

개발 프로세스 실습 3

Hyunchan, Park

<http://oslab.jbnu.ac.kr>

Division of Computer Science and Engineering

Jeonbuk National University



Integration and Deployment

(System) Integration

- 시스템의 구성
 - 최종 시스템은 여러 서브 시스템 혹은 서비스로 구성되고,
 - 서브시스템, 서비스는 여러 컴포넌트들로 구성되고,
 - 컴포넌트는 여러 파일 및 함수들로 구성됨
 - (용어는 환경에 따라 다를 수 있음)
- 시스템의 여러 구성 요소들을 통합하여 최종 시스템으로 빌드하는 작업
 - 모든 구성요소들을 bottom-up 으로 조립하며, 최종 시스템을 빌드
 - Integration test: 각 통합 단계 별로 테스트를 진행
 - 최하위 컴포넌트, 모듈들은 각기 Unit test를 통과한 상태
- * 참고: System Integration(SI) 업체
 - 고객사에 납품할 최종 시스템을 구축하기 위해 기획, 개발, 유지보수, 운영 등을 수행하는 업체
 - 보통 SI업체가 수주하고, 전체 시스템의 설계를 수행하고, 필요한 구성 요소들은 다시 하청을 통해 개발하는 경우가 많음 -> 각 하청에서 올라온 컴포넌트들을 Integration

Deployment (or Delivery)

- 고객에게 서비스를 제공하는 Production 환경에 새로운 서비스를 배포
 - 배포 이슈
 - 가동 중지 시간으로 인한 서비스 가용성 (availability) 저하
 - 서비스의 연속성 보장: 정확한 데이터의 보존 및 이전
 - 새로운 환경(OS, DB 업데이트 등)을 적용하는 경우, 그로 인한 문제점 발생 가능
 - 배포 이전에 실제 사용 환경과 유사한 종류, 강도의 테스트가 필수
 - 무중단 배포: 완전히 새로운 환경을 구성하는 것이 핵심
 - 예) A, B 환경이 있다고 할 때, 기존 배포는 A, 새로운 버전의 테스트는 B에서 진행
 - B환경에서 테스트 완료 후, 서비스 엔드포인트를 A에서 B로 변경해주면 즉각 업데이트가 완료됨. 이후 A환경은 정리하고, 다시 테스트 용도로 사용
 - IP 주소 할당을 변경하거나, 혹은 DNS 레코드를 변경
 - 두 시스템이 동일한 데이터로 작업하도록 시스템이 설계되어 있어야 함
 - 혹은 완전히 새로운 환경을 배포 때마다 구성 (Cloud 기반의 VM, Container 환경)

CI/CD

- Continuous Integration/Deployment (지속적인 통합 및 배포)
 - 목표: 서비스 개선 사항을 빠르게 고객에게 전달하는 것
 - 배포 오버헤드를 최소화하여, 작은 버그 수정 하나도 빠르게 실제 서비스에 반영
 - 정의
 - 통합 및 배포 단계에 자동화를 적용하여,
 - 소스 코드의 변화가 즉각 통합 및 배포로 이행되도록
 - 개발 프로세스를 운영하는 것
 - 단계별로 자동화된 전체 프로세스를 CI/CD Pipeline 이라고 함
- 참고: [Red Hat CI/CD](#), [ITworld CI/CD](#), [카카오 사례](#), [컴투스 사례](#), [우아한 형제들 사례](#)



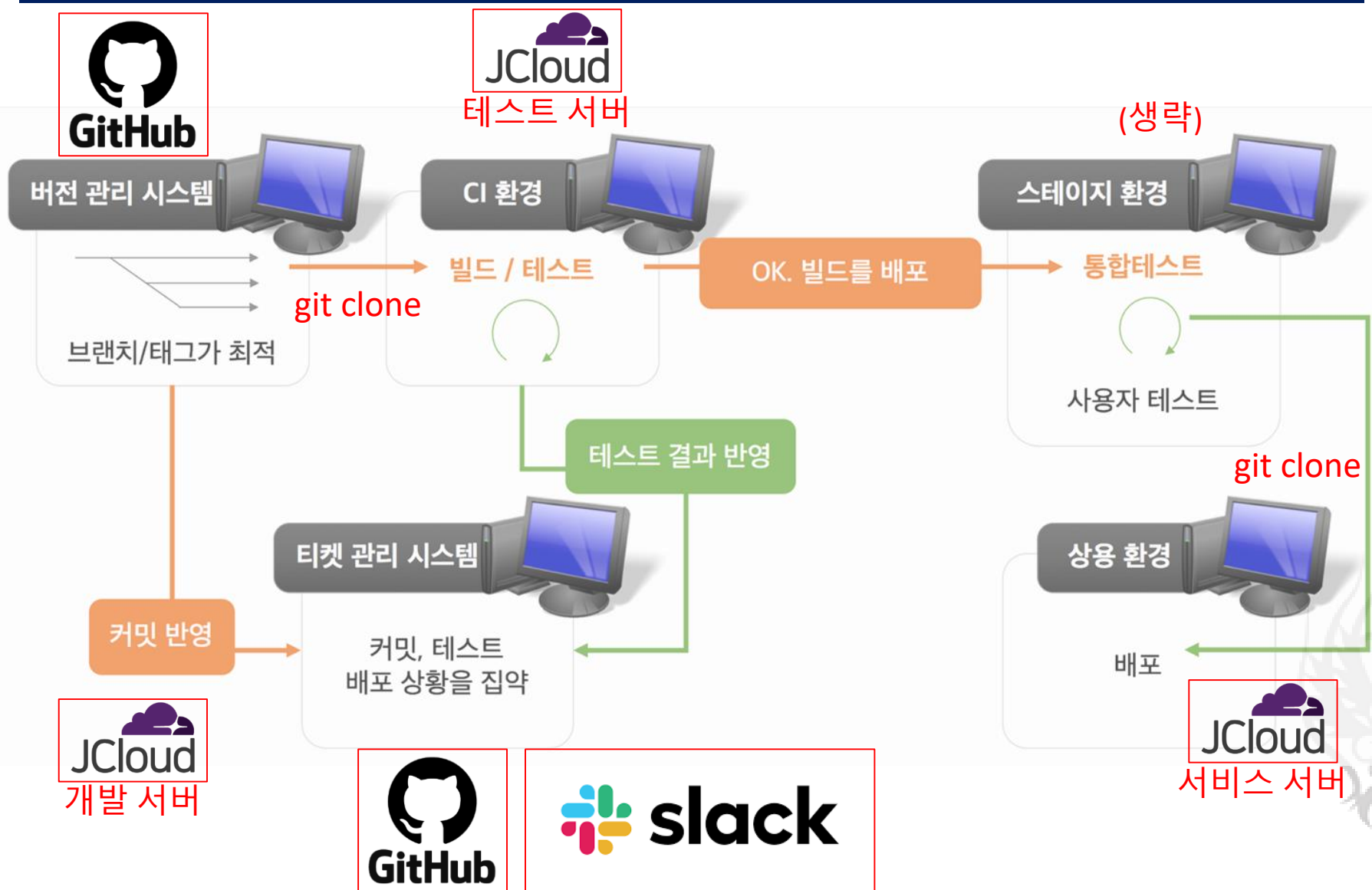
우아한 형제들 CI/CD 시스템



현재 우리의 I/D 는?

