



# Git/GitHub





Git / GitHub Training

Instructor



# 송태웅 (Taeung Song, <https://github.com/taeung>)

- NIPA KOSS(Korean Open Source Software) Lab. Software Engineer
- Linux Kernel 프로젝트 Contributor



Git / GitHub Training

Training Mode



우리가 스마트폰을 ( 메뉴얼 없이 ) 사용하면서 이해하듯

Git 이란 프로그램도 일단 써보면서 이해 해보자 ( 실습위주 교육 )



Git / GitHub Training

Training Mode



우리가 스마트폰을 ( 메뉴얼 없이 ) 사용하면서 이해하듯

Git 이란 프로그램도 일단 써보면서 이해 해보자 ( 실습위주 교육 )

백문이불여일행 ( 百聞不如一行 )



## Git / GitHub Training



### Training Mode

“Deduction”

Git 학습

add commit push pull  
show status log shortlog diff  
clone checkout merge fetch rebase  
reset stash blame cherry-pick submodule

Git 의 이해

“Induction”

상황별 적용

commit( 역사 한단위 ) 만들기      commit 에 서명 포함하기  
Github 에 밀어넣기      변화분 확인하기      브랜치 만들기  
commit 수정하기      commit 을 메인테이너에게 전송하기

야전경험  
( 시나리오 따라하며 )



Git / GitHub Training



## Table of Contents



Git 간단한 정의 / 기능



Git 기본 실습  
(How)



Git 이해하기  
(Why, What)



Git 고급  
(Advanced)



Opensource  
(with Git)



Git / GitHub Training

Contents



# Chapter 1. Git 소개

# Git 이란 ?

Git == History 관리 도구

# Git 이란 ?

Git 개발과정, 소스파일 등을 관리하는 도구

History 관리가 되어 개발되어온 과정, 역사를 볼 수 있고,  
특정시점으로 복구 가능

# Git 이란 ?

Ctrl+c, v 나 Alzip 압축파일 관리법

Source code management **tool**

그 사이에 뭐가 바뀌었는지  
차이 (Diff)를 알 수 없다.



과제 1\_최종\_2016\_02\_28.zip

▶ 과제 1\_진짜최종\_2016\_02\_29.zip

과제 1\_진짜진짜최종\_2016\_03\_01.zip

# Git 이란 ?

Ctrl+c, v 나 Alzip 압축파일 관리법

Source code management **tool**

그 사이에 뭐가 바뀌었는지  
차이 (Diff)를 알 수 없다.

Ctrl + c, v를 할수록  
차지하는 용량 X 2  
X 3 ... + diff



과제 1\_최종\_2016\_02\_28.zip

→ 과제 1\_진짜최종\_2016\_02\_29.zip

과제 1\_진짜진짜최종\_2016\_03\_01.zip

# Git 이란 ?

Ctrl+c, v 나 Alzip 압축파일 관리법

그 사이에 뭐가 바뀌었는지  
차이 (Diff)를 알 수 없다.

Ctrl + c, v를 할수록  
차지하는 용량 X 2  
X 3 ... + diff

Source code management **tool**

⟨ History 관리가능 ⟩  
**차이 (Diff)** 가 무엇이고  
수정 이유를 Log로 남길 수 있다.



과제 1\_최종\_2016\_02\_28.zip

→ 과제 1\_진짜최종\_2016\_02\_29.zip

과제 1\_진짜진짜최종\_2016\_03\_01.zip

# Git 이란 ?

Ctrl+c, v 나 Alzip 압축파일 관리법

그 사이에 뭐가 바뀌었는지  
차이 (Diff) 를 알 수 없다.

Ctrl + c, v 를 할수록  
차지하는 용량 X 2  
X 3 ... + diff

Source code management **tool**

〈 History 관리가능 〉  
**차이 (Diff)** 가 무엇이고  
**수정 이유**를 Log 로 남길 수 있다.



과제 1\_최종\_2016\_02\_28.zip

→ 과제 1\_진짜최종\_2016\_02\_29.zip

과제 1\_진짜진짜최종\_2016\_03\_01.zip

〈 타임머신 가능 〉  
**현재 파일들은 안전한 상태로**  
**과거상태** 그대로 복원가능 (반대도 가능)  
(각 버전별 차이만 저장해서 **size 감소**)

# Git 이란 ?

Ctrl+c, v 나 Alzip 압축파일 관리법

그 사이에 뭐가 바뀌었는지  
차이 (Diff) 를 알 수 없다.

Ctrl + c, v 를 할수록  
차지하는 용량 X 2  
X 3 ... + diff

Source code management **tool**

〈 History 관리가능 〉  
**차이 (Diff)** 가 무엇이고  
수정 이유를 Log 로 남길 수 있다.



과제 1\_최종\_2016\_02\_28.zip

과제 1\_진짜최종\_2016\_02\_29.zip

과제 1\_진짜진짜최종\_2016\_03\_01.zip

〈 특정 버전 관리 〉  
tag 나 release로  
관리 가능

〈 타임머신 가능 〉  
**현재 파일들은 안전한 상태로**  
**과거상태 그대로 복원 가능** (반대도 가능)  
(각 버전별 차이만 저장해서 **size 감소**)

# Git 이란 ?

Ctrl+c, v 나 Alzip 압축파일 관리법



Git 배우는데 시간 소비하느니  
Code 한줄이라도 더 개발 ..)

Source code management **tool**

좋은건 알겠는데 ..  
Git 을 쓸 이유가 부족 ..

Wants 는 맞지만 Needs 는 아니야



과제 1\_최종\_2016\_02\_28.zip  
과제 1\_진짜최종\_2016\_02\_29.zip  
과제 1\_진짜진짜최종\_2016\_03\_01.zip



Git / GitHub Training



How to use git

Git 간단한 정의 / 기능

Git 기본 실습  
(How)

Git 이해하기  
(Why, What)

Git 고급  
(Advanced)

Opensource  
(with Git)



Git / GitHub Training

Git 실습 (Basic)



Don't think about git,  
just do git

시나리오대로 단계 단계 똑같이 따라 해보자 Git

# Git 실습 준비단계

## ▶ 예제소스 다운받기

- git-training-ex-v2.zip 압축풀고 폴더 열어두기  
[https://www.dropbox.com/sh/9q2emkhxmyckoj6/AAA\\_H55BVhfRvGH0s9j7l9N2a?dl=1&pv=1](https://www.dropbox.com/sh/9q2emkhxmyckoj6/AAA_H55BVhfRvGH0s9j7l9N2a?dl=1&pv=1)

## ▶ Git 설치하기

- Window 사용자라면 Git-bash 설치 <https://git-scm.com/downloads>
- Mac 사용자는 git-osx-installer 를 통한 설치 <https://git-scm.com/downloads>
- Linux 사용자는 # sudo apt-get install git

## ▶ Editor 다운받기

- 이미 사용중인 Editor 가 있다면 그대로 이용
- 사용중인 Editor 가 없다면 Atom 써보기 <https://atom.io/>

## ▶ Github 회원가입

- 회원가입 <https://github.com/join>
- 메일 인증하기

# Git 실습 준비단계

가입정보 기입

Join GitHub  
The best way to design, build, and ship software.

