

jenkins

GitLab Plugin 설치

Q gitlab

/

이름 ↓	사용가능
<div>GitLab 1.8.0</div> <div>This plugin allows GitLab to trigger Jenkins builds and display their results in the GitLab UI. Report an issue with this plugin</div>	<div></div>
<div>GitLab API Plugin 5.3.0-91.v1f9a_fda_d654f</div> <div>This plugin provides GitLab4J API for other plugins. Report an issue with this plugin</div>	<div></div>

GitLab 연결

- gitlab token을 이용해 GitLab API token을 만들 후 연결

GitLab

Enable authentication for '/project' end-point ?

GitLab connections

Connection name ?
A name for the connection

GitLab

GitLab host URL ?
The complete URL to the GitLab server (e.g. http://gitlab.mydomain.com)

https://lab.ssafy.com/

Credentials ?
API Token for accessing GitLab

GitLab API token (GitLab)

+ Add ▾

고급 ▾

Test Connection

jenkins items

S	W	Name ↓	최근 성공	최근 실패
✓	☀	Battle	22 hr #117	2 days 1 hr #96
✓	☀	Config	18 hr #11	11 days #3
✓	☁	Eureka	5 days 0 hr #5	11 days #1
✓	☀	Flupet	1 day 23 hr #58	6 days 1 hr #27
✓	☀	Gateway	5 days 0 hr #14	—
✓	☀	Member	1 day 21 hr #108	2 days 0 hr #101

item 설정

- Build Triggers
해당 URL과 고급 탭에서 생성한 token을 GitLab webhook에 사용

Build Triggers

- ☐ Build after other projects are built ?
- ☐ Build periodically ?
- ☒ Build when a change is pushed to GitLab. GitLab webhook URL: https://jenkins.fluffit.shop/project/Battle ?

Enabled GitLab triggers

- ☒ Push Events ?
- ☐ Push Events in case of branch delete ?
- ☒ Opened Merge Request Events ?
- ☐ Build only if new commits were pushed to Merge Request ?
- ☐ Accepted Merge Request Events ?
- ☐ Closed Merge Request Events ?

Rebuild open Merge Requests ?

Never

- ☒ Approved Merge Requests (EE-only) ?
- ☒ Comments ?

Comment (regex) for triggering a build ?

Jenkins please retry a build

고급 ^

- ☒ Enable [ci-skip] ?
- ☒ Ignore WIP Merge Requests ?

Labels that launch a build if they are added (comma-separated) ?

- ☒ Set build description to build cause (eg. Merge request or Git Push) ?
- ☐ Build on successful pipeline events

Pending build name for pipeline ?

- ☐ Cancel pending merge request builds on update ?

Allowed branches

- ☒ Allow all branches to trigger this job ?
- ☐ Filter branches by name ?
- ☐ Filter branches by regex ?
- ☐ Filter merge request by label

Secret token ?

e09d067a6129b05c08c8e79acf4fd5a3

Generate

Pipeline

- Repository URL은 git clone 할 때 사용하는 url과 동일함
- gitlab 개인 토큰을 이용해 Credential을 만들어야 함

Pipeline

Definition

Pipeline script from SCM

SCM ?

Git

Repositories ?

Repository URL ?

https://lab.ssafy.com/s10-final/S10P31D204.git

Credentials ?

wnstj7866/***** (wnstj7866)

+ Add

고급

Add Repository

- 빌드 할 브랜치를 작성
- Jenkinsfile이 있는 경로를 지정 (빌드할 브랜치 기준)

Branches to build ?

Branch Specifier (blank for 'any') ?

*/be/dev-battle

Add Branch

Repository browser ?

(자동)

Additional Behaviours

Add

Script Path ?

backend/FluffitBattle/Jenkinsfile

☒ Lightweight checkout ?

Branches to build and Script Path

- gateway

Branches to build ?

Branch Specifier (blank for 'any') ?

Add Branch

Repository browser ?

(자동) ▼

Additional Behaviours

Add ▼

Script Path ?

- eureka

Branches to build ?

Branch Specifier (blank for 'any') ?

Add Branch

Repository browser ?

(자동) ▼

Additional Behaviours

Add ▼

Script Path ?

- config

Branches to build ?

Branch Specifier (blank for 'any') ?

*/be/dev-config

Add Branch

Repository browser ?

(자동)

Additional Behaviours

Add ▾

Script Path ?

backend/FluffitConfig/Jenkinsfile

• member

Branches to build ?

Branch Specifier (blank for 'any') ?

*/be/dev-member

Add Branch

Repository browser ?

(자동)

Additional Behaviours

Add ▾

Script Path ?

backend/FluffitMember/Jenkinsfile

• flupet

Branches to build ?