1. Development
2. Code Convention Check & Unit test
3. Commit
4. Pull Request
5. Code review
6. Merge <- Merge 가 이루어지면, 수동으로 I/D 시작
7. Integration and Build (on the Build environment, deployed by git clone)
8. Deploy the build results to the Test environment
9. Integration test <- 수동으로 테스트 환경에 deploy 후, 테스트 진행
10. Q/A: Quality Assurance on the Test environment
11. Deploy to the Service environment <- 테스트 통과 후, 수동으로 배포

현재 우리의 I/D 는?



실제로 개발 프로세스를 돌려보자

- 간단한 코드 수정
 - “영화” 입력에 대한 메시지 수정
- Commit (including code convention check)
- Pull request and Code review (리뷰는 생략)
- Merge: Repository updated!
- Integration 환경(테스트 서버)로 배포
 - Git clone (or pull) and Service start
- Mocha를 활용한 최종 테스트
- Service (production) 환경으로 배포
 - Git clone (or pull) and Service restart
 - .env 파일 오느

Service Environment 구성

- J-Cloud instance 추가 생성
 - Deployment server, production server, service server...
 - 서비스 용으로만 사용되는 서버
 - 우선 수행을 위한 환경 설정 (기존 개발 서버와 동일한 환경 구성)
 - Nodejs, NPM 등등 (이 환경에서는 테스트는 수행하지 않음)
 - 본래는 빌드 결과물인 바이너리를 받아와서 서비스를 재시작하는 동작을 해야하지만,
 - 우리는 빌드가 없으므로 그냥 git pull 로 새 버전을 받아와서 서비스를 재시작하는 동작을 수행
 - 사실 챗봇도 테스트용을 따로 만들어서 다른 Token 으로 관리해야 함
 - 기존 서비스가 동작 중에도 개발 과정이 진행되어야 하므로



Jenkins

Todo 1: Jenkins 서버 구축

Todo 2: Jenkins 환경 설정

Todo 3: Jenkins 빌드 설정



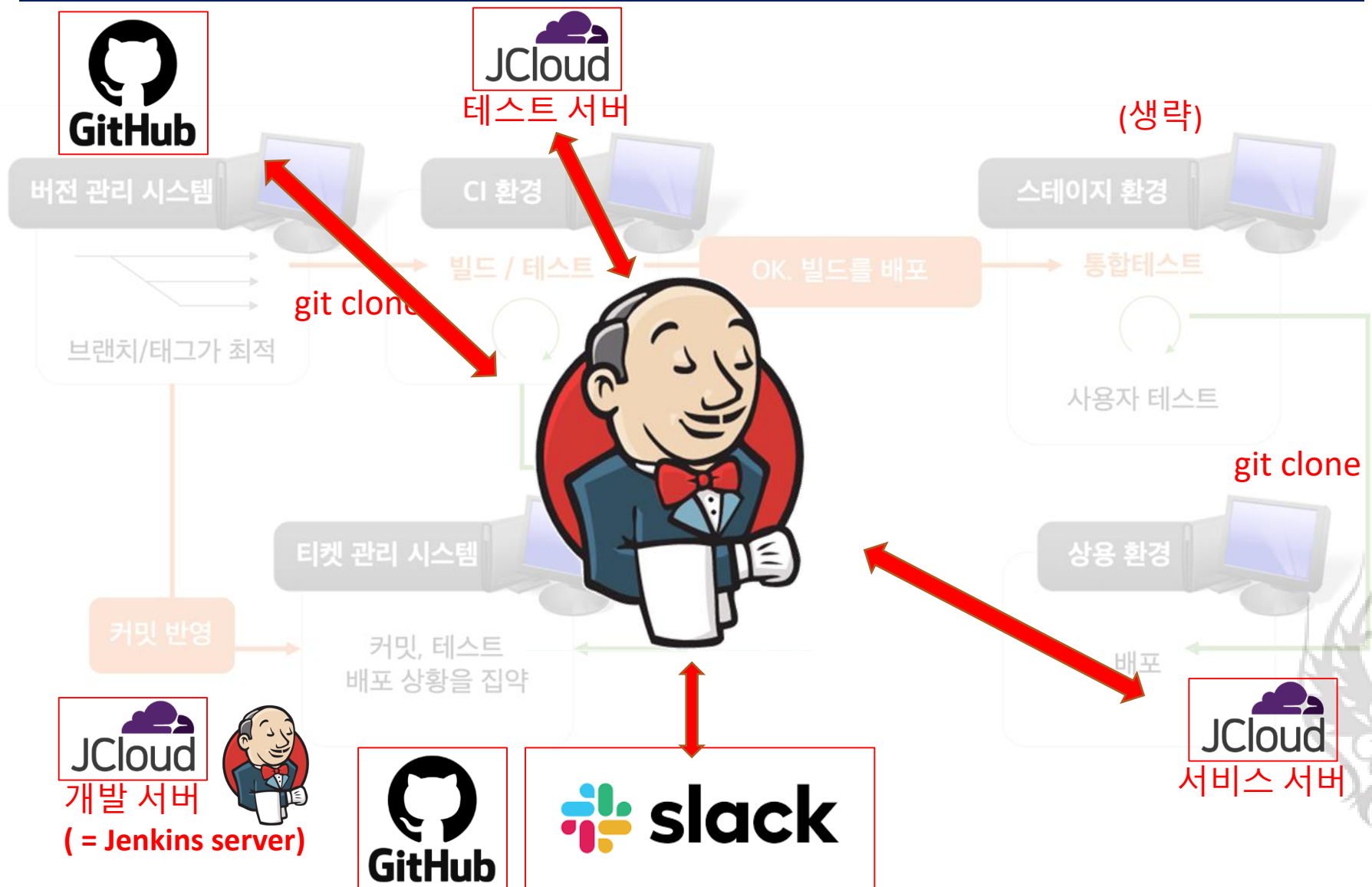
Jenkins



- Source repository, 개발, 빌드, 테스트, 배포 서버 등을 중앙에서 총괄하여, CI/CD 작업을 수행해주는 가장 널리 쓰이는 도구
- The open source automation server
 - Jenkins provides hundreds of plugins to support building, deploying and automating any project
 - CI/CD 외에 다양한 automation 을 구성할 수 있음
 - 초기 Sun microsystems 에서 Hudson 으로 개발 및 사용되었고, Oracle 이 Sun 을 인수한 후, 독자적인 오픈 소스 프로젝트로 분화함
- 다른 CI/CD 도구
 - Travis CI: GitHub 연동, 무료, CI 작업마다 환경을 새로 구성하여야 해서 다소 느림
 - 기타 다양한 클라우드 시스템에서 각기 서비스 제공
 - AWS: CodePipeline



Jenkins 서버를 추가 (해야 하지만..개발 서버에 더부살이)



