

개발 프로세스 실습 1

Hyunchan, Park

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

Division of Computer Science and Engineering

Jeonbuk National University



참고자료

- <http://www.nextree.co.kr/p8574/>
 - Node.js
- <https://www.lesstif.com/pages/viewpage.action?pageId=14745703>
 - Curl
- <https://www.leafcats.com/215>
 - Jenkins by Docker (plugin 설치 실패)
- <https://kkensu.tistory.com/58>
 - Jenkins 수동 설치 ubuntu
- <https://yaboong.github.io/jenkins/2018/05/14/github-webhook-jenkins/>
 - GitHub-Jenkins 연동
- <https://subicura.com/2016/07/11/coding-convention.html>
 - Linter
- <https://proinlab.com/archives/1885>
 - 슬랙 챗봇
- <https://heropy.blog/2018/03/16/mocha/>
 - 모카 테스트

참고자료

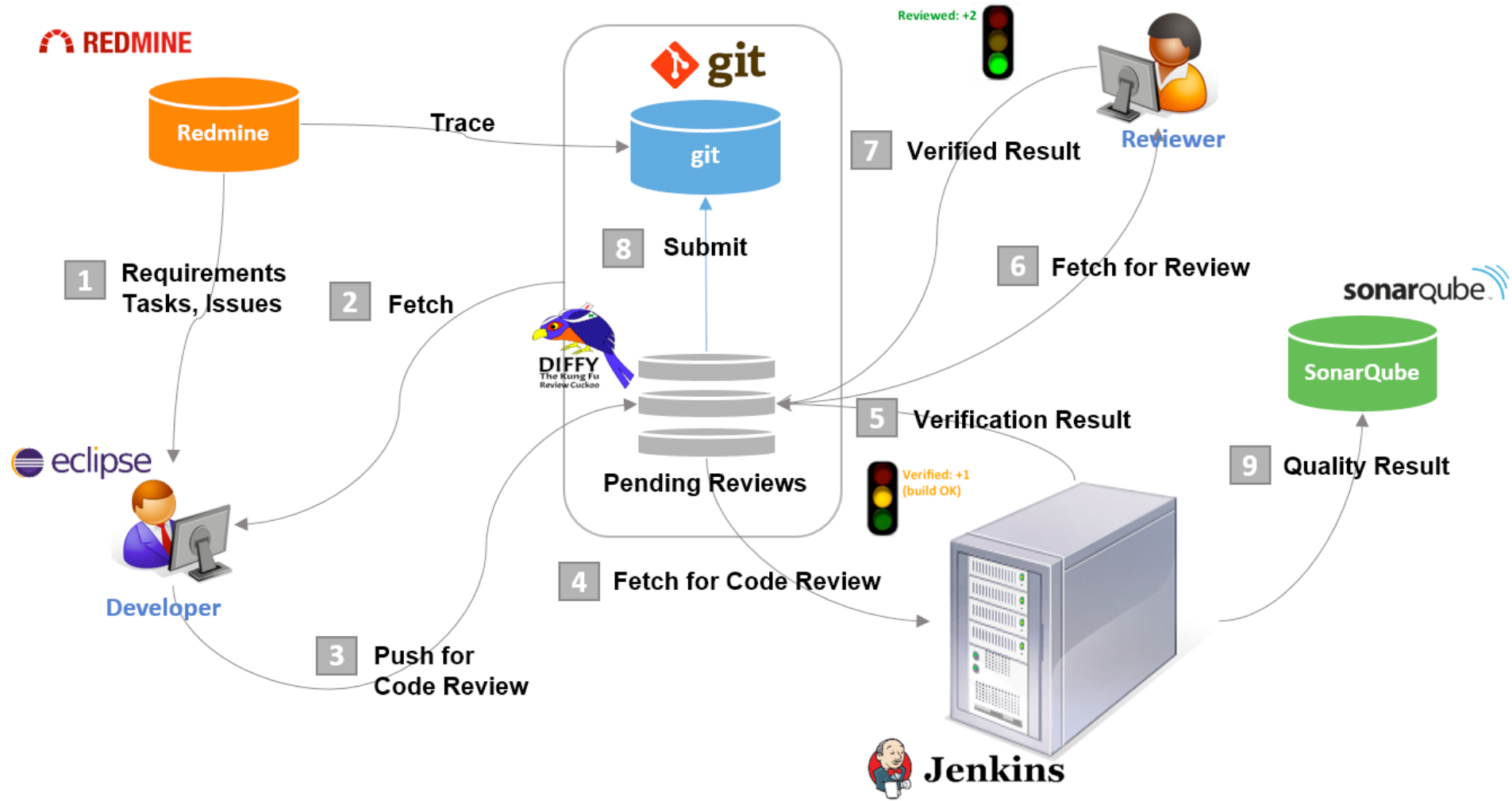
- <https://github.com/slackapi/node-slack-sdk>
 - Node-slack-sdk
- <https://bcho.tistory.com/759>
 - 조대협 개발자님 블로그: 개발 프로세스
- <https://bcho.tistory.com/1237>
 - Jenkins-GitHub 연동
- <https://kutar37.tistory.com/entry/Jenkins-Github-%EC%97%B0%EB%8F%99-%EC%9E%90%EB%8F%99%EB%B0%B0%ED%8F%AC-3>
 - Jenkins-GitHub 연동

Overview

목표: CI/CD의 이해

- CI/CD를 이해하려면?
 - 전체 개발과정의 이해가 필요
 - Integration, Deploy, Continuous ???
- 전체 개발 과정을 이해하려면?
 - 실제로 해봐야지~
- “Chatbot 개발” 을 통한 전체 개발 과정의 실습
 - 구성원: 관리자, 개발자, 테스터 (사실은 본인 1명)
 - 3대 서버 구축: 개발 시스템, Jenkins 시스템, 배포 시스템
 - 클라우드 서버 활용: J-Cloud
 - 활용: Slack, Node.js, ESLint, Mocha, GitHub webhooks, Jenkins 등

Typical ALM with Open Source



* ALM (Application Lifecycle Management)

Typical ALM Workflow (빌드 이전 단계)

