

# Continuous Integration

201214262 라가영, 201313250 서지혁

# Index

- CI (CTIP)
- Git
- GitLab
- GitLab CI
- References

# CI

- Continuous Integration, 지속적 통합
- 지속적으로 QC을 적용하는 프로세스를 실행하는 것
- 초기에 “통합의 지옥”의 함정을 피함
- CTIP = CI + 테스트
- 최근에는 CI가 테스트의 의미도 내포하고 있음

# CI 달성을 위한 기반

- 형상 관리
- 빌드 자동화
- 테스트 자동화
- 배포 자동화
- 결과 총람 (테스트 커버리지 변동 사항, 회귀 테스트 등)
- ...

# Version Control

- 형상 관리, 버전 관리, 소스 관리...
- 문서(코드, 책, 그림...)에 대한 변경사항을 관리하는 시스템
- 각각의 장단점이 있음
  - SVN: ACL, 파일 락 등 접근 관리 우수. 대용량 파일 저장 일원화.
  - Git: 분산형. 데이터 손상에 강하다. 저장소를 통째로 받아야 한다.
  - ...
- 필요에 맞는 것을 쓰자!

# Git

- 분산형 버전 관리 시스템
- 리눅스 커널 소스 트리를 관리하기 위해 만들어짐
- Git은 몰라도 GitHub은 안다
- GitHub 붐으로 인해 많은 사람들이 알고 사용하게 됨

# GitLab

- 웹 기반 Git 저장소, 마일스톤, 이슈, 위키 통합 관리 도구
- 무료 커뮤니티 에디션 (CE) 제공
- CI 프로세스를 위한 GitLab CI 내장

# 배포 환경

- Amazon Web Service
  - Seoul
  - EC2 (Ubuntu 14.04 LTS)
- Let's Encrypt



# GitLab 설치하기

```
ubuntu ~$ curl -sS https://packages.gitlab.com/install/  
repositories/gitlab/gitlab-ce/script.deb.sh | sudo bash
```

```
ubuntu ~$ sudo apt-get install gitlab-ce
```

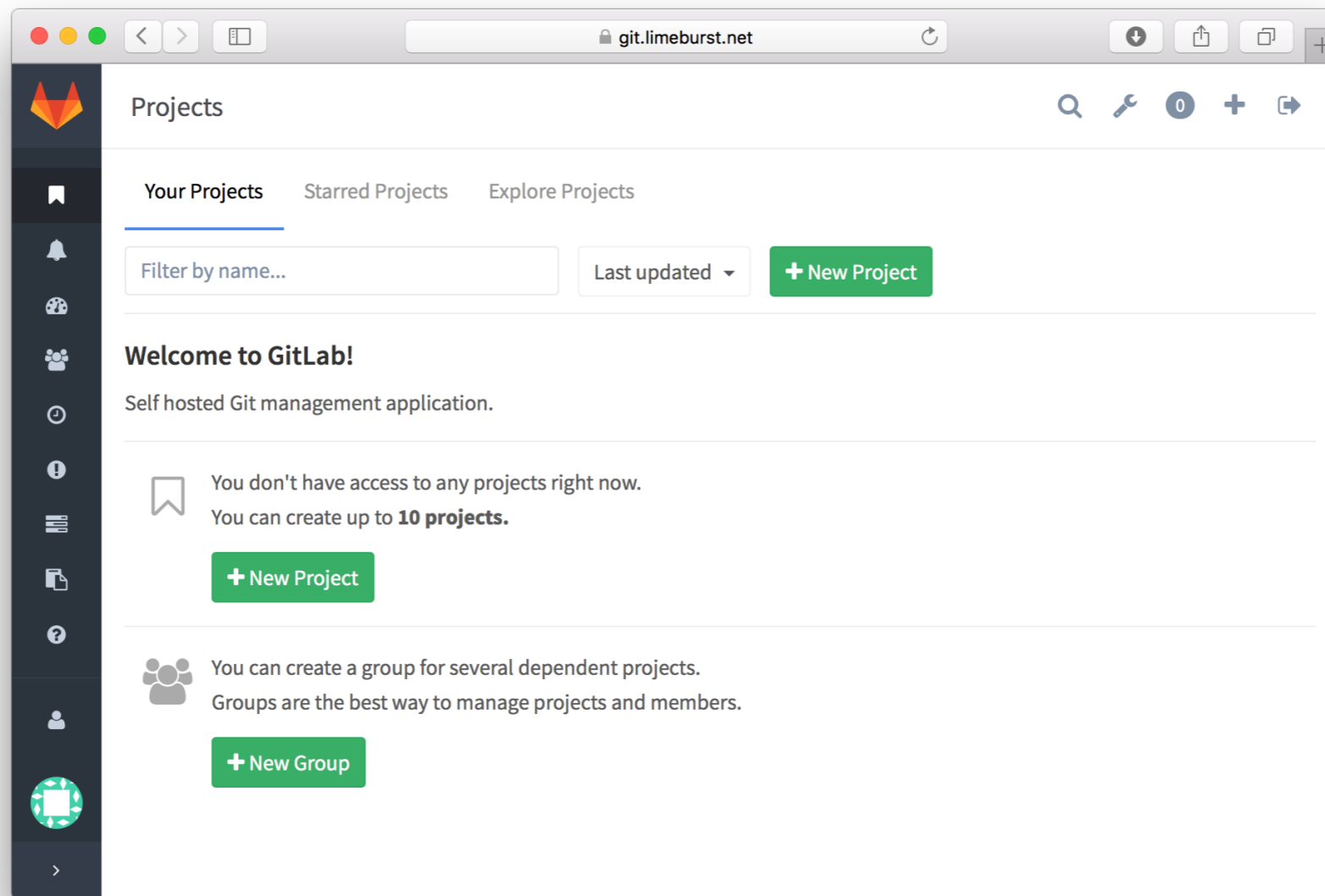
```
ubuntu ~$ sudo gitlabctl reconfigure
```

```
limeburst — ubuntu@ip-172-31-23-204: ~ — ssh -i .ssh/gitlab.pem ubuntu@5...
- execute the ruby block reload postgresql svlogd configuration
Recipe: gitlab::unicorn
* ruby_block[reload unicorn svlogd configuration] action create
  - execute the ruby block reload unicorn svlogd configuration
Recipe: gitlab::sidekiq
* ruby_block[reload sidekiq svlogd configuration] action create
  - execute the ruby block reload sidekiq svlogd configuration
Recipe: gitlab::gitlab-workhorse
* service[gitlab-workhorse] action restart
  - restart service service[gitlab-workhorse]
* ruby_block[reload gitlab-workhorse svlogd configuration] action create
  - execute the ruby block reload gitlab-workhorse svlogd configuration
Recipe: gitlab::nginx
* ruby_block[reload nginx svlogd configuration] action create
  - execute the ruby block reload nginx svlogd configuration
Recipe: gitlab::logrotate
* ruby_block[reload logrotate svlogd configuration] action create
  - execute the ruby block reload logrotate svlogd configuration

Running handlers:
Running handlers complete
Chef Client finished, 220/301 resources updated in 58 seconds
gitlab Reconfigured!
ubuntu@ip-172-31-23-204:~$
```