Instructions

- Jenkins Install
 - `sudo apt-get update`
 - `sudo apt-get install openjdk-8-jdk`
 - `sudo wget -q -O - https://pkg.jenkins.io/debian/jenkins-ci.org.key | sudo apt-key add -`
 - `sudo sh -c "echo deb http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list"`
 - `sudo apt-get update`
 - `sudo apt-get install Jenkins`
 - `sudo service jenkins start`
- (Troubleshooting for apt-get update error)
 - `sudo rm -rf /var/lib/dpkg/lock-frontend`
 - `sudo rm -rf /var/cache/apt/archives/lock`



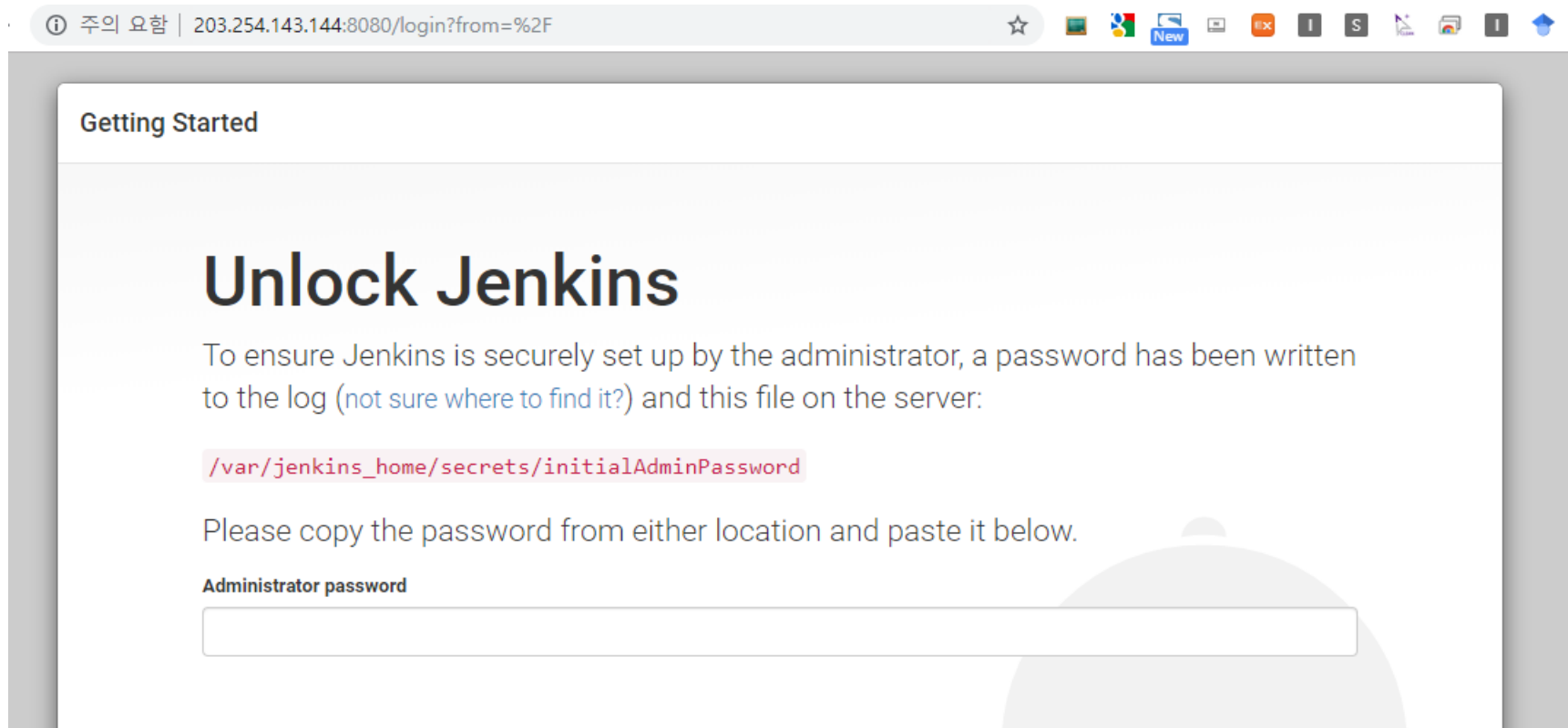
Jenkins 초기화 (10분 이상 소요됨)

```
ubuntu@hcpark:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword  
66601935dca44c17971327c3073e09d7
```

(공통 주소 부분)

http://203.254.143.211:17037

* Internal IP: 192.168.0.37



기본 구성으로 설정

Getting Started

Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

Install suggested plugins

Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.

Jenkins 2.60.3

Admin 계정 생성. 전부 admin 이라고 입력

Getting Started

Create First Admin User

계정명:

암호:

암호 확인:

이름:

이메일 주소:

Instance Configuration

Jenkins URL:

`http://203.254.143.211:17037/`

The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the `BUILD_URL` environment variable provided to build steps.

The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

완성!



Jenkins

Todo 1: Jenkins 서버 구축

Todo 2: Jenkins 작업 설정

Todo 3: Jenkins 빌드 설정



Jenkins 작업 설정

- 핵심: 수동으로 했던 일련의 작업을 자동으로 수행하게 설정
- Trigger: Git repository 에 merge 발생
- Task 1. Integration and Test (on Integration server)
 - Git clone or pull
 - Run tests
 - Pass 인 경우, 다음 작업 수행
- Task 2. Deployment (on Deployment server)
 - git clone or pull