1. 개발자는 Redmine에 작성되어 있는 요구사항, 업무, 이슈들을 이클립스의 작업 리스트에서 확인
2. 개발자는 자신의 업무와 관련 있는 소스 코드를 Git + Gerrit 으로부터 Fetch 받음
3. 개발자가 코딩과 Local Test를 마친 자신의 소스 코드를 리뷰 요청을 위해 Gerrit 에 Push 함
4. Jenkins는 빌드, 정적분석, 단위 테스트를 위해 Gerrit에 Push된 소스 코드를 Fetch 함
5. Jenkins는 확인 결과를 Gerrit에 등록
6. 리뷰어들은 개발자가 개발한 소스 코드의 리뷰를 수행
7. 리뷰어들은 검증 결과를 Gerrit에 등록
8. Gerrit은 코드 리뷰 결과와 빌드 검증 결과를 바탕으로 소스를 Git에 submit

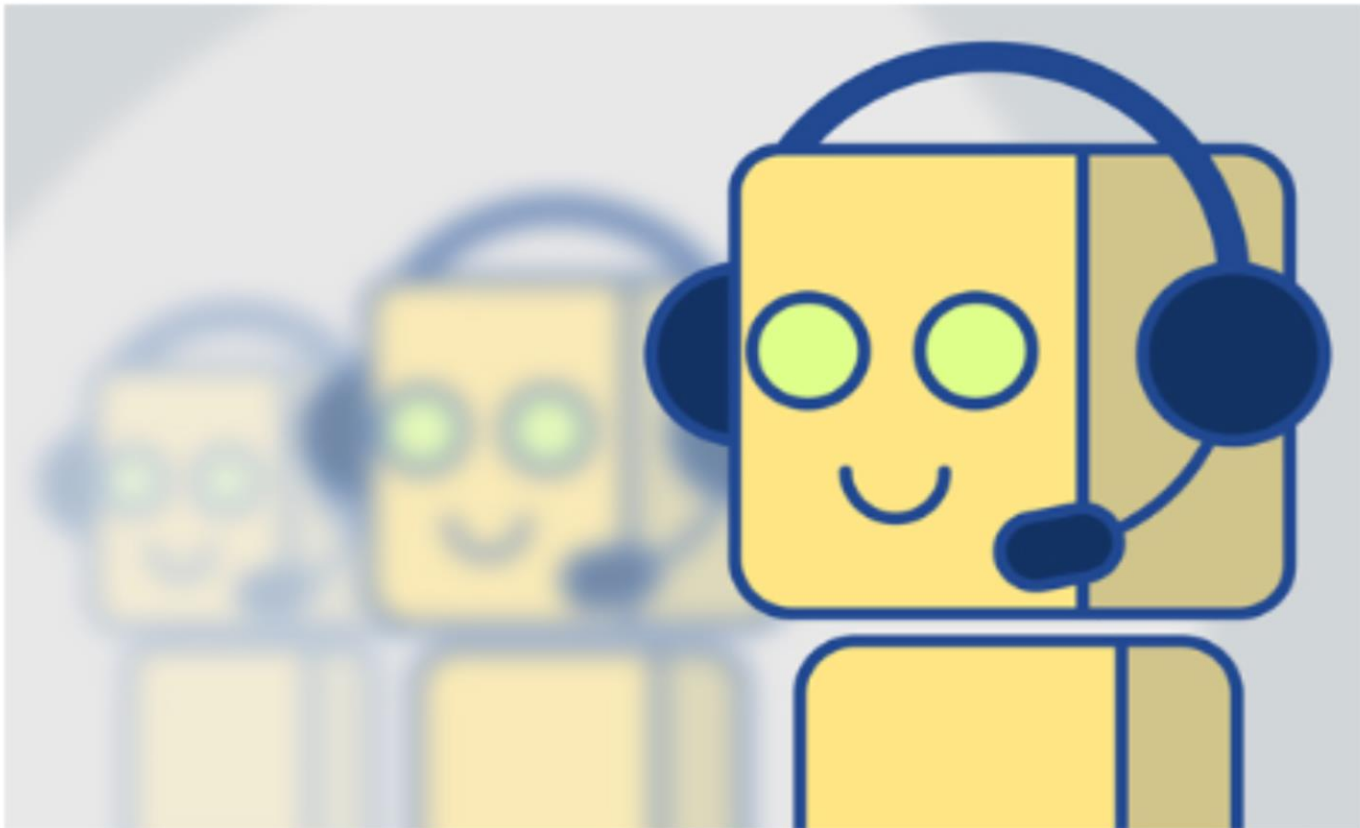
실습에서 진행해볼 개발 프로세스

1. Development
2. Pull Request
3. Code Convention Check
4. Unit Test
5. Code review
6. Integration (merge)
7. Build
8. Deploy to the Test Environment
9. Q/A: Quality Assurance (comprehensive test)
10. Deploy to the Service Environment

Chatbot

ChatBot

이제는 로봇에게 말하세요! 디지털 마케팅을 선도하는 '챗봇' 서비스
빅데이터 활용 등 개인 맞춤 서비스 제공에 탁월한 '챗봇' 서비스의 성장이 두드러져..



'챗봇' 서비스 / 출처 - COMPROVE 홈페이지

ChatBot: Briefing

[참고 1](#), [참고 2](#), [참고 3](#)

- 핵심

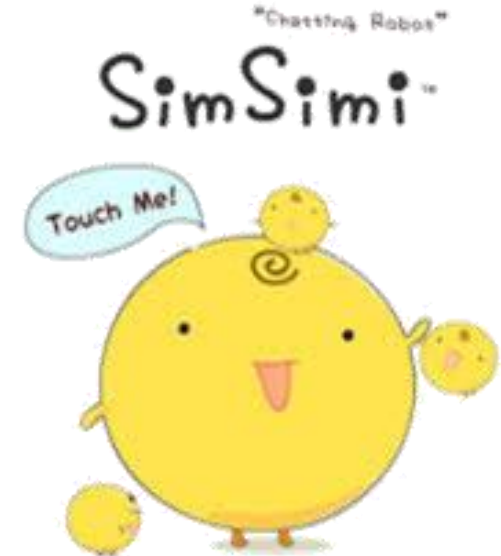
- 2016년 Mark Zuckerberg 가 처음 사용한 용어
- 카카오톡 등 메신저에 익숙한 사용자들을 위해, 서비스에 대화형 인터페이스를 도입
- 콜센터 등 상담 인력을 대체할 수단으로 각광을 받음

- Enabling Technique: AI

- 이러한 서비스는 기존에도 존재했지만 (심심이) 사용자에게 도움을 줄만큼 정확한 답변은 하지 못함
- AI 기술의 발전으로 답변의 정확도가 높아짐