Personal Open source Business Explore Pricing Blog Support Search GitHub Sign in Sign up

Step 1: Set up a personal account Step 2: Choose your plan Step 3: Go to your dashboard

Create your personal account

Username

This will be your username — you can enter your organization's username next.

Email Address

You will occasionally receive account related emails. We promise not to share your email with anyone.

Password

Use at least one lowercase letter, one numeral, and seven characters.

By clicking on "Create an account" below, you are agreeing to the [Terms of Service](#) and the [Privacy Policy](#).

Create an account

You'll love GitHub

- Unlimited collaborators
- Unlimited public repositories
- ✓ Great communication
- ✓ Friction-less development
- ✓ Open source community

# Git 실습 준비단계

본인 메일 확인해서 버튼 누르기



Hi @yukult400!

Help us secure your GitHub account by verifying your email address  
(yukult400@naver.com). This lets you access all of GitHub's features.

Verify email address

Button not working? Paste the following link into your browser:

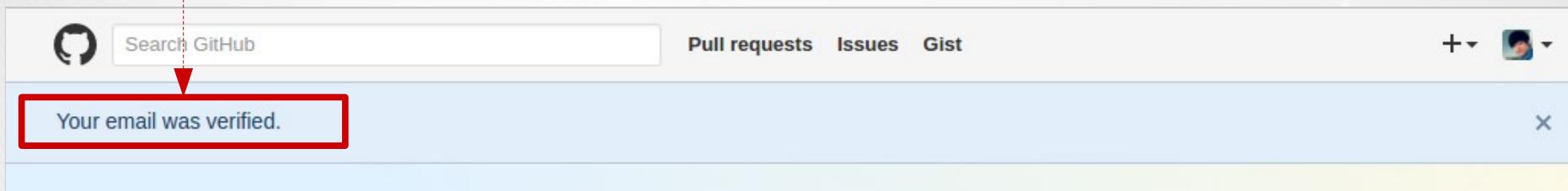
[https://github.com/users/yukult400/emails/21042026/confirm\\_verification/388b1d3024fc012d64848df72d2d8fc4d7dc3b5a](https://github.com/users/yukult400/emails/21042026/confirm_verification/388b1d3024fc012d64848df72d2d8fc4d7dc3b5a)

You're receiving this email because you recently created a new GitHub account or added a new email address. If this wasn't you, please ignore this email.

GitHub

# Git 실습 준비단계

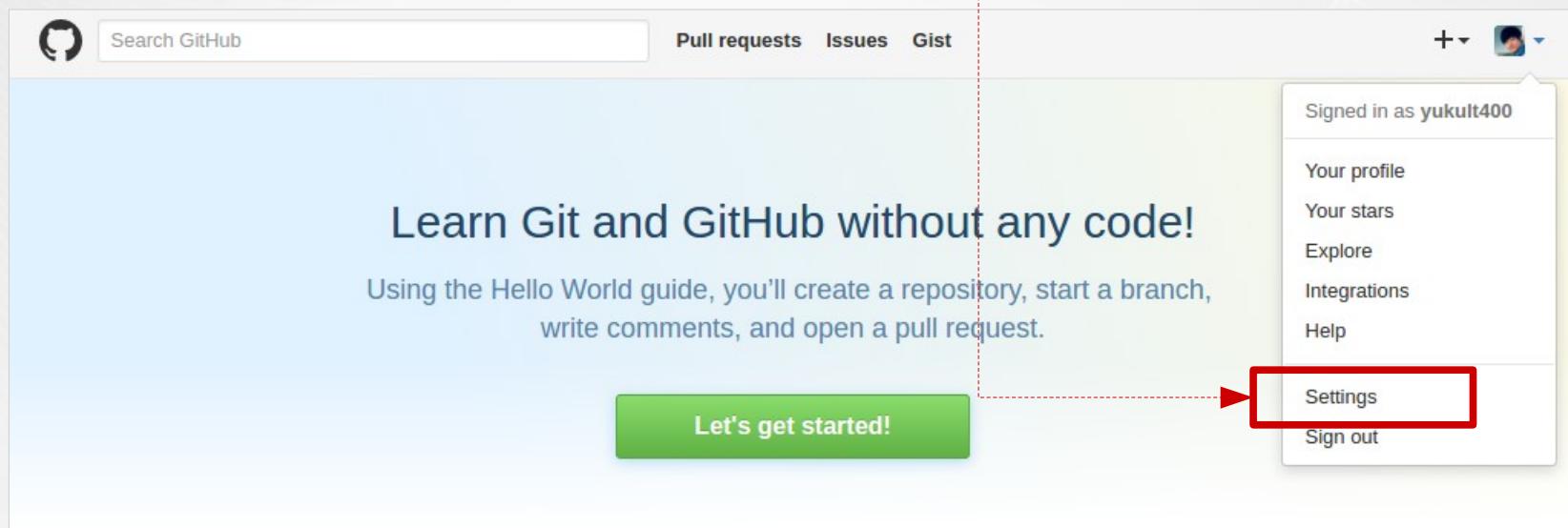
다시 Github 접속해서 메일인증 확인하기



<http://github.com>

# Git 실습 준비단계

기다려도 메일이 혹시 안왔으면 설정버튼



# Git 실습 준비단계

메일이 혹시 안왔으면 인증메일 다시 보내기

The screenshot shows the GitHub Personal Settings page under the 'Email' tab. On the left sidebar, 'Emails' is selected and highlighted with a red box and an exclamation mark icon. The main 'Email' section displays the primary email address 'yukult400@naver.com' which is marked as 'Primary', 'Public', and 'Unverified'. A red box highlights the status 'Unverified' next to the email address. To the right, a message says 'Verification email sent. Resend' with a trash can icon. The top navigation bar includes 'Search GitHub', 'Pull requests', 'Issues', 'Gist', and a user profile icon.

# Git 실습 준비단계

Git 실습 방법은 ?

직접 C 프로그래밍을 짜면서 Git 을 사용한다는 시나리오

report card ( 성적 출력하기 문제 )

- 1) 코딩은 Ctrl-c, v로 하고
- 2) Git 은 직접 명령어 쳐서 (git-bash)

# Git 실습 준비단계

Stage 해결하기 위해서 수단과 방법을 가리지 말자!

본인의 Github 계정에 본인의 프로젝트에 commit 기록이 다음과 똑같아 지면 된다.