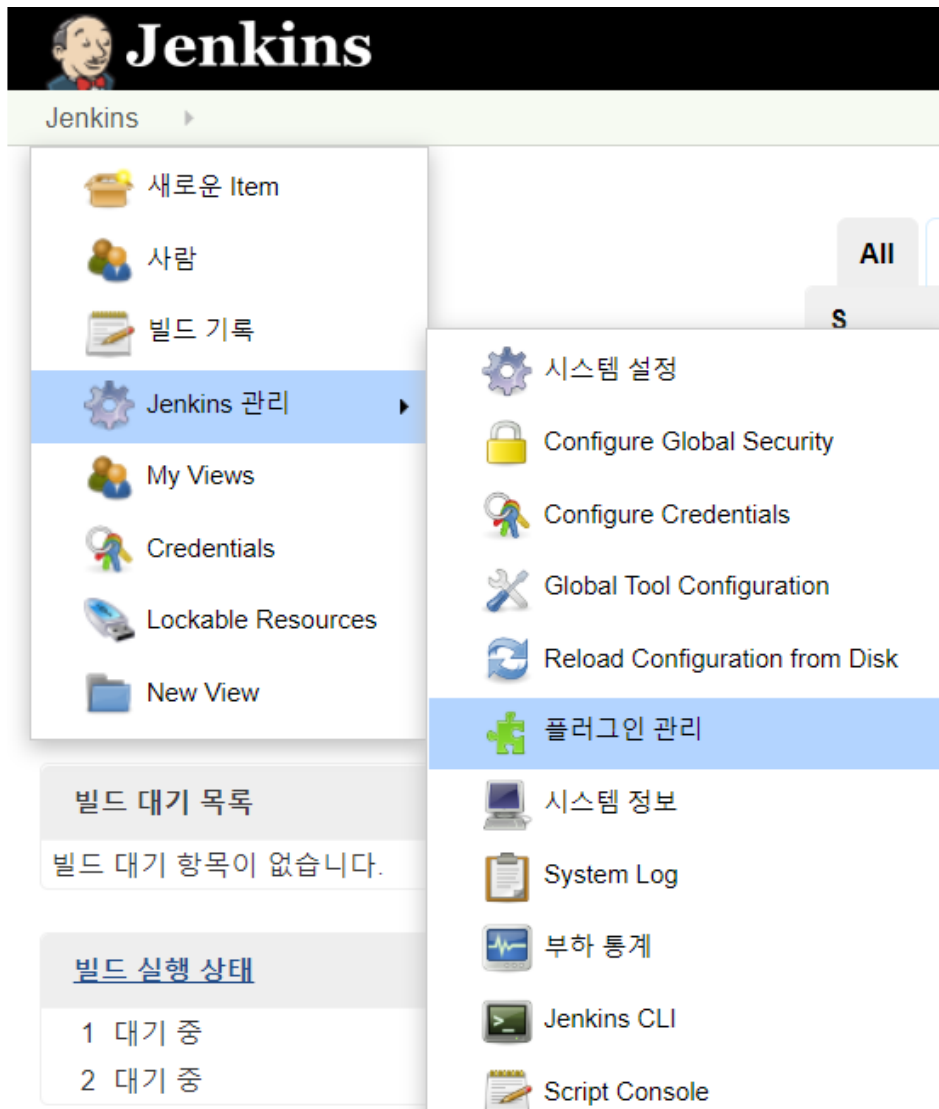


문제점: 원격 서버에 대한 작업 수행

- 어떻게 다른 서버에서 git clone 을 수행시킬 수 있을까?
 - Integration server, Deployment server는 Jenkins 서버와 분리되어 있음
 - (편의상 개발 서버와 Integration server 는 동일)
- 가능한 방법?
 - 다양한 방법이 있으니, 각자 고민해볼 것
- Publish Over SSH Plugin
 - SSH 로 접근해서 원하는 동작을 수행시켜주는 플러그인



Jenkins 플러그인 설치



Jenkins 플러그인 설치

필터:

업데이트된 플러그인 목록 **설치 가능** 설치된 플러그인 목록 고급

설치 ↓	이름	버전
<input type="checkbox"/>	Publish Over SSH Send build artifacts over SSH	1.20.1

재시작 없이 설치하기 **지금 다운로드하고 재시작 후 설치하기** Update information

obtained: 5 hr 45 min ago **지금 확인**

Infrastructure plugin for Publish Over X

● 성공

Publish Over SSH

● 성공

Loading plugin extensions

● Success

➡ [메인 페이지로 돌아가기](#)
(설치된 플러그인을 바로 이용하실 수 있습니다.)

➡ ☐ 설치가 끝나고 실행중인 작업이 없으면 Jenkins 재시작.

Publish Over SSH Plugin 설정



Publish Over SSH Plugin 설정

- J-Cloud 서버에 접근할 때 사용하던 keypair pem 파일의 내용을 복사해서 Key 항목에 붙여넣기

Publish over SSH

Jenkins SSH Key

Passphrase

Path to key

Key

Disable exec

```
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAu9zaAsRkK4nY/V7nhc4MgHsFckYu53uDrlo/ATlkgUUwZSmF
J0qMIABA3ngIR1cn6Vh00pUAwy/ZF6AeUtrwGqNwx08B3TIBggSd7H+0wpggNlrx
OI7/XI7QSpky5bQP08E1yafyK0tcnTbnyslLUAnO4Zv7oH0OH0pdBJEO4aHhD6
ympj3MRVABdToq+GkjsHLtsiLXU4mqqlhw74QPOsZpGWHOLF065dtRNlIXFph6hW
7B3M4M5Hw087E-84Q7B5E-A-MUXALL-31005-X-1-PM
```

☐

SSH 서버 등록

- 두 개의 SSH 서버를 “추가”
- IP는 internal IP 를 사용, Port: 7777
- 아래 Test 버튼 사용해서 success 확인

SSH Server

Name

Integration Server

Hostname

192.168.0.8

Username

ubuntu

Remote Directory

/home/ubuntu/

☐ Use password authentication, or use a different key

Jump host

Port

7777

<input type="checkbox"/>	Instance Name	IP Address
<input type="checkbox"/>	hcpark.jenkins	192.168.0.37
<input type="checkbox"/>	hcpark.deployment	192.168.0.46
<input type="checkbox"/>	hcpark.integration.dev	192.168.0.8

SSH Server

Name

Deployment server

Hostname

192.168.0.46

Username

ubuntu

Remote Directory

/home/ubuntu

☐ Use password authentication, or use a different key

Jump host

Port

7777

실제 작업 등록 시작



The image shows the Jenkins web interface. At the top is the Jenkins logo and a search bar. Below the logo is a sidebar with navigation links: '새로운 Item' (New Item), '사람' (People), '빌드 기록' (Build History), 'Jenkins 관리' (Manage Jenkins), 'My Views', 'Credentials', 'Lockable Resources', and 'New View'. The '새로운 Item' link is highlighted with a red box. The main area displays a welcome message: 'Jenkins에 오신 것을 환영합니다.' (Welcome to Jenkins). Below this, a light blue box contains the text '시작하려면 새 작업을 만들어 주시기 바랍니다.' (Please create a new job to get started). The word '새 작업' (New job) is highlighted with a red box. At the bottom, there are two panels: '빌드 대기 목록' (Build Queue) and '빌드 실행 상태' (Build Execution Status). The '빌드 대기 목록' panel shows '빌드 대기 항목이 없습니다.' (No build items in queue). The '빌드 실행 상태' panel shows '1 대기 중' (1 waiting) and '2 대기 중' (2 waiting).

Jenkins

검색

Jenkins

새로운 Item

사람

빌드 기록

Jenkins 관리

My Views

Credentials

Lockable Resources

New View

빌드 대기 목록

빌드 대기 항목이 없습니다.

빌드 실행 상태

1 대기 중

2 대기 중

Jenkins에 오신 것을 환영합니다.


시작하려면 새 작업을 만들어 주시기 바랍니다.

실제 작업 등록 시작

Enter an item name


Chatbot CICD

» Required field




Freestyle project

이것은 Jenkins의 주요 기능입니다. Jenkins은 어느 빌드 시스템과 어떤 SCM(형상관리)으로 묶인 당신의 프로젝트를 빌드할 것이고, 소프트웨어 빌드보다 다른 어떤 것에 자주 사용될 수 있습니다.




Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.




Multi-configuration project

다양한 환경에서의 테스트, 플랫폼 특성 빌드, 기타 등등 처럼 다수의 서로다른 환경설정이 필요한 프로젝트에 적합함.




Folder

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.



GitHub Organization

Scans a GitHub organization (or user account) for all repositories matching some defined markers.



GitHub Organization


Scans a GitHub organization (or user account) for all repositories matching some defined markers.

General

General소스 코드 관리빌드 유발빌드 환경Build빌드 후 조치

설명

Jenkins Project for ChatBot



[Plain text] [미리보기](#)

☐ GitHub project

☐ This build requires lockable resources






☐ Throttle builds

☐ 오래된 빌드 삭제

☐ 이 빌드는 매개변수가 있습니다

☐ 빌드 안함

☐ 필요한 경우 concurrent 빌드 실행



고급...