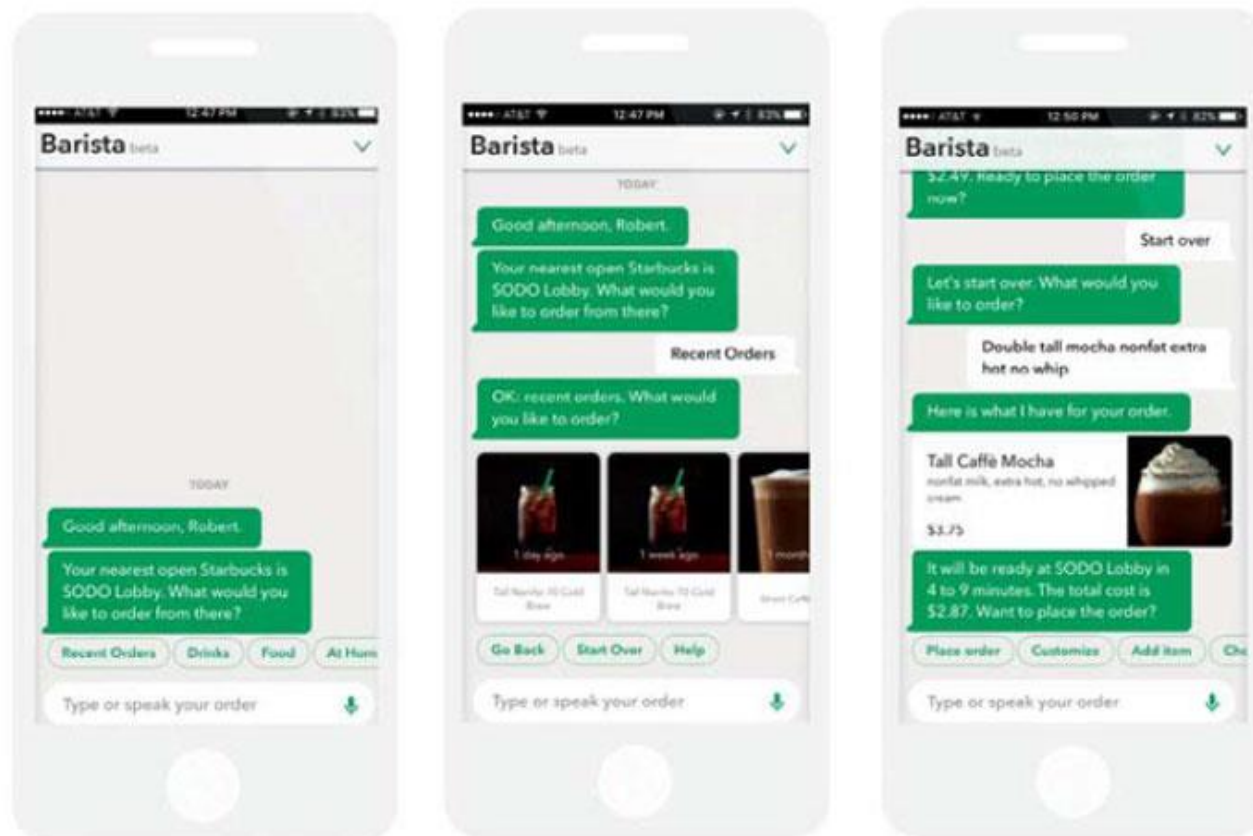
- 현재

- 거의 모든 서비스 업체에서 서비스를 출시하였거나, 준비 중
- 서비스의 영역도 확대 중 (상담 -> 예약, 주문 등 직접 제품을 판매)
- 코딩없이도 대화 모델만으로 챗봇을 만드는 서비스 다수 (Chatbot builder)
- 그래픽봇, 보이스봇으로 발전하는 중



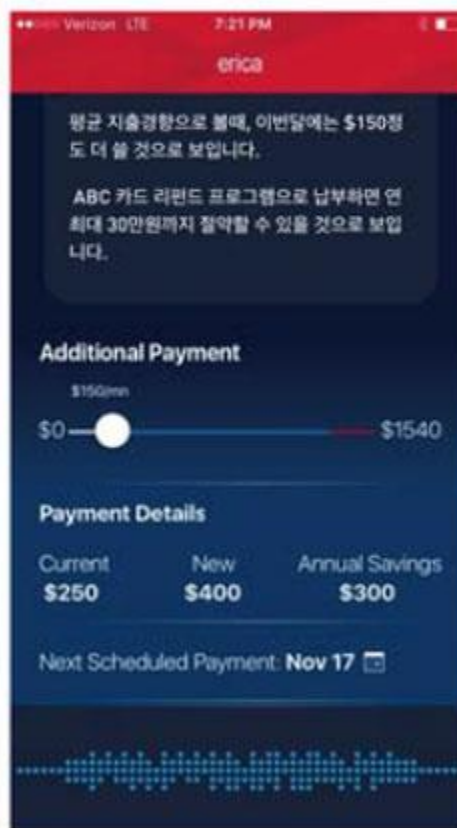
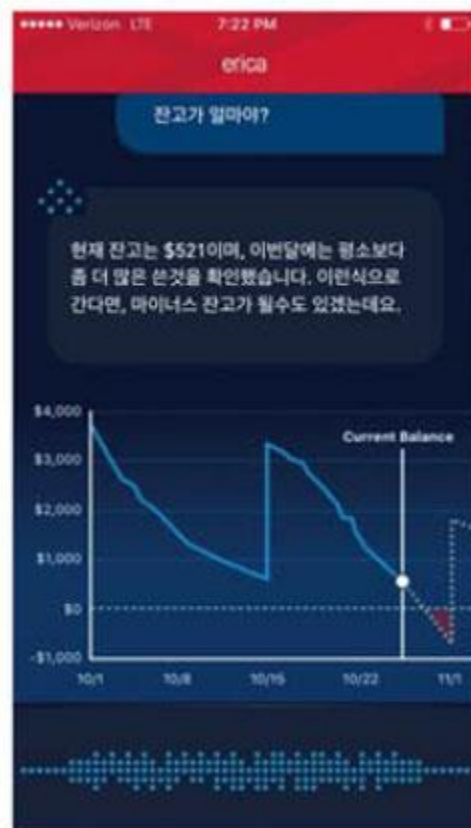
ChatBot: Cases

그림1 스타벅스의 브랜드컬러를 살린 '마이 스타벅스 바리스타' 챗봇



ChatBot: Cases

그림2 금융자문을 해주는 뱅크오브아메리카의 챗봇 '에리카'



ChatBot: Cases

그림4 답을 알려주진 않지만 답이 있는 곳을 알려주는
페덱스 챗봇

Speed up your international growth.
Reach your international customers earlier with premium services that deliver your urgent shipments intact and on time.
[Learn more >](#)

Quick Access

- Create a Shipment
- Ship Now with Credit Card
- Get Rates and Transit Times
- Schedule and Manage Pickups
- Access FedEx Delivery Manager®
- Order Shipping Supplies
- Pay your Bills Online
- Print Flyers, Posters and More

Track a Shipment
Enter up to 30 FedEx tracking numbers or FedEx Office order numbers

Find a Location

Ask FedEx

FedEx: Hello, I'm the FedEx Virtual Assistant. I'm here to help you with your questions regarding FedEx® services.