<https://github.com/yukult400/report-card/commits/master>

The screenshot shows a GitHub repository page for 'yukult400 / report-card'. The page has a navigation bar with 'This repository', 'Search', 'Pull requests', 'Issues', and 'Gist'. On the right, there are buttons for 'Unwatch', 'Star', 'Fork', and user profile. Below the navigation, the repository name 'yukult400 / report-card' is displayed, along with its status: '1 commit · 1 pull request · 0 issues · 0 forks · 0 stars'. A dropdown menu shows the current branch is 'master'. The main content area shows a list of commits made on April 11, 2016, by 'Taeung Song'. Each commit includes a small profile icon, the commit message, the date it was committed, and a copy icon. The commits are:

- report card: Get a average of grades ... (32096bd)
- report card: Show the sum of each grade ... (e414372)
- report card: Print grades of each subject (696826a)
- report card: Print a message of introduction (d8e802e)
- report card: Add base code (b3dea79)
- report card: Add question PDF (ebce51d)

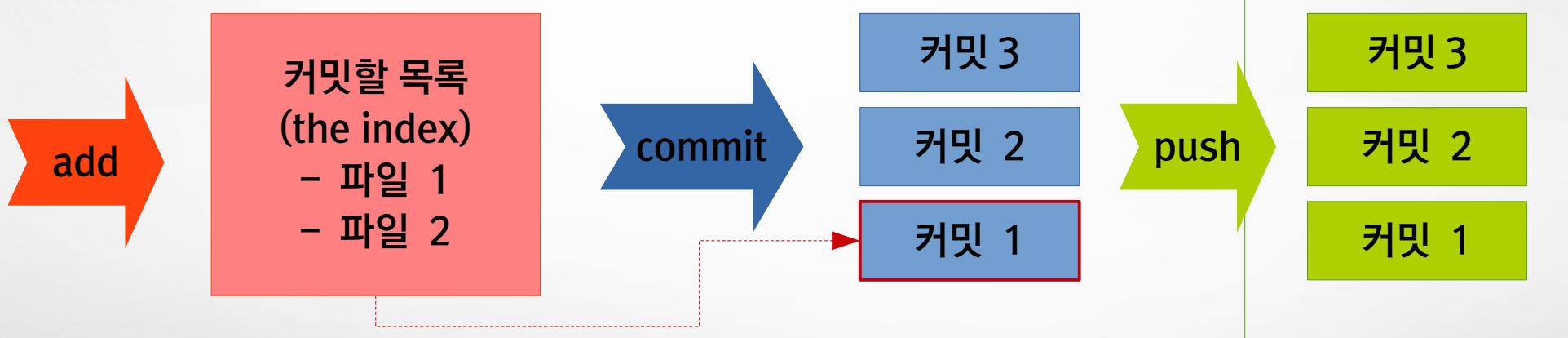
# Git 실습에 필요한 기초개념

## 〈 Git 필수 명령 〉

**add** : 커밋할 목록에 추가

**commit** : 커밋 ( 히스토리의 한단위 ) 만들기

**push**: 현재까지 역사 (commits) Github 에 밀어넣기



# Git 실습 준비

Prompt, 프롬프트

Arguments, 인자(1 개)

```
# git add report_card.pdf
```

Git 의 Sub-command 중 하나

Arguments, 인자 (2 개)

```
# git config --global user.name "Taeung Song"
```

--로 시작하면 보통은 Long Name 옵션

```
# git commit -s
```

-로 시작하면 보통은 Short Name 옵션

# Git 실습 목표

- 본 강의는 **실습 90%** 이론 10% (스파르타)
- 나중에 하는건 없다 무조건 오늘 목표는 이루고 가자
- **질문은 모두를 이롭게 한다** (taeung@kosslab.kr 메일로 대체 )
- 내 손으로 명령어를 직접 입력해보자
- 터미널 환경 경험하자
- GUI(Graphic User Interface) 대신 CLI(Command Line Interface) 경험하자
- vi 에디터를 사용해보자



Git / GitHub Training

Contents



# Chapter 2. Git 초기화 및 첫 commit

## Stage 1. 초기화 및 첫 commit

1) Git bash를 실행(명령어 칠 준비), 폴더 생성하기

```
# mkdir report-card
```

2) 경로 이동

```
# cd report-card
```

3) 해당 폴더를 git 초기화

```
# git init
```

4) 프로그램 문제 PDF 파일 추가 (**커밋할 목록에 추가 add**) (commit1 폴더내 파일 활용 )

```
# git add report_card.pdf
```

5) 첫 commit 하기 (역사 한단위 만들기 )

```
# git commit -m "report card: Add question PDF"
```

Git 상태확인 명령어  
(중간중간 치면서 수시로 확인하자)

```
# git show  
# git log  
# git shortlog  
# git diff  
# git status
```

## Stage 1. 초기화 및 첫 commit

몰라도 좋으니, 일단 따라해보자.

- 6) 소스코드 추가하기 (커밋할 목록에 추가 add) (commit2 폴더내 파일 활용)

```
# git add report_card.c
```

- 7) commit 하기 (역사 한단위 만들기)

```
# git commit -m "report card: Add base code"
```

Git 상태확인 명령어  
(중간중간 치면서 수시로 확인하자)

```
# git show  
# git log  
# git shortlog  
# git diff  
# git status
```

## Stage 2. diff 사용과 추가 commit

몰라도 좋으니, 일단 따라해보자.

1) 상태를 확인한다

```
# git status
```

2) **commit3** 폴더에 있는 report\_card.c 소스 파일로 수정, 덮어쓰기 후 확인

```
# git diff
```

3) diff 를 통해서 변화분을 확인했다면 add 진행

```
# git add report_card.c
```

4) 준비된 소스파일을 commit 한다

```
# git commit -m "report card: Print a message of introduction"
```

5) 지금까지한 3 개의 commit 들을 확인하자

```
# git log
```

## Stage 2. diff 사용과 추가 commit

몰라도 좋으니, 일단 따라해보자.

6) **commit4 폴더**에 있는 report\_card.c 소스 파일로 수정, 덮어쓰기 후 확인

```
# git diff
```

7) diff를 통해서 변화분을 확인했다면 add 진행

```
# git add report_card.c
```

8) 준비된 소스파일을 commit 한다

```
# git commit -m "report card: Print grades of each subject"
```

9) 지금까지한 **4 개의 commit**들을 확인하자

```
# git log
```

## Stage 3. commit message 에 본인서명 포함하기

1) 나의 Github 계정 이메일과 이름을 적자

```
# git config user.email "본인메일적으세요@gmail.com"
```

```
# git config user.name "본인이름적으세요 Taeung Song"
```

2) **commit5** 폴더에 있는 report\_card.c 소스 파일로 수정, 덮어쓰기 후 확인

```
# git diff
```

3) diff를 통해서 변화분을 확인했다면 add 진행

```
# git add report_card.c
```

4) 서명과 함께 commit 한다 (-s 옵션으로 서명을 포함한다.)

```
# git commit -sm "report card: Show the sum of each grade"
```

## Stage 3. commit message 에 본인서명 포함하기

5) commit6 폴더에 있는 report\_card.c 소스 파일로 수정, 덮어쓰기 후 확인

```
# git diff
```

6) diff를 통해서 변화분을 확인했다면 add 진행

```
# git add report_card.c
```

7) 서명과 함께 commit 한다

```
# git commit -sm "report card: Get a average of grades"
```



Git / GitHub Training

Contents



# Chapter 3. Github 에 Push 하기 Commit 수정, 삭제 및 Add 취소

## Stage 4. Github 에 Push 하기

몰라도 좋으니, 일단 따라해보자.