소스 코드 관리

소스 코드 관리

☐ None

☒ Git

Repositories

Repository URL

https://github.com/hyunchan-park/chatbot

Failed to connect to repository : Command "git ls-remote -h -- https://github.com/hyunchan-park/chatbot HEAD" returned status code 128:

stdout:

stderr: remote: Invalid username or password.

fatal: Authentication failed for 'https://github.com/hyunchan-park/chatbot'

Credentials

- none -

Add

Name

Refspec

Add Repository

Branches to build

Branch Specifier (blank for 'any')

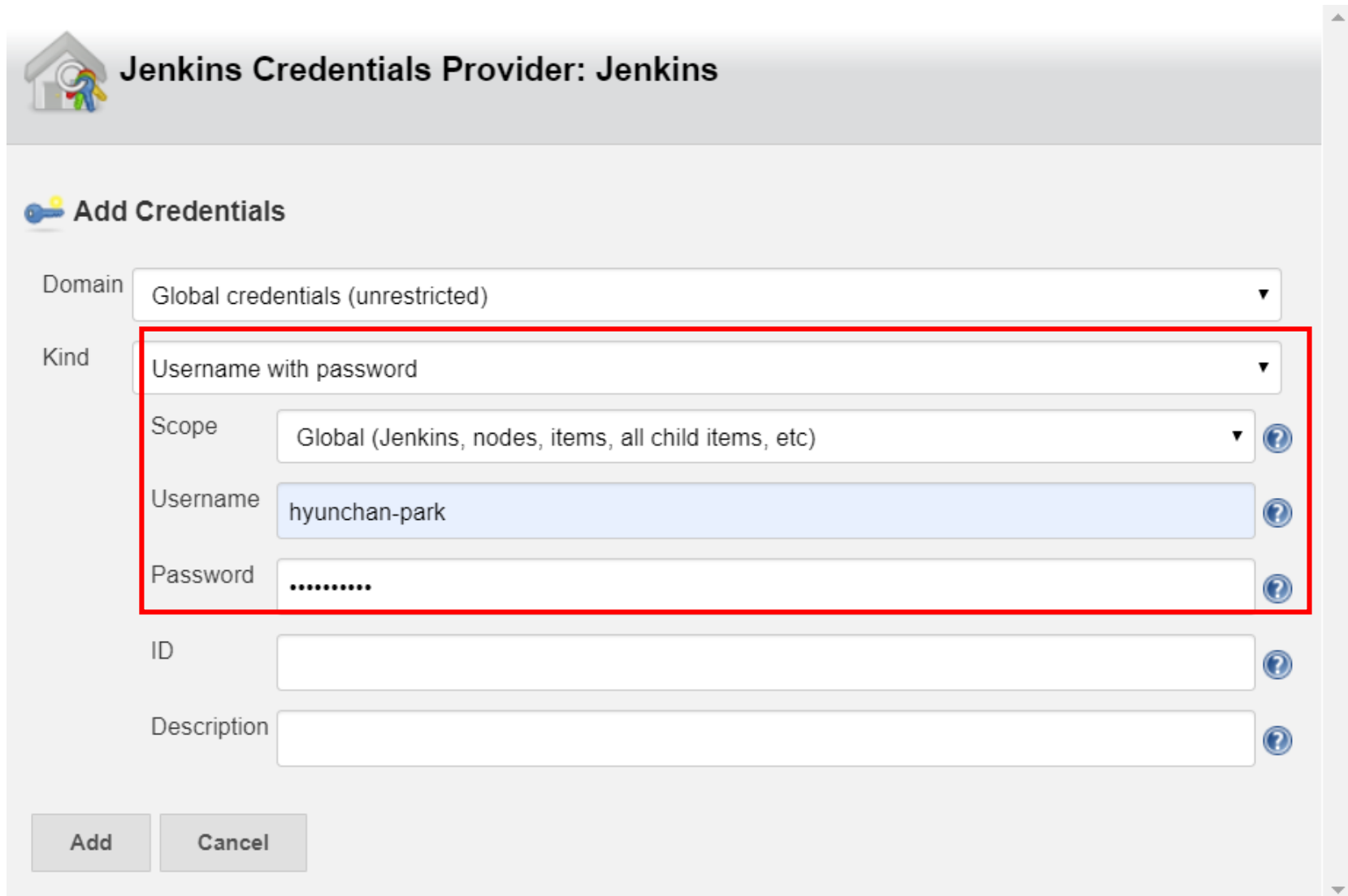
*/master

Add Branch

저장

Apply

Private Repo 인 경우, add credential 로 계정 정보를 추가해주어야 접속 가능



The image shows the Jenkins 'Add Credentials' configuration page. The title is 'Jenkins Credentials Provider: Jenkins'. Below the title is a section 'Add Credentials' with a key icon. The form contains several fields: 'Domain' (Global credentials (unrestricted)), 'Kind' (Username with password), 'Scope' (Global (Jenkins, nodes, items, all child items, etc)), 'Username' (hyunchan-park), 'Password' (masked with dots), 'ID' (empty), and 'Description' (empty). A red rectangle highlights the 'Kind', 'Scope', 'Username', and 'Password' fields. At the bottom are 'Add' and 'Cancel' buttons.

Jenkins Credentials Provider: Jenkins

Add Credentials

Domain: Global credentials (unrestricted)

Kind: Username with password

Scope: Global (Jenkins, nodes, items, all child items, etc)

Username: hyunchan-park

Password:

ID:

Description:

Add **Cancel**

Private repo 접근 성공

Repositories

Repository URL

https://github.com/hyunchan-park/chatbot

?

Credentials

hyunchan-park/*****

▼

Add

▼

Name

?

Refspec

?

Add Repository

Branches to build

Branch Specifier (blank for 'any')

*/master

?

Add Branch

Repository browser

(자동)

▼

?

Additional Behaviours

Add

▼

Build triggers (작업 유발)





빌드 유발

- ☐ 빌드를 원격으로 유발 (예: 스크립트 사용)
- ☐ Build after other projects are built
- ☐ Build periodically
- ☒ GitHub hook trigger for GITScm polling
- ☐ Poll SCM



빌드 환경

빌드 환경

- ☐ Delete workspace before build starts
- ☐ Use secret text(s) or file(s) 
- ☐ Send files or execute commands over SSH before the build starts 
- ☐ Send files or execute commands over SSH after the build runs 
- ☐ Abort the build if it's stuck
- ☐ Add timestamps to the Console Output
- ☐ Inspect build log for published Gradle build scans
- ☐ With Ant 

Build (실제 수행할 작업들)

* 일단 GitHub 와의 연동 테스트를 위해 간단한 메시지 출력만 수행

Build

 Execute shell

X

?