You: How much does it cost to send an item from Seoul to New York?

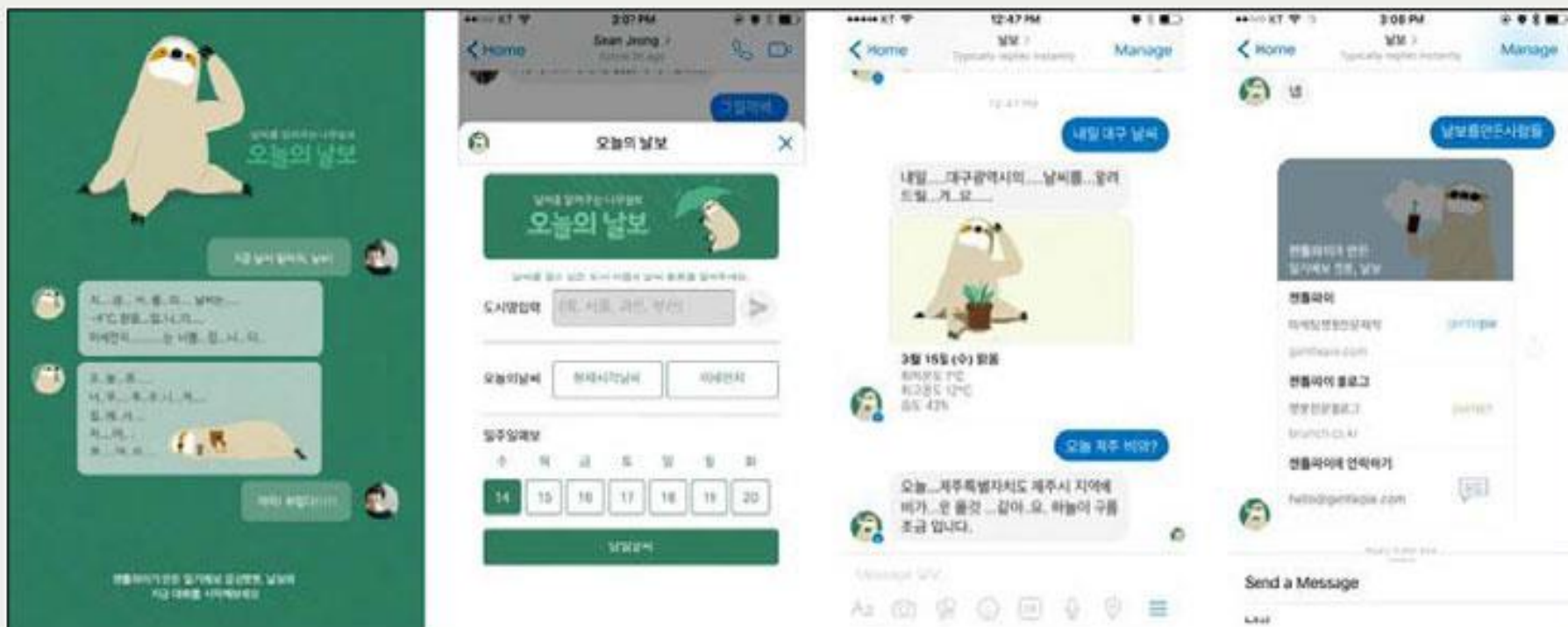
FedEx: The cost to ship a package depends on a few factors, such as the actual weight, dimensions, the service selected and the distance the package is being shipped. I can take you to a page where you can fill out this information and get an estimate or check transit times.

1. Go to [Get Rates & Transit Times](#).
2. Enter the From and To Postal/ZIP codes.
3. Enter the number of packages.
4. Enter the weight of the package.
5. To see discounts associated with your account, check the box **Ship using a FedEx account number**.

[Ask](#)

ChatBot: Cases

그림3 감성적인 측면을 강조한 날씨 챗봇 '날보'



ChatBot: Darkness

- 그래서 실제로 쓰는 사람?
 - 현실: 다양한 유인책으로 챗봇 사용을 유도하고 있지만, 특별한 성공 사례 없음
- 한계: “고객들은 사람을 원한다!”
 - 문장을 입력하고 있기가 불편하다.
 - 잘 못 알아듣는다. 여전히 실질적인 도움을 얻기엔 부족하다.
 - 실제 문제 상황은 다양하고, 복잡하다.
- 기술적 한계: 데이터 부족
 - AI가 학습할 대화 데이터가 부족함
- ChatBot 도입 시 유의할 점
 - 진짜 필요한가? : 도입 필요성의 검증
 - 어디에 필요한가? : 선택과 집중
 - 데이터는 충분한가? : 데이터 수집 및 학습, 테스트의 선행 필요





**PREPARATION
IS THE KEY**

Slack + Bots

Todo: 클라이언트 구성



업무용 메신저: 협업도구로서의 필요성

- 업무용 커뮤니케이션 도구
 - Long term (연 단위)
 - 보고서, 제안서 등 정형화된 문서
 - 예) dropbox에 올라간 워드, pdf 문서
 - Mid term (분기/월 단위)
 - 보다 자유로운 형태의 이슈, 계획 관리를 위한 문서
 - 예) GitHub milestone, 구글 캘린더
 - Short term (주/일 단위)
 - Daily, weekly 업무 진행 상황 확인
 - 예) 구글 메일, Docs, 캘린더
- and...real time!
 - Hourly, minutely, secondly 업무 확인



업무용 메신저: 협업도구로서의 필요성

- 왜 일반 메신저는 협업 도구 활용이 안 되는가?
 - 사적, 공적 메신저의 분리를 요구함
 - 짧은 메시지/파일 저장 기간
 - 제한적인 검색 기능
 - 타 협업도구와의 연계 부족
 - 보안성 부족
- 업무용 메신저 (팀 메신저, 사내 메신저)
 - 파일 공유: 대용량, 무제한 저장 기간
 - 이슈 별, 업무 별 대화방 및 관리자 기능 (퇴사자 접속 차단)
 - 타 협업도구와 자유로운 연계: 공개 API 지원
 - 강화된 보안성
 - 종류: Slack, Workplace at Facebook, 잔디, Company Talk, Meet Talk, Grap