- 1) 상태를 확인하고 현재 브랜치명 master 를 확인하자

```
# git status
```

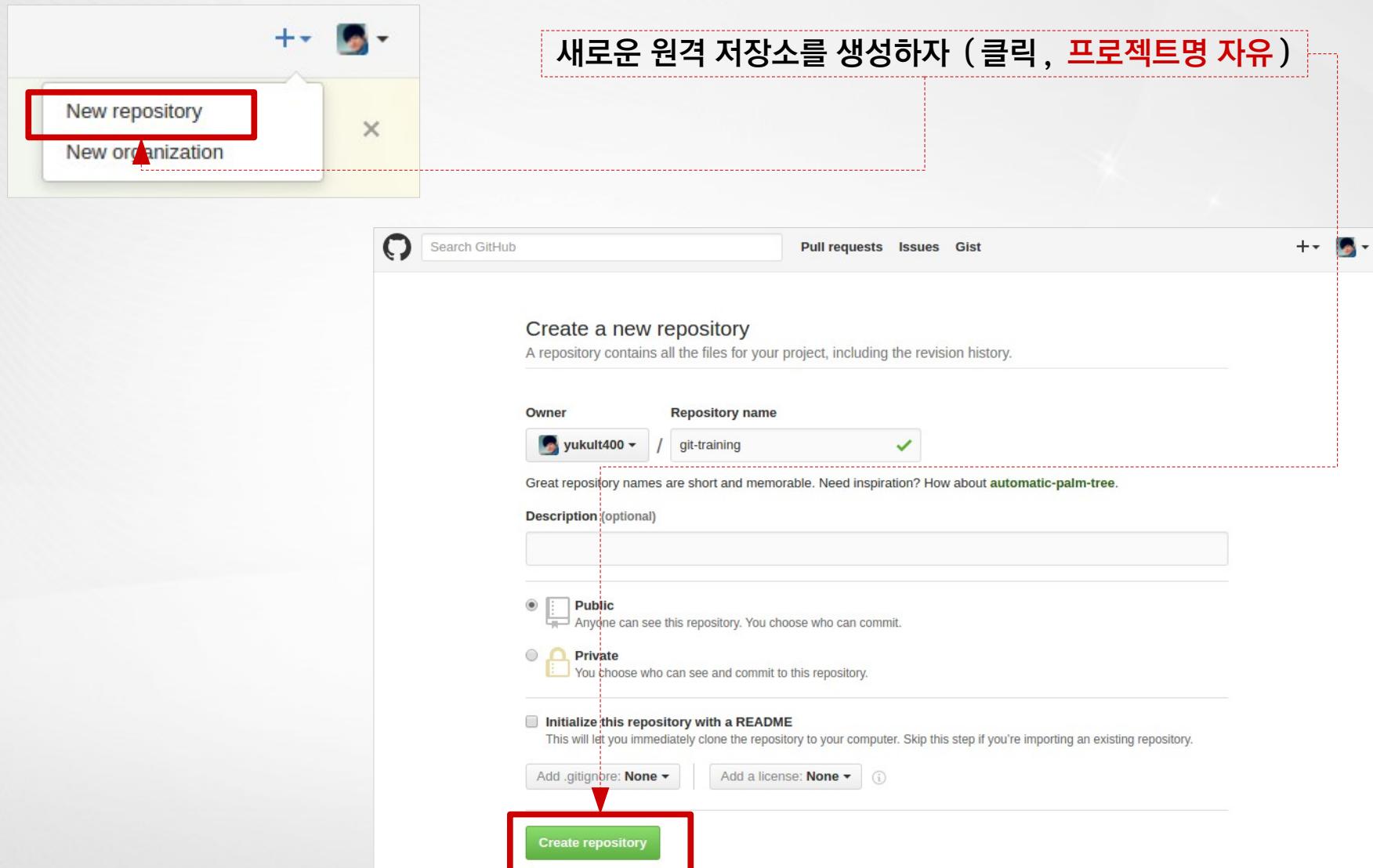
- 2) 지금까지한 commit 들을 확인하자 (6 개가 아니면 다시 확인하자)

```
# git shortlog
```

- 3) Github 원격저장소 URL 를 등록하자

(잠깐 멈추고 <http://github.com> 를 켜고 repository 새로 생성하자)

## Stage 4. Github 에 원격저장소 만들기



# Stage 4. Github 에 원격저장소 만들기

해당 URL 복사해서 Stage5 의 4) 으로 이어서 진행

This screenshot shows a GitHub repository page for 'yukult400 / git-training'. The page includes a navigation bar with links for 'Pull requests', 'Issues', 'Gist', and a user profile icon. Below the navigation bar, there are buttons for 'Unwatch', 'Star', and 'Fork'. The main content area features a 'Quick setup' section with instructions for cloning the repository. It provides options for 'HTTPS' and 'SSH', with the 'HTTPS' link highlighted by a red box. The URL 'https://github.com/yukult400/git-training.git' is also highlighted by a red box. Below this, a note says 'We recommend every repository include a README, LICENSE, and .gitignore.' Further down, there's a section for command-line repository creation with a code snippet:

```
echo "# git-training" >> README.md  
git init  
git add README.md
```

## Stage 4. Github 에 Push 하기

4) 아까복사한 URL로 Github 원격저장소 등록하자

```
# git remote add origin <아까복사한 URL>
```

5) Github 원격저장소 (origin)에 밀어 넣자

```
# git push origin master
```

6) Github 웹페이지 열고 확인하자

# Stage 4. Github 에 Push 결과확인

Github 들어가서 **본인** 프로젝트의 commit 기록 둘러보자

The screenshot shows a GitHub repository page for the user 'yukult400' with the repository name 'report-card'. A red box highlights the repository URL 'yukult400 / report-card'. The page displays basic repository statistics: 6 commits, 1 branch, 0 releases, and 0 contributors. A green button labeled 'New pull request' is visible. Below the stats, a commit list shows three files: 'report\_card.c', 'report\_card.pdf', and a commit message 'Taeung Song report card: Get a average of grades'. All items in the commit list were updated 2 hours ago. At the bottom, there's a note to 'Add a README'.