Command

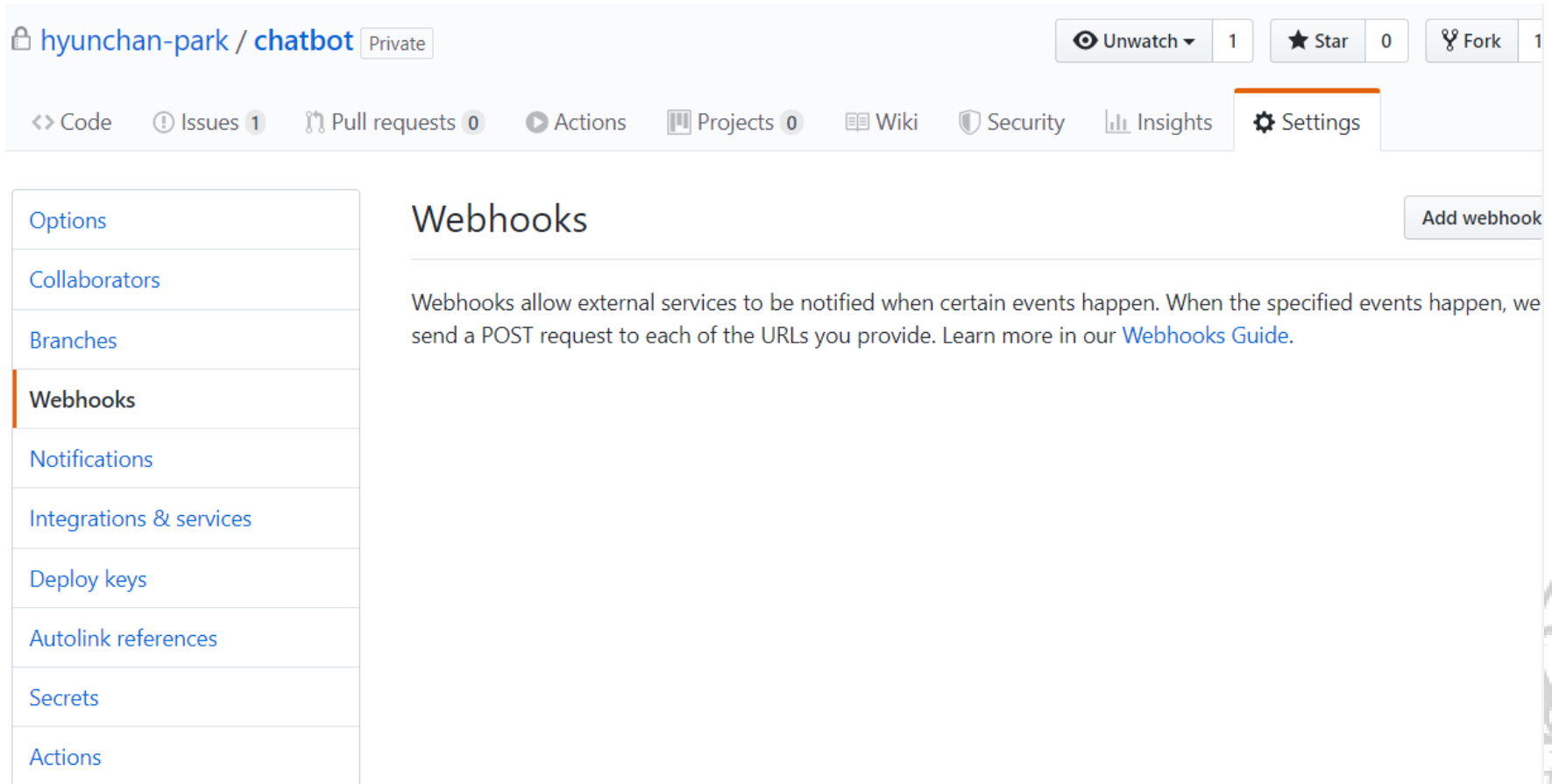
echo "빌드를 시작합니다!"

[See the list of available environment variables](#)

고급...

Add build step ▼

GitHub Webhook 을 통한 Jenkins 연동



The screenshot shows the GitHub repository settings page for 'hyunchan-park / chatbot'. The repository is marked as 'Private'. The top navigation bar includes links for Code, Issues (1), Pull requests (0), Actions, Projects (0), Wiki, Security, Insights, and Settings (which is the active tab). On the right, there are buttons for Unwatch (1), Star (0), and Fork (1). The left sidebar contains a list of settings categories: Options, Collaborators, Branches, Webhooks (highlighted with an orange bar), Notifications, Integrations & services, Deploy keys, Autolink references, Secrets, and Actions. The main content area is titled 'Webhooks' and includes an 'Add webhook' button. Below the title, a text block explains that webhooks allow external services to be notified when certain events happen, and that a POST request is sent to the provided URLs. A link to the 'Webhooks Guide' is also present.

hyunchan-park / chatbot Private

Unwatch 1 Star 0 Fork 1

Code Issues 1 Pull requests 0 Actions Projects 0 Wiki Security Insights Settings

Options Collaborators Branches **Webhooks** Notifications Integrations & services Deploy keys Autolink references Secrets Actions

Webhooks

Add webhook

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we send a POST request to each of the URLs you provide. Learn more in our [Webhooks Guide](#).

GitHub Webhook 등록

Webhooks / Add webhook

We'll send a POST request to the URL below with details of any subscribed events. You can choose the format you'd like to receive (JSON, x-www-form-urlencoded, etc). More information can be found in the [documentation](#).

Payload URL *

`http://203.254.143.211:17037/github-webhook/`

Content type

application/x-www-form-urlencoded

Secret

Which events would you like to trigger this webhook?

☒ Just the push event.

☐ Send me **everything**.

☐ Let me select individual events.


☒ **Active**

We will deliver event details when this hook is triggered.

Add webhook

간단히 PR 을 하나 만들어 테스트해보자

- GitHub 에서 Fork 한 후, PR 을 만들고 Merge 까지 수행
- Merge가 Push 이벤트로 인식되어, Webhook을 통해 Jenkins로 이벤트 전달
- 빌드가 수행됨을 알 수 있음
- 끝나고 나서 빌드 번호를 눌러보자



Jenkins

Chatbot CICD

대시보드로 돌아가기

상태

변경사항

작업공간

Build Now

Project 삭제

구성

GitHub Hook Log

Rename

Project Chatbot CICD

Jenkins Project for ChatBot

작업 공간

최근 변경사항

고정링크

Build History


find

#1

2019. 11. 17 오후 12:23

RSS (전체) RSS (실패)

빌드 정보 나옴 (파란 버튼은 성공을 뜻함)

 **Jenkins**

Jenkins > Chatbot CICD > #1

 프로젝트로 돌아가기

 상태

 바뀐점


 **Console Output**


 빌드 정보 수정

 Delete build '#1'

 Polling Log

 Git Build Data

 No Tags

 **빌드 #1 (2019. 11. 17 오후 12:2...** 2 n 소!

 No changes.