Slack: 가장 유명

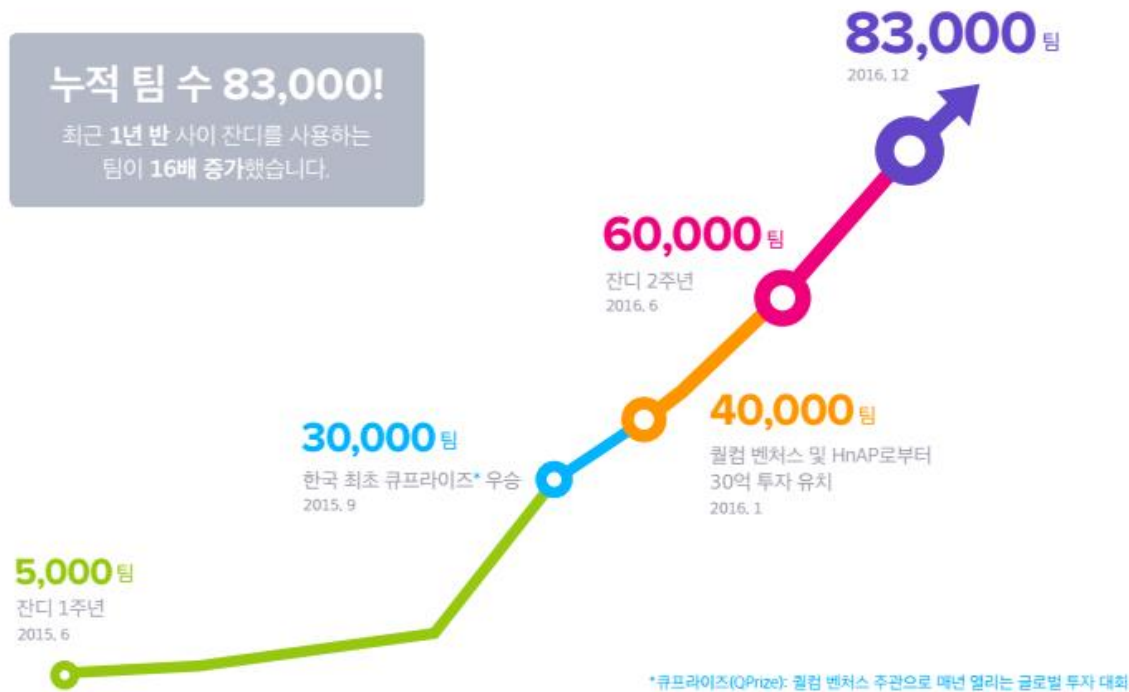


- Slack: Dominating group chat platform
- 세계적으로 가장 많이 쓰는 업무용 메신저
- Simple and powerful
 - 간단한 검색 기능
 - 다양한 장치에서 사용 가능: 웹, 모바일, 데스크탑
 - 간단하고 편리한 파일 공유
- 단점
 - 한글 지원: 한국어 검색, 대화방 개설 불가
 - 파일 검색 기능 제한
 - 느린 속도 (한국 사용자)
 - 팀 생성 제한 (이메일 인증 필요)



Alternative: 잔디

- 국내 스타트업 토스랩 개발
 - Slack의 국내 시장 진입 어려움을 파악하고, 한국형 협업 메신저 개발
 - 2014년 launch
 - <https://www.jandi.com/landing/kr/features>






Sign In

New to Slack? We'll help you [get started](#).

