){
                sh 'docker build -t pumg/springserver ./'
            }
        }

        stage('Deploy') {
            steps{

                sh 'docker ps -f name=springserver -q | xargs --no-run-if-empty docker container stop'
                sh 'docker container ls -a -fname=springserver -q | xargs -r docker container rm'
                sh 'docker images --no-trunc --all --quiet --filter="dangling=true" | xargs --no-run-if-empty docker rmi'
                sh 'docker run -d --name springserver -p 6999:6999 pumg/springserver'


            }
        }

        stage('Finish') {
            steps{
                sh 'docker images -qf dangling=true | xargs -I{} docker rmi {}'
            }
        }
    }
}

```

유니티 배포

PUMG v1.1.zip

 https://drive.google.com/file/d/1O3W8xR85A_TP2hYibYly6RfCS6T-LA0a/view

1. 해당 파일을 다운로드 받는다.
2. 압축을 푼다.
3. 즐겁게 플레이 한다.