 [Started by GitHub push by hyunchan-park](#)

 **Revision:** 8b47dcb7580119f1962e9cc9ed994b0cdcbbbfba

- refs/remotes/origin/master

콘솔 출력

Started by GitHub push by hyunchan-park

Running as SYSTEM

Building in workspace /var/lib/jenkins/workspace/Chatbot CI/CD

using credential 7e5f5a05-3428-439b-a410-08b24146f976

Cloning the remote Git repository

Cloning repository <https://github.com/hyunchan-park/chatbot>

> git init /var/lib/jenkins/workspace/Chatbot CI/CD # timeout=10

Fetching upstream changes from <https://github.com/hyunchan-park/chatbot>

> git --version # timeout=10

using GIT_ASKPASS to set credentials

> git fetch --tags --progress -- <https://github.com/hyunchan-park/chatbot>

+refs/heads/*:refs/remotes/origin/* # timeout=10

> git config remote.origin.url <https://github.com/hyunchan-park/chatbot> # timeout=10

> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10

> git config remote.origin.url <https://github.com/hyunchan-park/chatbot> # timeout=10

Fetching upstream changes from <https://github.com/hyunchan-park/chatbot>

using GIT_ASKPASS to set credentials

> git fetch --tags --progress -- <https://github.com/hyunchan-park/chatbot>

+refs/heads/*:refs/remotes/origin/* # timeout=10

> git rev-parse refs/remotes/origin/master^{commit} # timeout=10

> git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10

Checking out Revision 8b47dcb7580119f1962e9cc9ed994b0cdcbbfba (refs/remotes/origin/master)

> git config core.sparsecheckout # timeout=10

> git checkout -f 8b47dcb7580119f1962e9cc9ed994b0cdcbbfba # timeout=10

Commit message: "Merge pull request #3 from HyunchanPark-Class/master"

First time build. Skipping changelog.

[Chatbot CI/CD] \$ /bin/sh -xe /tmp/jenkins8190082982857522543.sh

+ echo 빌드를 시작합니다!

빌드를 시작합니다!

Finished: SUCCESS


Workspace 가 만들어진 것을 확인


**Jenkins**

검색

Jenkins ▾ Chatbot CICD ▸

 대시보드로 돌아가기

 상태

 변경사항

 **작업공간**

 Build Now

 Project 삭제

 구성

 GitHub Hook Log

 Rename


 **Build History**

추이 ≡

x

 **#1**

2019. 11. 17 오후 12:23

 [RSS \(전체\)](#)  [RSS \(실패\)](#)

Project Chatbot CICD

Jenkins Project for ChatBot

 **작업 공간**

 [최근 변경사항](#)

고정링크

- [Last build,_\(#1\),5 min 4 sec 전](#)
- [Last stable build,_\(#1\),5 min 4 sec 전](#)
- [Last successful build,_\(#1\),5 min 4 sec 전](#)
- [Last completed build,_\(#1\),5 min 4 sec 전](#)

Git Repo 를 가져온 것을 확인할 수 있음

Workspace of Chatbot CICD on master



- [.git](#)
- [backup](#)
- [node_modules](#)
- [test](#)
- [v1](#)
- [v3](#)
- [v4](#)

.env	2019. 11. 17 오후 12:23:45	200 B	보기
.eslintrc.js	2019. 11. 17 오후 12:23:45	349 B	보기
food.js	2019. 11. 17 오후 12:23:45	182 B	보기
index.js	2019. 11. 17 오후 12:23:45	773 B	보기
movie.js	2019. 11. 17 오후 12:23:45	318 B	보기
package.json	2019. 11. 17 오후 12:23:45	621 B	보기
package-lock.json	2019. 11. 17 오후 12:23:45	106.46 KB	보기
README.md	2019. 11. 17 오후 12:23:45	10 B	보기

[\(zip 파일로 압축\)](#)

- 빌드를 위해 필요한 파일들은 모두 이 workspace 에서 관리됨
- 필요하면 항상 새로 만들 수도 있지만, 빌드 성능이 저하될 수 있음



Jenkins

Todo 1: Jenkins 서버 구축

Todo 2: Jenkins 작업 설정

Todo 3: Jenkins 빌드 설정



빌드 설정

- GitHub 연동 테스트는 했으니, 실제 빌드할 내용을 설정

1. Test (integration)

1. SSH 를 이용해 Git pull (이미 git clone 으로 환경이 구성되어 있음)
2. 이미 동작 중인 프로세스가 있다면 강제 종료
3. Mocha를 이용한 동작 테스트
4. 실패하면 중단

2. Deploy

1. SSH 를 이용해 Git pull (이미 git clone 으로 환경이 구성되어 있음)
2. 이미 동작 중인 프로세스가 있다면 강제 종료
3. `execute nodejs index.js`



이슈

- Deployment Server 에서 이미 동작 중인 챗봇 서버
 - 이미 deploy server 에서 챗봇 서버가 서비스 중이라는 점
 - 따라서 기존 챗봇들 외에 추가로 테스트를 위한 챗봇을 추가해야 함
 - (편의상, deploy server의 챗봇 서버는 동작을 멈추고 진행할 것)
- GitHub project 가 private 인 경우, 자동으로 pull 이 안됨
 - ID, PW를 입력해야 하기 때문
 - `$ git config credential.helper store`
 - `$ git config --global credential.helper 'cache --timeout 7200'`
 - 7200초 동안 다시 로그인할 필요 없음 (일주일: 604800초)
 - Private repo를 쓸 경우, 로그인 정보는 정기적으로 관리할 필요가 있음
 - `$ git pull` <- 여기서 한 번 로그인



빌드 설정: 빌드 단계 추가

Build

Execute shell X ?

Command `echo "빌드를 시작합니다!"`

See [the list of available environment variables](#)

고급...

Add build step ▼

- Execute Windows batch command
- Execute shell
- Invoke Ant
- Invoke Gradle script
- Invoke top-level Maven targets
- Run with timeout
- Send files or execute commands over SSH**
- Set build status to "pending" on GitHub commit

(빌드 설정) 0. 서비스 중지

Send files or execute commands over SSH

SSH Publishers

SSH Server

Name

고급...

Transfers

Transfer Set

Source files

Remove prefix

Remote directory

Exec command

All of the transfer fields (except for Exec timeout) support substitution of [Jenkins environment variables](#)

고급...

Add Transfer Set

(빌드 설정) 1. Test

Name

☐ Publish to other SSH servers if an error occurs

☒ Fail the build if an error occurs

☐ Always SSH from master

Remote directory

Exec command

```
cd /home/ubuntu/chatbot
git pull
killall nodejs
nodejs index.js &
cd test
nodejs test.js

if [ $? -eq 0 ];
then
    echo "통합 테스트 통과!"
    killall nodejs
    exit 0
else
    echo "통합 테스트 실패! 빌드가 취소되었습니다."
    killall nodejs
    exit 1
fi
```

아래 고급 옵션

* 마지막 fi 뒤에
꼭 enter 를 넣을 것

(빌드 설정) 1. Test

```
cd /home/ubuntu/chatbot
```

```
git pull
```

```
killall nodejs
```

```
nodejs index.js &
```

```
cd test
```

```
nodejs test.js
```

```
if [ $? -eq 0 ];
```

```
then
```

```
    echo "통합 테스트 통과!"
```

```
    killall nodejs
```

```
    exit 0
```

```
else
```

```
    echo "통합 테스트 실패! 빌드가  
    취소되었습니다."
```


```
    killall nodejs
```

```
    exit 1
```

```
fi
```