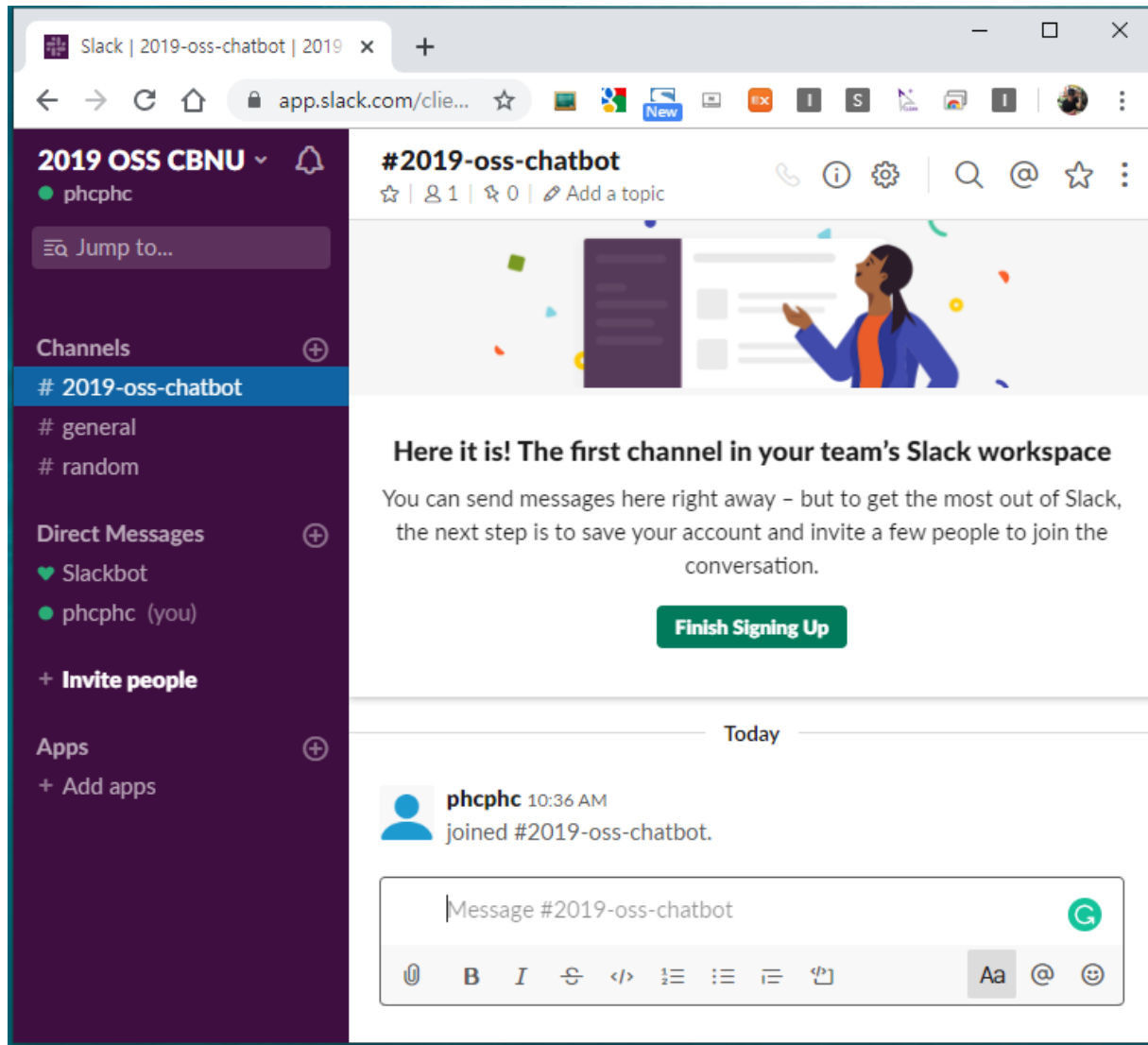


Slack 신규 가입

- Get started
- E-mail 계정 입력
-  My team isn't using Slack yet
Create a Slack workspace for your team. >
- E-mail verification (메일 확인하여 6자리 숫자 입력)
- 회사 이름: “2019 OSS CBNU” (각자 임의로 입력)
- 프로젝트 이름: “OSS ChatBot” (각자 임의로 입력)
- 다른 사람 초대: skip now
- 작업 분류 지정: IT (tool은 사용 안 함)
- Tada! (완료 및 생성된 채널로 이동)



기본 화면

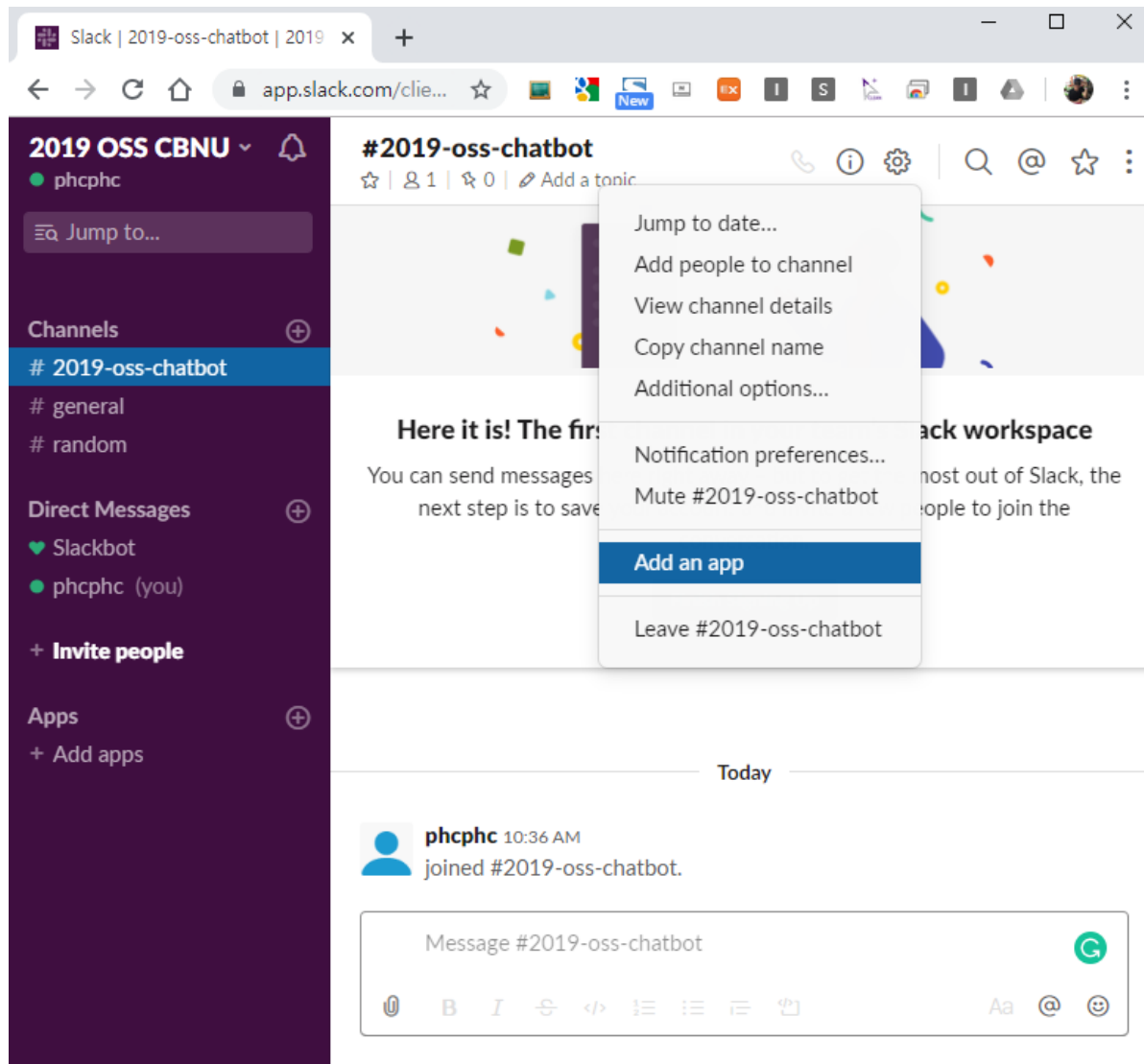


기본 기능

- Workspace: 팀 단위 협업 공간
- Channels: 대화방
 - 다양한 용도의 대화방을 만들어 사용 가능
 - 채널 별로 주고받은 파일 목록을 따로 확인 가능
 - Public and Private
- Direct Messages: 개인간 대화
- Apps: Plug-ins
- 특징
 - 모든 대화창이 Web-based 로 관리되어,
 - 다양한 API를 통해 접근, 제어, 관리가 가능함



Bots: 설치



Bots: 설치

Add apps to # 2019-oss-chatbot

[View App Directory](#)

From the App Directory



Bots

Connect a bot to the Slack Real Time Messaging API.

[Install](#)

Bots

[Add to Slack](#)

Bots: 설치

Username

Start by choosing a username for your bot

Username must be all lowercase. They cannot be longer than 21 characters and can only contain letters, numbers, periods, hyphens, and underscores. Most people choose to use their first name, last name, nickname, or some combination of those with initials.

Add bot integration

By creating a bot integration, you agree to the [Slack API Terms of Service](#).