This repository Search Pull requests Issues Gist + Unwatch 1 Star 0 Fork 0

yukult400 / report-card

No description or website provided. — Edit

6 commits 1 branch 0 releases 0 contributors

Branch: master New pull request New file Upload files Find file HTTPS https://github.com/yuku Download ZIP

Taeung Song report card: Get a average of grades ... Latest commit 32096bd 2 hours ago

report\_card.c report card: Get a average of grades 2 hours ago

report\_card.pdf report card: Add question PDF 2 hours ago

Help people interested in this repository understand your project by adding a README. Add a README

# Stage 4. Github 에 Push 결과확인

본인이 추가한 commit 들이 나오는걸 확인하자 (본인 Github)

The screenshot shows a GitHub repository page for 'yukult400 / report-card'. The top navigation bar includes 'Pull requests', 'Issues', and 'Gist' tabs. Below the search bar, the repository name 'yukult400 / report-card' is displayed, along with statistics: 1 unwatched, 0 stars, and 0 forks. The main content area shows a list of commits on the 'master' branch from April 11, 2016. A red box highlights the first five commits, which all have the same author: 'Taeung Song committed an hour ago'. These commits are titled: 'report card: Get a average of grades', 'report card: Show the sum of each grade', 'report card: Print grades of each subject', 'report card: Print a message of introduction', and 'report card: Add base code'. To the right of each commit, there are icons for copy, view history, and fork.

Commit Message	Author	Date
report card: Get a average of grades	Taeung Song	committed an hour ago
report card: Show the sum of each grade	Taeung Song	committed an hour ago
report card: Print grades of each subject	Taeung Song	committed an hour ago
report card: Print a message of introduction	Taeung Song	committed 2 hours ago
report card: Add base code	Taeung Song	committed 2 hours ago
report card: Add question PDF	Taeung Song	committed 2 hours ago

## Stage 5. Commit 수정

- 1) report\_card.c 소스내의 'Mean' 변수명을 'Average'로 바꾼다고 가정하자  
(commit6-1 풀더내 소스 활용)

```
# git diff
```

- 2) diff를 통해서 변화분을 확인했다면 add 진행

```
# git add report_card.c
```

- 3) 가장 위에 있는 commit을 수정하자

```
# git commit --amend
```

## Stage 5. Remote 의 Commit 수정

4) 바로 push 해보자 (충돌 오류발생)

```
# git push origin master
```

5) 강제로 push 해서 수정하자

```
# git push origin master -f
```

6) 다시 Github 가서 제대로 변경되었는지 확인해보자

## Stage 6. Commit 삭제

1) 아까 test 파일 여전히 존재하는지 확인 ( 지웠으면 다시 만들기 )

```
# git status
```

2) 임의로 실수의 commit 을 만들어 낸다 (**:로 명령어들을 연속적 실행가능**)

```
# git add test; git commit -sm "test"
```

3) 그리고 push 까지 해서 Github 에 있는 tree 까지 실수 commit 을 넣어버린다

```
# git push origin master
```

4) 그리고 가장 최근 commit 을 지운다

```
# git reset HEAD~1
```

5) 강제로 Github 에 있는 tree 도 밀어넣어서 수정한다

```
# git push origin master -f
```

## Stage 7. Add 취소

1) touch로 빈파일 생성하고 add 하자 (;로 명령어들을 연속적 실행 가능)

```
# touch test; git add test
```

2) 현재상태 확인해보고

```
# git status
```

3) reset으로 add 한거 취소해보자

```
# git reset
```

4) 현재 상태 다시 한번 확인해본다

```
# git status
```



Git / GitHub Training

Contents



# Chapter 4. Pull-request 하기

# Github에서 Fork 하기

http://github.com/taeung/git-training 가서 fork 버튼을 누르자

This repository

Search

Pull requests Issues Gist

Taeung / git-training

Watch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Wiki Pulse Graphs

Don't think about git, just do git

9 commits 1 branch 0 releases 1 contributor

Branch: master New pull request

New file Upload files Find file HTTPS https://github.com/Taeu Download ZIP

Taeung packing knapsack: Implement code to solve this question ... Latest commit fd87207 14 hours ago

packing\_knapsack packing knapsack: Implement code to solve this question 2 hours ago

README.md Add README file 2 hours ago

# Github에서 Fork 하기

fork 가 되면 내 원격저장소가 추가 된다

The screenshot shows a GitHub repository page for 'yukult400 / git-training-1'. A red box highlights the repository name in the top navigation bar. Another red box highlights the 'Clone or download' button in the main content area. A red arrow points from the 'Clone or download' button to a callout box on the right containing the repository's URL and description. A red box also highlights the URL 'git-training-1' in the callout box. A red box highlights the 'forked from Taeung/git-training' note in the repository details.

This repository

Pull requests Issues Gist

yukult400 / **git-training-1**  
forked from Taeung/git-training

Watch 1 Star 0 Fork 66

Code Pull requests 0 Wiki Pulse Graphs

Don't think about git, just do git

9 commits 3 branches 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

This branch is 9 commits ahead, 12 commits behind Taeung:master.

Taeung packing knapsack: Implement code to solve this question ...

Latest commit fd87207 on Mar 25

packing\_knapsack README.md

packing knapsack: Implement code to solve this question  
Add README file

Popular repositories

git-training

**git-training-1**  
Don't think about git, just do git

fork 해서 만들어진 repo 의 URL 복사

나의 profile 페이지에서도 확인 가능 (Github)

## Stage 8. Pull-request 하기

1) clone 으로 fork 한 repo 받아오기 (주의 : .git이 생성된 폴더내에서 clone 하지말것)

```
# git clone <아까 fork 한 repo에서 복사한 URL>
```

2) pull-request 작업할 브랜치(develop) 따로 만들기

```
# git checkout -b develop
```

3) pull\_request\_test 폴더로 이동해서

```
# cd pull_request_test
```

4) 내 이름으로된 폴더 만들고

```
# mkdir taeung
```

5) 내 이름으로 된 폴더에 작업한 소스 및 임의의 파일을 넣고나서

## Stage 8. Pull-request 하기

6) 추가한 폴더(내가 작업한 소스 내용) 통째로 add

```
# git add <나의 소스작업폴더>
```

7) 준비된 파일들 commit

```
# git commit -sm "test pull request"
```

8) 내가 fork한 repo의 develop 브랜치로 push (주의 : master 아님)

```
# git push origin develop
```

# Stage 8. Pull-request 하기

The screenshot shows a GitHub repository page for 'yukult400 / git-training-1'. The top navigation bar includes 'Pull requests', 'Issues', and 'Gist'. Below the repository name, it shows 'forked from Taeung/git-training'. The main navigation tabs are 'Code', 'Pull requests 0', 'Wiki', 'Pulse', 'Graphs', and 'Settings'. A red box highlights the '4 branches' link in the top navigation bar. Below this, a summary bar shows '9 commits', '4 branches' (with a red arrow pointing up to the 'Branch' tab), '0 releases', and '1 contributor'. The 'Branch' tab is selected, showing a dropdown menu 'Branch: master' and a button 'New pull request'. To the right are buttons for 'Create new file', 'Upload files', 'Find file', and a green 'Clone or download' button. A message below the summary bar states 'This branch is 9 commits ahead, 12 commits behind Taeung:master.' A red dashed box highlights this message. Below this, a commit list shows 'Taeung packing knapsack: Implement code to solve this question' (latest commit on Mar 25) and two other commits for 'packing\_knapsack' and 'README.md'. The bottom right corner of the commit list has a red dashed box.