# GitLab 설정 화면

```
$ sudo gitlab-ctl reconfigure
```



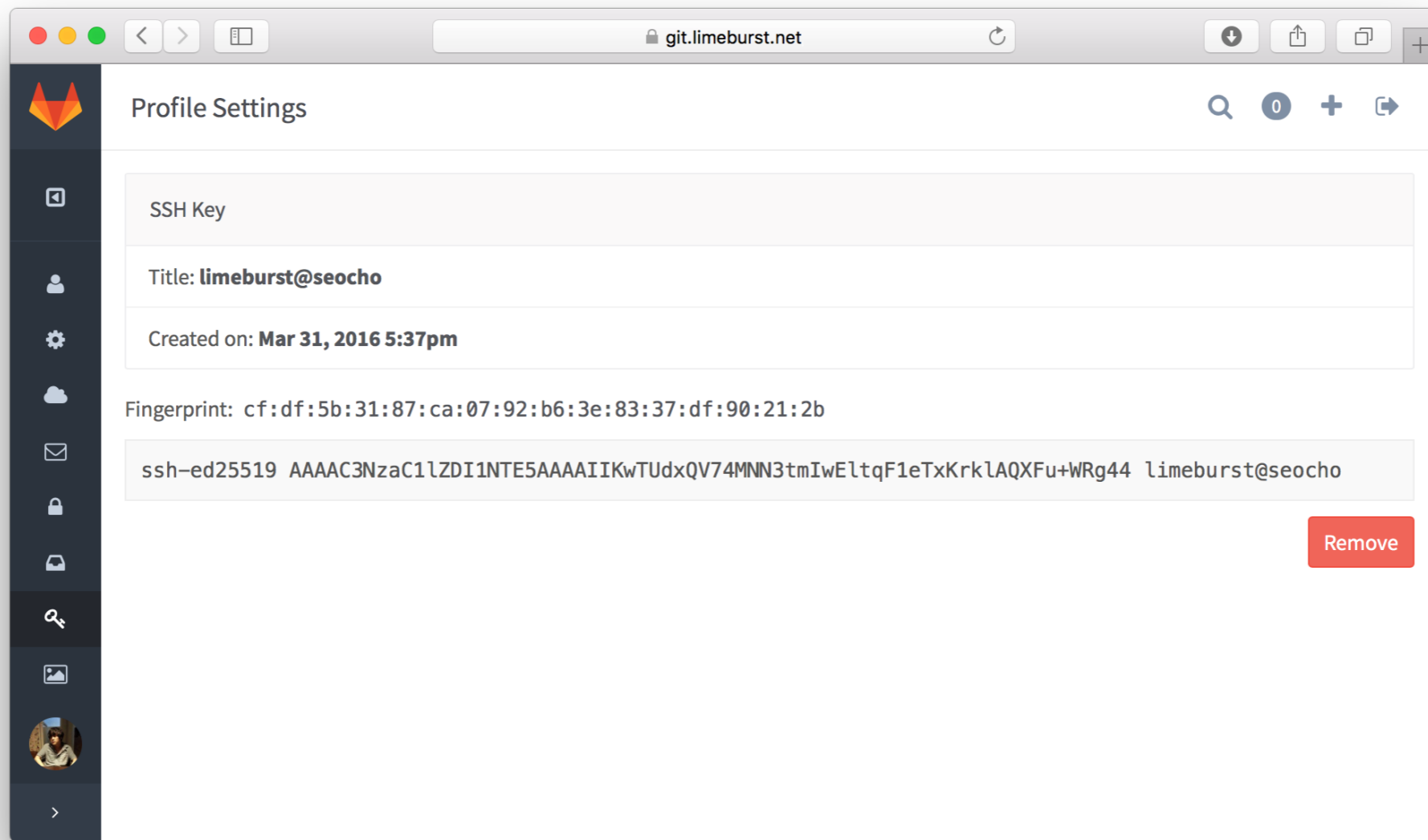
# GitLab 설치 완료 화면

<https://git.limeburst.net>

```
limeburst — -bash — 80x24
[Seocho:~ limeburst$ ssh-keygen -t ed25519 -C limeburst@seocho ]
Generating public/private ed25519 key pair.
Enter file in which to save the key (/Users/limeburst/.ssh/id_ed25519): ]
[Enter passphrase (empty for no passphrase): ]
[Enter same passphrase again: ]
Your identification has been saved in /Users/limeburst/.ssh/id_ed25519.
Your public key has been saved in /Users/limeburst/.ssh/id_ed25519.pub.
The key fingerprint is:
SHA256:DwfYnmFLAY7/turV2GNgg/0nC74tWXNhqyUfzf68CiQ limeburst@seocho
The key's randomart image is:
+--[ED25519 256]--+
|      .+.      |
|     = 0       |
|    0 *        |
|   + +   0     |
|  SE+. . =    |
|   *0*+ = 0   |
|    0+=B 0    |
|   *0=+.0..   |
|  .+. =++=. . =|
+-----[SHA256]-----+
Seocho:~ limeburst$
```

# GitLab 인증용 키 생성

```
$ ssh-keygen -t ed25519 -C limeburst@seocho
```



# GitLab 인증용 키 등록

```
$ pbcopy < ~/.ssh/id_ed25519.pub
```

# CI 도구

- 빌드, 테스트, 릴리즈 자동화와 그 외 CI 프로세스를 위한 도구
- 조직의 요구 사항과 밀접한 관련이 있다
  - 최대한 일반화된 도구가 필요하다: Buildbot
  - 빌드 스크립트의 일원화가 필요하다: Jenkins
  - 그냥 쓰고 싶다: Travis CI, GitLab CI
- VCS와 마찬가지로 필요에 맞는 것을 선택해서 사용

# GitLab CI

- GitLab 내장 CI 도구 (빌드 상태 저장 및 열람 인터페이스)
- 실제 빌드는 GitLab CI Runner 에서 진행
- 각종 Runner 타입 및 실행 엔진 지원
  - Shared Runner, Specific Runner
  - SSH, Shell, Docker, VirtualBox, Parallels, etc.

# Docker

- 애플리케이션을 소프트웨어 컨테이너로 배포하는 도구
- 선언적 애플리케이션 정의 (Dockerfile)
- 리눅스 운영체제 계층의 경량 가상화를 통한 격리
- 빌드 환경 간 간섭이 없어야 하는 CI Runner에 적합