Bots: token 가져오기

Integration Settings

API Token

The library you are using will want an API token for your bot.

* 주의: API token 은 노출되면 위험함.

xoxb-824792789584-813333340835-dqOxBFEBKzl

[Regenerate](#)



Be careful when sharing bot user tokens with applications. Do not publish bot user tokens in public code repositories. [Review token safety tips.](#)

Save Integration

Bots: 설정 종료

2019 OSS CBNU

Hyunchan Park

Jump to...

Channels

2019-oss-chatbot

general

random

+ Add a channel

Direct Messages

Slackbot

Hyunchan Park (you)

+ Invite people

Apps

noticebot

+ Add apps

noticebot

Messages About

Bring your team into Slack

Slack is better with teammates – invite them to start collaborating.

Add People

Today

Hyunchan Park 10:56 AM
Hello

Message noticebot

Aa @ 😊

* 색이 없는 동그라미는 현재 동작 중이 아님을 뜻

Node.js and npm

Todo 1: 클라우드 서버 구성

Todo 2: 챗봇 서버 구축



1. J-Cloud 에서 우분투 가상머신 (인스턴스) 생성 및 접속





JCloud는 전북대학교 컴퓨터공학부에서 제공하는
교육 및 연구용 클라우드 서비스입니다.

전북대 학생들로 구성된 J-DevOps 팀이 OpenStack을 기반으로 구축하였고,
교육 및 연구 용도에 맞춰 필요한 사항을 수정하며 운영합니다.

2018년 2월 현재, 시험 운영 중이며 아래 두 가지 서비스를 제공 중입니다.

(로고를 클릭하면 해당 서비스로 이동합니다.)

J-Cloud



JCloud 컴퓨팅 서비스

클라우드 상의 가상 머신, 가상 네트워크, 가상 스토리지 등의 서비스를 제공합니다.

학생들은 수업 및 개인 학습 용도로 활용할 수 있습니다.



Web IDE 서비스

웹 상에서 프로그램 개발 및 테스트를 수행할 수 있는 Web IDE 서비스를 제공합니다.

C,C++,Java,Python,.Net,PHP,Node.js 등 다양한 언어를 지원합니다.
본 서비스는 JCloud 를 기반으로 동작하고 있습니다.

사용 방법

- J-Cloud 시스템: <http://jcloud.jbnu.ac.kr>
- 소개 및 사용 방법: <https://jcloud-devops.github.io/>
 - Document 에서 아래 참조
 - 인스턴스 생성 방법
 - SSH 접속 Xshell, putty 등 사용
- OSS 수강생들의 계정은 모두 생성해놓은 상태
 - ID: oss학번, PW: 학번
 - 예) oss41983//41983
- 문의: jcloud@jbnu.ac.kr
- 참고: 학기 종료 직후, 인스턴스 및 계정 삭제됨
 - 컴퓨터공학과 학생들은 학번으로 계정이 생성되어 있음
 - 타 학과 학생들은 별도 신청하여 사용 가능



OS 수업용 인스턴스(가상머신) 만들기

- Details
 - Instance Name: ID 와 동일하게 oss학번
 - Count: 1 (1인당 1대만 만들어서 사용할 것)
 - Availability Zone: “Any Availability Zone”
- Source: Image, “Ubuntu 18.04.2 ssh7777” 선택
 - Create New Volume: 아니오
- Flavor: t3.small
- Network: oss2019.internal
- Key Pair: 미리 생성해서 꼭 다운로드 받고, 파일 잘 관리할 것
 - J-Cloud 상의 가상머신에 접속할 때 사용하는 인증파일
- 그 외 기본 설정 사용

J-Cloud 에서 인스턴스 생성

- 인스턴스 생성 후 화면
 - 본인의 인스턴스에 대해 IP 확인하고, SSH 접속하여 다음 내용들을 진행할 것

JCloud 2019oss hcpark

Project / Compute / Instances

Instances

Instance ID = Filter [Launch Instance](#) [Delete Instances](#)

Displaying 1 item

<input type="checkbox"/>	Instance Name	Image Name	IP Address	Flavor	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Ac
<input type="checkbox"/>	hcpark-test	Ubuntu 18.04.2 LTS (HVM) - 7777	192.168.0.8	t3.small	hcpark-jcloud-rocky	Active	nova	None	Running	1월, 23시간	

Displaying 1 item

SSH 클라이언트를 이용한 접근

- <https://jcloud-devops.github.io/user-guide.html>
 - SSH 연결 방법을 확인할 것
 - 공용 IP: 203.254.143.211
 - SSH 포트 번호: 19xxx
 - xxx는 본인 인스턴스 IP의 끝자리. 예) 192.168.0.2 → 19001, 192.168.0.199 → 19199
 - Port forwardings: 18xxx -> 80, 17xxx -> 8080 (HTTP와 Jenkins 용 8080)
- SSH 클라이언트
 - SSH 프로토콜을 이용해 원격 서버에 접속하는 프로그램
 - PuTTY 등이 유명함: 단순하고 기본 기능 충실. 그러나 너무 단순함
- Xshell
 - 개인용, 교육용 무료 버전 배포
 - http://www.netsarang.co.kr/download/free_license.html
- (자유롭게 본인에 맞는 프로그램 사용하는데...Xshell 쓰세요)

Node.js and npm

Todo 1: 클라우드 서버 구성

Todo 2: 챗봇 서버 구축



Node.js

<https://ko.wikipedia.org/wiki/Node.js>

- 확장성 있는 네트워크 애플리케이션 개발에 사용되는 소프트웨어 플랫폼
 - 특히 손쉽게 백엔드 시스템 구현이 가능하여 널리 사용됨
- 작성 언어로 [자바스크립트](#)를 활용하며
 - Non-blocking I/O와 단일 스레드 이벤트 루프를 통한 높은 처리 성능 제공
- 내장 HTTP 서버 라이브러리를 포함하고 있어 웹 서버에서 아파치 등의 별도의 소프트웨어 없이 동작하는 것이 가능
 - 이를 통해 웹 서버의 동작에 있어 더 많은 통제가 가능
- v8 (자바스크립트 엔진)으로 빌드 된 이벤트 기반 자바스크립트 런타임
- 위키에 간단한 예제들이 있으니 확인해 볼 것
- (간단하게, 웹서버 만들 때 쓰는 손쉬운 도구)
 - 챗봇, 슬랙 연동 등의 예제도 많이 나옴