File	Commit Message	Date
packing_knapsack	packing knapsack: Implement code to solve this question	4 months ago
README.md	Add README file	4 months ago

방금 Push 했던 브랜치를 확인하기 위해서 Branch 탭 클릭

# Stage 8. Pull-request 하기

This screenshot shows a GitHub repository page for 'yukult400 / git-training-1'. The repository was forked from 'Taeung/git-training'. The 'Pull requests' tab is selected, showing 0 pull requests. The page includes navigation links for Code, Pull requests, Wiki, Pulse, Graphs, and Settings. Below these are tabs for Overview, Yours, Active, Stale, and All branches, with 'Overview' selected. A search bar allows searching for branches. The 'Default branch' section shows the 'master' branch, which was updated 4 months ago by Taeung. The 'Your branches' section shows the 'develop' branch, which was updated 3 months ago by Taeung. A red box highlights the 'New pull request' button in the top right corner of the 'Your branches' section. A dashed red arrow points from a callout box at the bottom left to this button.

Pull requests Issues Gist

yukult400 / git-training-1  
forked from Taeung/git-training

Unwatch 1 Star 0 Fork 66

Code Pull requests 0 Wiki Pulse Graphs Settings

Overview Yours Active Stale All branches Search branches...

Default branch

master Updated 4 months ago by Taeung Default Change default branch

Your branches

develop Updated 3 months ago by Taeung 9 | 13 New pull request

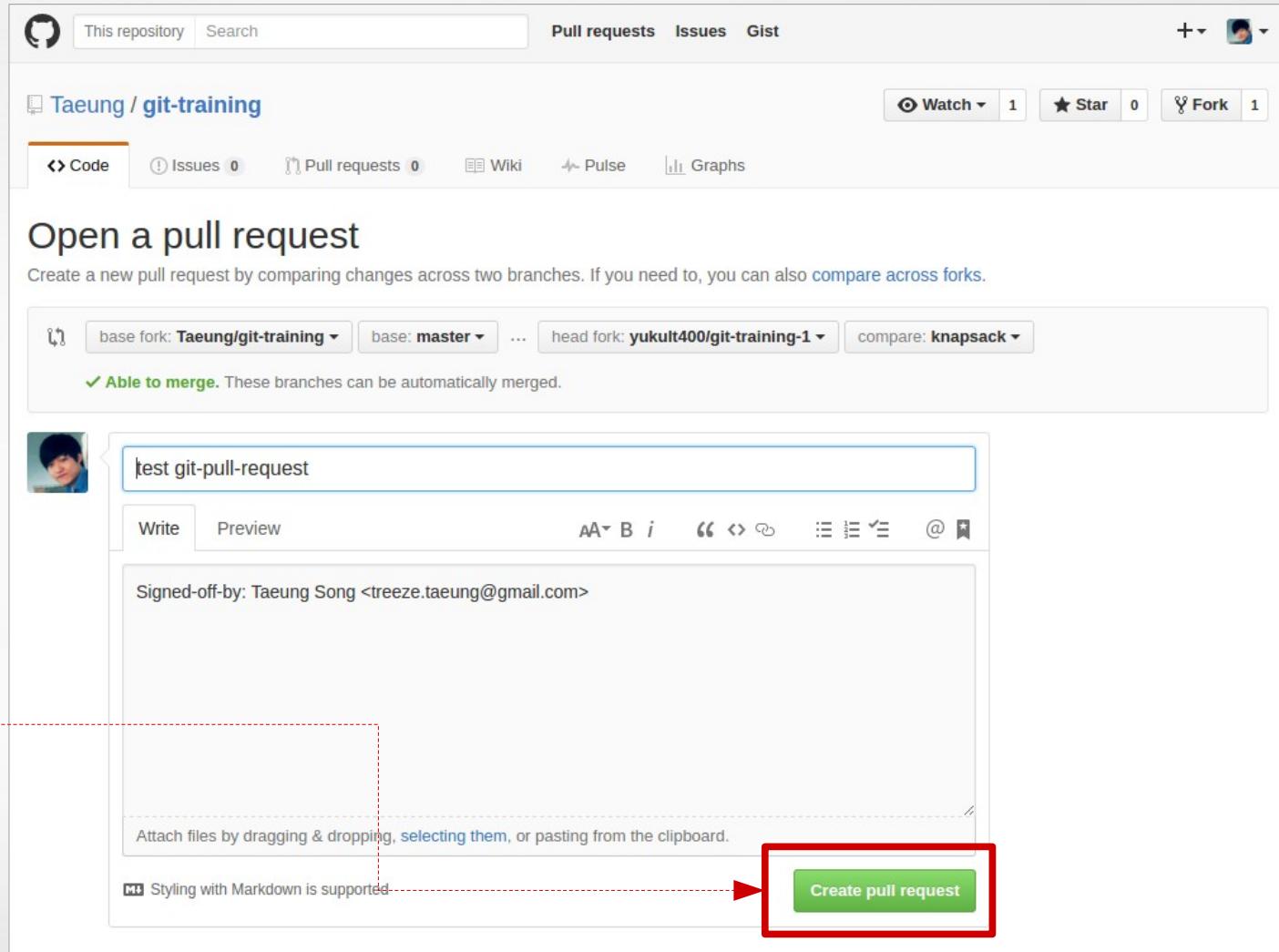
Pull-request 하려는 브랜치에서 New pull-request 버튼을 클릭

# Stage 8. Pull-request 하기

The screenshot shows a GitHub repository page for 'yukult400 / git-training-1'. The repository has 9 commits, 2 branches, 0 releases, and 1 contributor. A red box highlights the 'Compare & pull request' button in the top right corner of the main content area. Below the repository stats, there's a list of recently pushed branches, with 'knapsack-v2' being the most recent. The bottom section shows a list of files: 'packing\_knapsack' and 'README.md'. A red dashed box points from the bottom left towards the 'Compare & pull request' button.

다음과 같이 방금 Push 한 내용이 Pull-request 할 수 있도록  
버튼이 만들어지기도 한다.