# GitLab CI Runner 설치

```
ubuntu ~$ curl -sSL https://get.docker.com/ | sh
```

```
ubuntu ~$ curl -L https://packages.gitlab.com/  
install/repositories/runner/gitlab-ci-multi-runner/  
script.deb.sh | sudo bash
```

```
ubuntu ~$ sudo apt-get install gitlab-ci-multi-runner
```

```
limeburst — ubuntu@ip-172-31-23-204: ~ — ssh -i .ssh/gitlab.pem ubuntu@gi...
[ubuntu@ip-172-31-23-204:~$ sudo gitlab-ci-multi-runner register
Running in system-mode.
WARNING: The user-mode requires you to manually start builds processing:
WARNING: $ gitlab-runner run

Please enter the gitlab-ci coordinator URL (e.g. https://gitlab.com/ci):
[https://git.limeburst.net/ci
Please enter the gitlab-ci token for this runner:
[RXD_RviBPX5QDaM_GoSN
Please enter the gitlab-ci description for this runner:
[[ip-172-31-23-204]:
Please enter the gitlab-ci tags for this runner (comma separated):

Registering runner... succeeded runner=RXD_RviB
Please enter the executor: docker, docker-ssh, parallels, shell, ssh, virtualbox
, docker+machine, docker-ssh+machine:
[docker
Please enter the default Docker image (eg. ruby:2.1):
[ruby:2.1
Runner registered successfully. Feel free to start it, but if it's running alrea
dy the config should be automatically reloaded!
ubuntu@ip-172-31-23-204:~$
```

# GitLab CI Runner 등록

\$ sudo gitlab-co-multi-runner register

The screenshot shows the 'Admin Area' of a Git LIMEBURST instance. The page provides instructions on how to register a new runner using a registration token. The token is displayed as `RYD_RviBPX5QDaM_GoSN`. Below this, there is a button to 'Reset runners registration token'. The page also explains that a 'runner' is a process that runs a build and can be in one of three states: `shared` (run builds from all unassigned projects), `specific` (run builds from assigned projects), or `paused` (runner will not receive any new builds). At the bottom, there is a table of runners with the following data:

Type	Runner token	Description	Projects	Builds	Tags	Last contact	
shared	8ced6e4a	ip-172-31-23-204 (edit)	-	0		less than a minute ago	Edit Pause Remove

# Runner가 등록된 모습

<https://git.limeburst.net/admin/runners>

```
test — ubuntu@ip-172-31-23-204: ~ — vi .gitlab-ci.yml — 80x24
before_script:
  - apt-get update -qq && apt-get install -y -qq sqlite3 libsqlite3-dev nodejs
  - ruby -v
  - which ruby
  - gem install bundler --no-ri --no-rdoc
  - bundle install --jobs $(nproc) "${FLAGS[@]}"

rspec:
  script:
    - bundle exec rspec

rubocop:
  script:
    - bundle exec rubocop
~
~
~
~
~
~
~
~
~
```

# CI 스크립트 파일 추가

```
$ git add .gitlab-ci.yml
```

The screenshot shows a GitLab CI build page for a project named 'test' by user 'Jihyeok Seo'. The build is identified as 'Build #5 for commit 99aee159 from master'. The build status is 'failed' with a duration of 24 seconds, completed 'about a minute ago'. The terminal output shows the runner environment (gitlab-ci-multi-runner 1.1.0) using a Docker executor with image ruby:2.1. The runner is on ip-172-31-23-204. The build process includes cloning the repository, checking out the commit, and running a script. The script fails at the apt-get update command due to a dependency issue with libc-ares2:amd64. The build summary on the right shows the commit hash 99aee159, branch master, author Jihyeok Seo, and message 'added ci script'. There is one other build for this commit.

매 커밋/푸쉬마다 CI가 트리거되는 모습

\$ git commit && git push

# Tips & Tricks

- APT 저장소에서 응답이 없을 땐 패키지를 직접 받고 dpkg
- AWS로의 Let's Encrypt 인증서 배포는 letsencrypt-aws
- 첫 CI Runner는 Shared Runner로 등록하면 편하다

# References

- <https://aws.amazon.com>
- <https://letsencrypt.org>
- <http://yaml.org>
- <https://github.com/alex/letsencrypt-aws/>
- <https://about.gitlab.com/downloads/#ubuntu1404>
- <https://gitlab.com/gitlab-org/omnibus-gitlab/blob/master/doc/settings/nginx.md>
- <https://gitlab.com/gitlab-org/gitlab-ci-multi-runner/blob/master/docs/install/linux-repository.md>
- [https://en.wikipedia.org/wiki/Continuous\\_integration](https://en.wikipedia.org/wiki/Continuous_integration)
- [https://en.wikipedia.org/wiki/Version\\_control](https://en.wikipedia.org/wiki/Version_control)
- [https://en.wikipedia.org/wiki/Git\\_\(software\)](https://en.wikipedia.org/wiki/Git_(software))
- <https://en.wikipedia.org/wiki/GitLab>
- [https://en.wikipedia.org/wiki/Docker\\_\(software\)](https://en.wikipedia.org/wiki/Docker_(software))

감사합니다