예제

Hello world HTTP 서버:

```
var http = require('http');

http.createServer(function (request, response) {
  response.writeHead(200, {'Content-Type': 'text/plain'});
  response.end('Hello World\n');
}).listen(8000);

console.log('Server running at http://localhost:8000/');
```

관련 도구 설치

- `sudo apt-get update`
 - Error 발생 시, 재부팅 후 다시 시도
- `sudo apt-get install nodejs npm`
 - 5분 이상 걸림
- `sudo npm install @slack/rtm-api dotenv`



Chatbot v1

```
const { RTMClient } = require('@slack/rtm-api');
var token = 'xoxb-824792789584-813333340835-dq0xBFER''', GYaVwe';

var rtm = new RTMClient(token);
rtm.start();

rtm.on('message', function (message) {
  var channel = message.channel;
  var text = message.text;

  if (text == '안녕') {
    rtm.sendMessage('헬로', channel);
  } else {
    rtm.sendMessage('응?', channel);
  }
});
```

챗봇 서버 실행

```
ubuntu@hcpark:~$ nodejs index.js
```

2019 OSS CBNU

Hyunchan Park

Jump to...

Channels

2019-oss-chatbot

general

random

+ Add a channel

Direct Messages

Slackbot

Hyunchan Park (you)

+ Invite people

Apps

noticebot

+ Add apps

Hyunchan Park (you)

active



Bring your team into Slack

Slack is better with teammates – invite them to start collaborating.

Add People



Hyunchan Park

This is your space. Draft messages, list your to-dos, or keep links and files handy. You can also talk to yourself here, but please bear in mind you'll have to supply both sides of the conversation.

Got something down



B

I



</>

≡

≡

≡



Aa

@



* Bot이 연동되어,
초록색 불이 들어옴

테스트

+ Add a channel

Direct Messages (+)





♥ Slackbot

👤 Hyunchan Park (you)

+ Invite people

Apps (+)

○ noticebot

-  **Hyunchan Park** 11:06 PM
안녕
-  **noticebot** APP 11:06 PM
헬로
-  **Hyunchan Park** 11:06 PM
ㅎㅎ
-  **noticebot** APP 11:06 PM
응?

Chatbot v2

```
require('dotenv').config();
const { RTMClient } = require('@slack/rtm-api');

var token = process.env.SLACK_TOKEN;

var rtm = new RTMClient(token);
rtm.start();

rtm.on('message', function (message) {
  var channel = message.channel;
  var text = message.text;

  switch (text) {
    case '영 화 ':
      rtm.sendMessage('취 향 에 맞 춘 영 화 를 추 권 해 드 릋 게 요 .', channel);
      break;
    case '밥 ':
      rtm.sendMessage('주 변 맛 집 을 추 권 해 드 릋 게 요 .', channel);
      break;
    case '놀 이 ':
      rtm.sendMessage('고 만 해 .', channel);
      break;
    default:
      rtm.sendMessage('안 념 하 세 요 . 영 화 , 밥 , 놀 이 중 에 말 씀 해 주 세 요 .', channel);
  }
});
```

ubuntu@hccpark:~/chatbot/v2\$ vi .env

SLACK_TOKEN='xoxb-824792789584-813333340835'

* 보안상 위험하므로, Token 정보는 .env 파일에 따로 관리하고, GitHub 에 올리지 않음

테스트

general
random
+ Add a channel

Direct Messages (+)









♥ Slackbot
● Hyunchan Park (you)

+ Invite people

Apps (+)

● noticebot

+ Add apps

-  **Hyunchan Park** 11:32 AM
영화
-  **noticebot** APP 11:32 AM
영화를 추천해드릴게요.
-  **Hyunchan Park** 11:32 AM
밥
-  **noticebot** APP 11:32 AM
주변 맛집을 추천해드릴게요.
-  **Hyunchan Park** 11:32 AM
놀이
-  **noticebot** APP 11:32 AM
고만해.
-  **Hyunchan Park** 11:32 AM
하하
-  **noticebot** APP 11:32 AM
안녕하세요. 영화,밥,놀이 중에 말씀해주세요.

Message noticebot 

 **B** *I*       **Aa**  

To Be an Open Source Project!

- 챗봇 기본 틀 완성
- 챗봇에 기능들을 oss화 해서, 팀원을 모집하려고 함
 - 영화 기능
 - 취향을 분석해서 상영 중인 영화를 추천하는 기능
 - 밥 기능
 - 위치를 파악해서 주변 맛집을 추천하는 기능
 - 위 기능들을 파일 별로 모듈화한 v3을 작성
- GitHub 에 올리고, 이슈들을 만들어 관리하자
 - 이슈 1. 영화 기능 -> 팀원1
 - 이슈 2. 밥 기능 -> 팀원2
 - (나는 제1개발자로서 index.js 에 대한 개발을 수행하고, 관리자로서 전체 프로젝트를 관리함)

Chatbot v3: Three files

index.js

```
require('dotenv').config();
const { RTMClient } = require('@slack/rtm-api');

var token = process.env.SLACK_TOKEN;

var rtm = new RTMClient(token);
rtm.start();

var food = require('./food');
var movie = require('./movie');

rtm.on('message', function (message) {
  var channel = message.channel;
  var text = message.text;

  switch (text) {
    case '영화':
      movie(rtm, channel);
      break;
    case '밥':
      food(rtm, channel);
      break;
    case '놀이':
      rtm.sendMessage('고만해 .', channel);
      break;
    default:
      rtm.sendMessage('안녕하세요 . 영화 , 밥 , 놀이 중에 말씀해주세요 .', channel);
  }
});
```

movie.js

```
var movie = function(rtm, channel) {
  console.log('영화를 추천합니다 .');
  rtm.sendMessage('취향에 맞춘 영화를 추천해드릴게요 .', channel);
}

module.exports = movie;
```

food.js

```
var food = function(rtm, channel) {
  console.log('밥집을 추천합니다 .');
  rtm.sendMessage('주변 맛집을 추천해드릴게요 .', channel);
}

module.exports = food;
```

다시 테스트



Hyunchan Park 11:21 PM

영화

영화



noticebot APP 11:21 PM

취향에 맞춘 영화를 추천해드릴게요.



Hyunchan Park 11:21 PM

놀이



noticebot APP 11:21 PM

고만해.



Hyunchan Park 11:21 PM

밥



noticebot APP 11:21 PM

주변 맛집을 추천해드릴게요.



Hyunchan Park 11:21 PM

하하



noticebot APP 11:21 PM

안녕하세요. 영화,밥,놀이 중에 말씀해주세요.