Branch Specifier (blank for 'any') ?

*/be/dev-flupet

Add Branch

Repository browser ?

(자동)

Additional Behaviours

Add ▾

Script Path ?

backend/FluffitFlupet/Jenkinsfile

• battle

Branches to build ?

Branch Specifier (blank for 'any') ?

*/be/dev-battle

Add Branch

Repository browser ?

(자동)

Additional Behaviours

Add ▾

Script Path ?

backend/FluffitBattle/Jenkinsfile

Jenkinsfile

- gateway

```
pipeline {
  agent any

  environment {
    JWT_SECRET = credentials('JWT_SECRET')
  }

  stages {
    stage('Gradle build') {
      steps {
        dir('backend/FluffitGateway') {
          sh 'chmod +x ./gradlew'
          sh './gradlew clean build'
        }
      }
    }

    stage('Docker image build') {
      steps {
        dir('backend/FluffitGateway') {
          sh "docker stop gateway || true && docker rm gateway || true"
          sh 'docker rmi gateway || true'
          sh 'docker build -t gateway .'
        }
      }
    }

    stage('Docker container run') {
      steps {
        dir('backend/FluffitGateway') {
          sh 'docker run -d -e JWT_SECRET=${JWT_SECRET} -p 8000:8000 --name gateway -e TZ=Asia/Seoul --network dev-net gateway'
        }
      }
    }

    stage('Cleanup dangling images') {
      steps {
        sh 'docker image prune -f'
      }
    }
  }
}
```

- eureka

```
pipeline {
  agent any

  stages {
    stage('Gradle build') {
      steps {
        dir('backend/FluffitEureka') {
          sh 'chmod +x ./gradlew'
        }
      }
    }
  }
}
```

```

        sh './gradlew clean build'
    }
}

stage('Docker image build') {
    steps {
        dir('backend/FluffitEureka') {
            sh "docker stop eureka || true && docker rm eureka || true"
            sh 'docker rmi eureka || true'
            sh 'docker build -t eureka .'
        }
    }
}

stage('Docker container run') {
    steps {
        dir('backend/FluffitEureka') {
            sh 'docker run -d -p 8761:8761 --name eureka -e TZ=Asia/Seoul --network dev-net eureka'
        }
    }
}

stage('Cleanup dangling images') {
    steps {
        sh 'docker image prune -f'
    }
}
}
}

```

- config
- apiEncryptionKey.jks(암호화 키) 생성후 볼륨 마운트
- apiEncryptionKey.jks 생성을 위해선 jdk가 설치되어 있어야 함

```

pipeline {
    agent any

    environment {
        JASYPT_KEY = credentials('JASYPT_KEY')
        JKS_PWD = credentials('JKS_PWD')
        JKS_ALIAS = credentials('JKS_ALIAS')
    }

    stages {
        stage('Gradle build') {
            steps {
                dir('backend/FluffitConfig') {
                    sh 'chmod +x ./gradlew'
                    sh './gradlew clean build'
                }
            }
        }

        stage('Docker image build') {
            steps {
                dir('backend/FluffitConfig') {
                    sh "docker stop config || true && docker rm config || true"
                    sh 'docker rmi config || true'
                    sh 'docker build -t config .'
                }
            }
        }

        stage('Docker container run') {
            steps {
                dir('backend/FluffitConfig') {
                    sh 'docker run -d -e JASYPT_KEY=${JASYPT_KEY} -e JKS_PWD=${JKS_PWD} -e JKS_ALIAS=${JKS_ALIAS} -e TZ=Asia/Seoul -p 8888:8888 --net'
                }
            }
        }
    }
}

```

- member

```

pipeline {
    agent any
    environment {
        JWT_SECRET = credentials('JWT_SECRET')
        HMAC_KEY = credentials('HMAC_KEY')
    }

    stages {
        stage('Gradle build') {

```

```

        steps {
            dir('backend/FluffitMember') {
                sh 'chmod +x ./gradlew'
                sh './gradlew clean build'
            }
        }
    }

    stage('Docker image build') {
        steps {
            dir('backend/FluffitMember') {
                sh "docker stop member || true && docker rm member || true"
                sh 'docker rmi member || true'
                sh 'docker build -t member .'
            }
        }
    }

    stage('Docker container run') {
        steps {
            dir('backend/FluffitMember') {
                sh 'docker run -d -e JWT_SECRET=${JWT_SECRET} -e HMAC_KEY=${HMAC_KEY} --name member -e TZ=Asia/Seoul --network dev-net member'
            }
        }
    }

    stage('Cleanup dangling images') {
        steps {
            sh 'docker image prune -f'
        }
    }
}
}