# Stage 8. Pull-request 하기



The screenshot shows the GitHub interface for creating a pull request. At the top, there's a navigation bar with 'Pull requests', 'Issues', and 'Gist'. Below it, the repository 'Taeung / git-training' is selected. The main area is titled 'Open a pull request' with the sub-instruction 'Create a new pull request by comparing changes across two branches. If you need to, you can also compare across forks.' A dropdown menu shows 'base fork: Taeung/git-training', 'base: master', 'head fork: yukult400/git-training-1', and 'compare: knapsack'. A green checkmark indicates 'Able to merge'. The main text input field contains 'test git-pull-request'. Below it, a preview section shows 'Signed-off-by: Taeung Song <treeze.taeung@gmail.com>'. A note says 'Attach files by dragging & dropping, selecting them, or pasting from the clipboard.' A red arrow points to the 'Create pull request' button at the bottom right.

Pull-request 보내기

# Stage 8. Pull-request 확인

This repository Search Pull requests Issues Gist

Taeung / git-training Watch 1 Star 0 Fork 1

Pull requests 1

Code Issues 0 Pull requests 1 Wiki Pulse Graphs

## test git-pull-request #1

Open yukult400 wants to merge 1 commit into Taeung:master from yukult400:knapsack

Conversation 0 Commits 1 Files changed 4 +144 -0

yukult400 commented just now  
Signed-off-by: Taeung Song treeze.taeung@gmail.com

test git-pull-request ... ffc6d04

Add more commits by pushing to the knapsack branch on yukult400/git-training-1.

This branch has no conflicts with the base branch  
Only those with write access to this repository can merge pull requests.

Labels None yet

Milestone No milestone

Assignee No one assigned

Notifications Unsubscribe  
You're receiving notifications because you authored the thread.

2 participants

Write Preview

Leave a comment

만들어진 pull-request 확인하기



Git / GitHub Training

## How to use git



Git 간단한 정의 / 기능

Git 기본 실습  
(How)

Git 이해하기  
(Why, What)

Git 고급  
(Advanced)

Opensource  
(with Git)



Git / GitHub Training

Contents



# Chapter 5. Git 을 사용하는 진짜 이유

# Git 을 쓰는 이유

**협업** 때문에 Git 을 쓴다 (집단지성의 극대화)

현대적인, 선진화된 소스코드 개발 과정의 필수도구로 Git 을 쓴다 (**Needs**)



<https://geektimes.ru>

## Software 의 특징과 문제점



## Commit 단위개발 Idea

Commit 단위로 코딩하고 리뷰하고 토론하고 적용한다 . ( 집단지성의 극대화 )

소프트웨어의 취약점을 극복하는 전략



## Commit 단위개발의 전제

혼자가면 빨리가지만 (짧은다리로 빨리 달리기)

함께가면 멀리간다 (긴다리로 천천히 가기)



## Git 을 통한 협명한 협업

### Software 의 특징

비가시성 (Invisibility)

변경성 (Changeability)

공장에서의 제조 (Manufacture) 가 아니라 개발 (Development)



Git 과 Commit 단위 개발을 하면서 극복

Coding Style 또는 Coding Convention 은 기본

Commits( 변화분 ) 단위 관리를 통해 ( 코드를 수정한 이유가 명확 )

▶ 이 변화분을 통한 Review 와 Discussion 그리고 적용 ( 다수의 버그 / 오류 미리 차단 )

▶ 개발과정 가시화 + 협업용이 + 유지보수 + 추적 용이 + 소스코드 품질 상승

# Commit 단위 관리 예시

\* 참고 : Linux kernel 은 Github에서 mirror 까지는 되지만 다음 Repository 가 공식  
<http://git.kernel.org/cgit/linux/kernel/git/torvalds/linux.git/>

Commits(변화분) 단위 관리를 통해 (코드를 수정한 이유가 명확)

torvalds / linux

This repository Search

Pull requests Issues Gist

Watch 4,576 Unstar 30,768 Fork 12,172

Code Pull requests 96 Pulse Graphs

Linux kernel source tree

588,144 commits 1 branch 457 releases ∞ contributors

Branch: master New pull request New file Upload files Find file HTTPS https://github.com/torvalds/linux Download ZIP

torvalds Merge tag 'asm-generic-4.6' of git://git.kernel.org/pub/scm/linux/ker... ... Latest commit 11caf57 9 hours ago

⚠ Failed to load latest commit information.

Branch	Commit Message	Time Ago
Documentation	Merge tag 'pm+acpi-4.6-rc1-2' of git://git.kernel.org/pub/scm/linux/k...	9 hours ago
arch	Merge tag 'asm-generic-4.6' of git://git.kernel.org/pub/scm/linux/ker...	9 hours ago
block	Merge branch 'for-linus' of git://git.kernel.dk/linux-block	12 hours ago
certs	certs: Fix misaligned data in extra certificate list	25 days ago
crypto	Merge branch 'akpm' (patches from Andrew)	7 days ago
drivers	Merge tag 'asm-generic-4.6' of git://git.kernel.org/pub/scm/linux/ker...	9 hours ago
firmware	WHENCE: use https://linuxtv.org for LinuxTV URLs	4 months ago

<https://github.com/torvalds/linux>

# Commit 단위 관리 예시

Commits( 변화분 ) 단위 관리를 통해 ( 코드를 수정한 이유가 명확 )

The screenshot shows a list of four commits made by Taeung and Arnaldo Carvalho de Melo on Feb 25, 2016. Each commit includes the author's profile picture, the commit message, the date, the commit hash, and two icons for copy and share.

- Merge tag 'for-v4.5-rc' of git://git.kernel.org/pub/scm/linux/kernel/...  
torvalds committed 29 days ago  
1ebe383
- perf script: Remove duplicated code and needless script\_spec\_\_findnew()  
Taeung committed with Arnaldo Carvalho de Melo 29 days ago  
8560bae
- x86/mm: Fix slow\_virt\_to\_phys() for X86\_PAE again  
dcui committed with Thomas Gleixner 29 days ago  
bf70e55
- perf script: Exception handling when the print fmt is empty  
Taeung committed with Arnaldo Carvalho de Melo 29 days ago  
8579aca

Commits on Feb 25, 2016

<https://github.com/torvalds/linux>

# Commit 단위 관리 예시

Commits(변화분) 단위 관리를 통해 (코드를 수정한 이유가 명확)

Merge tag 'for-v4.5-rc' of git://git.kernel.org/pub/scm/linux/kernel/...  
torvalds committed 29 days ago

perf script: Remove duplicated code and needless script\_spec\_\_findnew()  
Taeung committed with Arnaldo Carvalho de Melo 29 days ago

script\_spec\_register() called two functions: script\_spec\_find() and script\_spec\_findnew(). But this way script\_spec\_find() gets called two times, directly and via script\_spec\_findnew().