* 마지막 fi 뒤에
꼭 enter 를 넣을 것

빌드 설정 테스트

 대시보드로 돌아가기

 상태

 변경사항

 작업공간


 Build Now


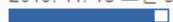







 Project 삭제



 구성

 GitHub Hook Log

 Rename


 Build History 추이

 #7	2019. 11. 18 오전 3:28		
 #6	2019. 11. 18 오전 3:26		
 #5	2019. 11. 18 오전 3:23		
 #4	2019. 11. 17 오후 2:09		
 #3	2019. 11. 17 오후 1:18		
 #2	2019. 11. 17 오후 1:14		
 #1	2019. 11. 17 오후 12:23		

 RSS (전체)  RSS (실패)

Project Chatbot CICD

Jenkins Project for ChatBot

 작업 공간

 최근 변경사항

고정링크

- [Last build_\(#6\), 2 min 18 sec 전](#)
- [Last stable build_\(#3\), 14 hr 전](#)
- [Last successful build_\(#6\), 2 min 18 sec 전](#)
- [Last unstable build_\(#6\), 2 min 18 sec 전](#)
- [Last unsuccessful build_\(#6\), 2 min 18 sec 전](#)
- [Last completed build_\(#6\), 2 min 18 sec 전](#)

빌드 설정 테스트



빌드 #8 (2019. 11. 18 오전 3:34:38)



No changes.



사용자 [admin](#)에 의해 시작됨



Revision: bed5b65358b9f55003fa43b0d5762f0a7d89d9a5

- refs/remotes/origin/master

```
+ echo 빌드를 시작합니다!
빌드를 시작합니다!
SSH: Connecting from host [hcpark]
SSH: Connecting with configuration [Integration Server] ...
SSH: EXEC: STDOUT/STDERR from command [cd /home/ubuntu/chatbot
git pull
killall nodejs
nodejs index.js &
cd test
nodejs test.js
killall nodejs
killall nodejs
if [ $? -eq 0 ];
then
    echo "통합 테스트 통과!"
    exit 0
else
    echo "통합 테스트 실패! 빌드가 취소되었습니다."
    exit 1
fi

] ...
Already up to date.
CQ2J7UM5F
보낸 메시지: 테스트를 시작합니다.
받은 메시지: 안녕하세요. 영화,밥,놀이 중에 말씀해주세요.
CQ2J7UM5F
영화를 추천합니다.
보낸 메시지: 영화
받은 메시지: 취향에 맞춘 영화를 추천해드릴게요.
CQ2J7UM5F
보낸 메시지: 밥
받은 메시지: 주변 맛집을 추천해드릴게요.
CQ2J7UM5F
보낸 메시지: 놀이
받은 메시지: 고만해.
테스트가 정상 종료되었습니다.
통합 테스트 통과!
SSH: EXEC: completed after 5,604 ms
SSH: Disconnecting configuration [Integration Server] ...
SSH: Transferred 0 file(s)
Build step 'Send files or execute commands over SSH' changed build result to SUCCESS
Finished: SUCCESS
```

(빌드 설정) 2. Deploy

The screenshot shows the Jenkins 'SSH Publishers' configuration page. The 'SSH Server' section has 'Name' set to 'Deployment server'. The 'Transfers' section has 'Transfer Set' selected. The 'Exec command' field contains a shell script. A red text box with a bullet point explains the 'daemonize' command. Another red text box explains the purpose of the 'fi' statement. A third red text box explains the purpose of the 'echo' command. A fourth red text box explains the purpose of the 'exit' command. A fifth red text box explains the purpose of the 'fi' statement.

SSH Publishers

SSH Server

Name: Deployment server

Transfers

Transfer Set

Source files

Remove prefix

Remote directory

Exec command

```
cd /home/ubuntu/chatbot
git pull
killall nodejs
daemonize -E BUILD_ID=dontKillMe nohup nodejs index.js &
if [ $? -eq 0 ]
then
    echo "배포 성공!"
    exit 0
else
    echo "배포 실패! 빌드가 취소되었습니다."
    exit 1
fi
```

Execute shell

Command: echo "빌드가 정상 종료되었습니다!"

See [the list of available environment variables](#)

• 이 옵션은 Jenkins 가 작업 종료 후, 작업에 사용된 프로세스를 모두 종료시키기 때문에, 이로 인해 챗봇 서버가 종료되는 것을 방지하기 위한 명령

* 마지막 fi 뒤에 꼭 enter 를 넣을 것

(빌드 설정) 2. Deploy

* Deploy serve에 daemonize 설치

```
ubuntu@hpcpark:~/chatbot$ sudo apt install daemonize
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  grub-pc-bin
Use 'sudo apt autoremove' to remove it.
The following NEW packages will be installed:
  daemonize
0 upgraded, 1 newly installed, 0 to remove and 113 not upgraded.
Need to get 11.3 kB of archives.
After this operation, 39.9 kB of additional disk space will be used.
Get:1 http://nova.clouds.archive.ubuntu.com/ubuntu bionic/universe amd64 daemonize amd64 1.7.7-1 [11.3 kB]
Fetched 11.3 kB in 1s (8479 B/s)
Selecting previously unselected package daemonize.
(Reading database ... 100964 files and directories currently installed.)
Preparing to unpack .../daemonize_1.7.7-1_amd64.deb ...
Unpacking daemonize (1.7.7-1) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
Setting up daemonize (1.7.7-1) ...
```

빌드 설정 테스트 완료!

 빌드 #31 (2019. 11. 18 오전 6:45:16)



No changes.



사용자 [admin](#)에 의해 시작됨



Revision: bed5b65358b9f55003fa43b0d5762f0a7d89d9a5

• refs/remotes/origin/master

```
SSH: Connecting from host [hcpark]
SSH: Connecting with configuration [Deployment server] ...
SSH: EXEC: STDOUT/STDERR from command [cd /home/ubuntu/chatbot
git pull
killall nodejs
daemonize -E BUILD_ID=dontKillMe nohup nodejs index.js &
if [ $? -eq 0 ]
then
    echo "배포 성공!"
    exit 0
else
    echo "배포 실패! 빌드가 취소되었습니다."
    exit 1
fi
] ...
Already up to date.
nodejs: no process found
배포 성공!
SSH: EXEC: completed after 2,002 ms
SSH: Disconnecting configuration [Deployment server] ...
SSH: Transferred 0 file(s)
[Chatbot CI/CD] $ /bin/sh -xe /tmp/jenkins5103415554827218481.sh
+ echo 빌드가 정상 종료되었습니다!
빌드가 정상 종료되었습니다!
Finished: SUCCESS
```


개인 과제 #12 : CI/CD 구축

- Jenkins 를 이용한 CI/CD 구축
 - 전체 빌드 설정 화면
 - GitHub에 간단한 PR 및 merge 진행 (완료된 PR 화면 캡처)
 - 간단한 메시지를 수정해서, slack 에서 확인할 수 있도록
 - 최종 빌드 결과 화면 (console output 에서 배포 성공 메시지 확인)
 - Slack 에서 새로운 서비스가 잘 배포되었음을 확인하는 화면
- 제출 기한:
 - 12/15 (일) 23:59 (firm deadline: 지각 제출 없음)