```

- flupet

```

pipeline {
    agent any
    environment {
        DB_URL = credentials('DB_URL')
        DB_USER = credentials('DB_USER')
        DB_PASSWORD = credentials('DB_PASSWORD')
    }

    stages {
        stage('Gradle build') {
            steps {
                dir('backend/FluffitFlupet') {
                    sh 'chmod +x ./gradlew'
                    sh './gradlew clean build'
                }
            }
        }

        stage('Docker image build') {
            steps {
                dir('backend/FluffitFlupet') {
                    sh "docker stop flupet || true && docker rm flupet || true"
                    sh 'docker rmi flupet || true'
                    sh 'docker build -t flupet .'
                }
            }
        }

        stage('Docker container run') {
            steps {
                dir('backend/FluffitFlupet') {
                    sh 'docker run -d -e DB_URL=${DB_URL} -e DB_USER=${DB_USER} -e DB_PASSWORD=${DB_PASSWORD} --name flupet -e TZ=Asia/Seoul --network dev-net flupet'
                }
            }
        }

        stage('Cleanup dangling images') {
            steps {
                sh 'docker image prune -f'
            }
        }
    }
}

```

- battle

```

pipeline {
    agent any

    stages {
        stage('Gradle build') {

```



```
        steps {
            dir('backend/FluffitBattle') {
                sh 'chmod +x ./gradlew'
                sh './gradlew clean build'
            }
        }
    }

    stage('Docker image build') {
        steps {
            dir('backend/FluffitBattle') {
                sh "docker stop battle || true && docker rm battle || true"
                sh 'docker rmi battle || true'
                sh 'docker build -t battle .'
            }
        }
    }

    stage('Docker container run') {
        steps {
            dir('backend/FluffitBattle') {
                sh 'docker run -d --name battle -e TZ=Asia/Seoul --network dev-net battle'
            }
        }
    }

    stage('Cleanup dangling images') {
        steps {
            sh 'docker image prune -f'
        }
    }
}
}
```

jenkins Credentials

- jenkins pipeline의 environment에서 사용하기 위해 jenkins에 등록된 자격 증명(Credentials) 값

- FluffitConfig 환경 변수

```
JASYPT_KEY=YOUR_JASYPT_KEY
JKS_PWD=YOUR_JKS_PWD
JKS_ALIAS=YOUR_JKS_ALIAS
```

- FluffitFlupet 환경변수

```
DB_URL=jdbc:mariadb://fluffit.shop:3308/fluffit_flupet?serverTimezone=UTC&useUnicode=yes&characterEncoding=UTF-8
DB_USER=d204
DB_PASSWORD=gumid204
```

- 키값들

```
JWT_SECRET=YOUR_JWT_SECRET
HMAC_KEY=YOUR_HMAC_KEY
```

설정 파일 암호화를 위한 자바 key tool 사용법(비대칭키 생성 방법)

- keystore파일(jks) 생성

```
keytool -genkeypair -alias apiEncryptionKey -keyalg RSA -dname "CN=Wonjun Jung, OU=API Development, O=fluffit.shop, L=Seoul, C=KR"
-keypass "ben0722" -keystore apiEncryptionKey.jks -storepass "ben0722"
```

- 생성된 jks 파일을 config 컨테이너를 만들 때 볼륨 마운트

```
sh 'docker run -d -e JASYPT_KEY=${JASYPT_KEY} -e JKS_PWD=${JKS_PWD} -e JKS_ALIAS=${JKS_ALIAS}
-e TZ=Asia/Seoul -p 8888:8888 --network dev-net --name config -v /home/ubuntu/key/apiEncryptionKey.jks:/config/apiEncryptionKey.jks config'
```

GitLab Webhook 세팅

- URL : jenkins item url
- Secret token : jenkins item 고급탭에서 생성한 token
- Trigger : be/dev-config 파일에 관한 푸시를 감지함

Webhook

Webhooks enable you to send notifications to web applications in response to events in a group or project. We recommend using an [integration](#) in preference to a webhook.

URL

https://jenkins.fluffit.shop/project/Config

URL must be percent-encoded if it contains one or more special characters.

- ☒ Show full URL
- ☐ Mask portions of URL
Do not show sensitive data such as tokens in the UI.

Secret token

.....

Used to validate received payloads. Sent with the request in the `X-Gitlab-Token` HTTP header.

Trigger

- ☒ Push events
- ☐ All branches
- ☐ Wildcard pattern
- ☒ Regular expression

^be/dev-config\$

Regular expressions such as `^(feature|hotfix)/` are supported.