So remove script\_spec\_findnew() and make script\_spec\_register() only call once script\_spec\_find().

Signed-off-by: Taeung Song <treeze.taeung@gmail.com>  
Acked-by: Jiri Olsa <jolsa@kernel.org>  
Cc: Namhyung Kim <namhyung@kernel.org>  
Link: <http://lkml.kernel.org/r/1456413190-12378-1-git-send-email-treeze.taeung@gmail.com>  
Signed-off-by: Arnaldo Carvalho de Melo <acme@redhat.com>

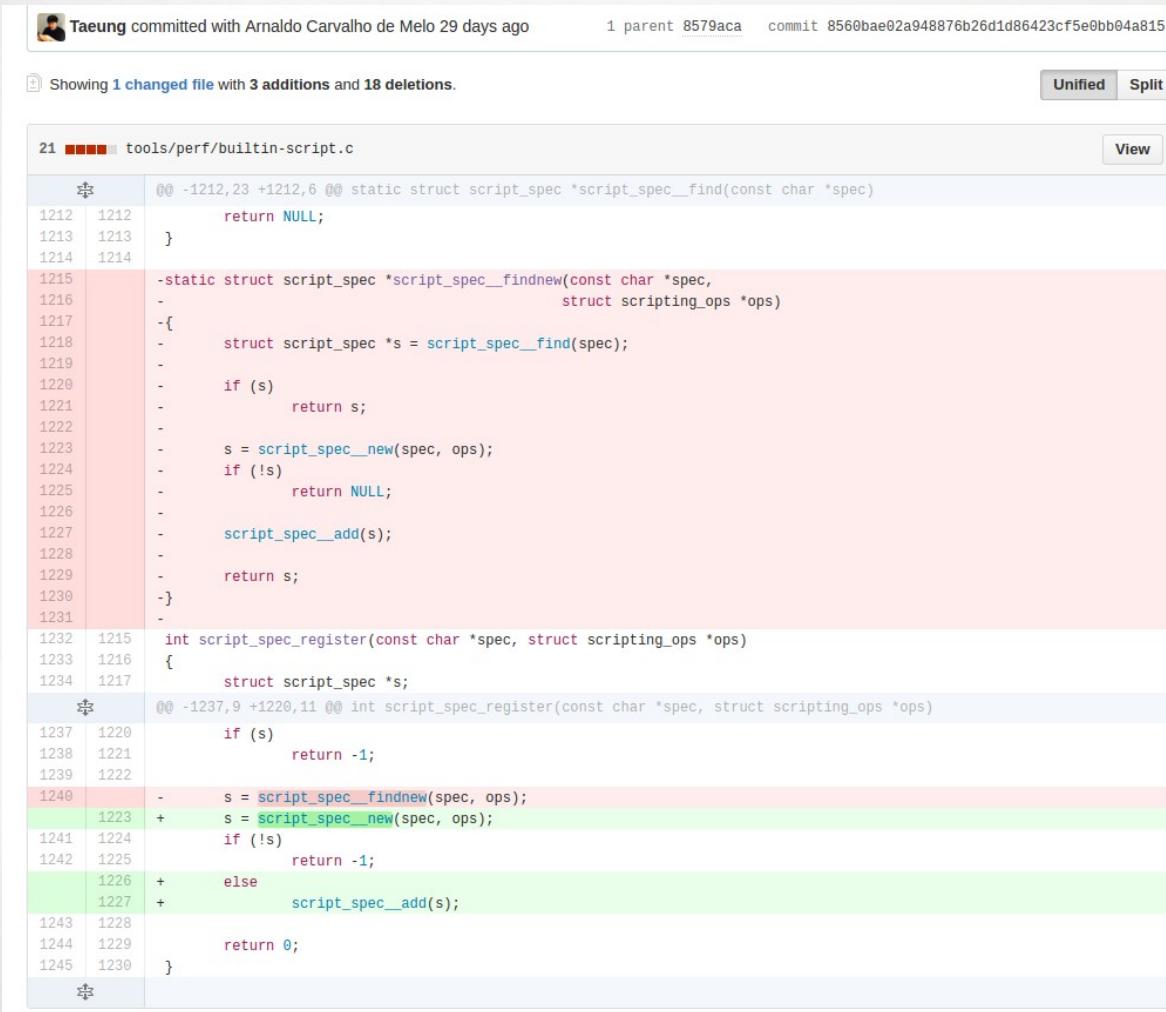
x86/mm: Fix slow\_virt\_to\_phys() for X86\_PAE again  
dcui committed with Thomas Gleixner 29 days ago

perf script: Exception handling when the print fmt is empty  
Taeung committed with Arnaldo Carvalho de Melo 29 days ago

<https://github.com/torvalds/linux>

# Commit 단위 관리 예시

Commits(변화분) 단위 관리를 통해 (코드를 수정한 이유가 명확)



Taeung committed with Arnaldo Carvalho de Melo 29 days ago

1 parent 8579aca commit 8560bae02a948876b26d1d86423cf5e0bb04a815

Showing 1 changed file with 3 additions and 18 deletions.

Unified Split

View

```
21 1212,23 +1212,6 @@ static struct script_spec *script_spec__find(const char *spec)
1212     return NULL;
1213 }
1214
1215 -static struct script_spec *script_spec__findnew(const char *spec,
1216 -                                               struct scripting_ops *ops)
1217 -{
1218 -    struct script_spec *s = script_spec__find(spec);
1219 -
1220 -    if (s)
1221 -        return s;
1222 -
1223 -    s = script_spec__new(spec, ops);
1224 -    if (!s)
1225 -        return NULL;
1226 -
1227 -    script_spec__add(s);
1228 -
1229 -    return s;
1230 -
1231
1232 1215 int script_spec_register(const char *spec, struct scripting_ops *ops)
1233 1216 {
1234 1217     struct script_spec *s;
1235
1236 1218     if (s)
1237 1219         return -1;
1238 1220
1239 1221
1240 1222 -    s = script_spec__findnew(spec, ops);
1241 1223 +    s = script_spec__new(spec, ops);
1242 1224     if (!s)
1243 1225         return -1;
1244 1226 +    else
1245 1227 +        script_spec__add(s);
1246 1228
1247 1229     return 0;
1248
1249 }
```

<https://github.com/torvalds/linux>

# Commit 단위 Review, Discussion 예시

이 변화분 (commits)을 통한 Review 와 Discussion 그리고 적용 (다수의 버그 / 오류 미리 차단)

The screenshot shows the GitHub repository page for 'atom / electron'. At the top, there are navigation links for 'Pull requests', 'Issues', and 'Gist'. Below the header, it displays the repository name 'atom / electron' and statistics: 1,261 watches, 25,814 stars, and 2,594 forks. A red arrow points to the 'Pull requests 13' button. The main content area shows a summary bar with 7,832 commits, 4 branches, 216 releases, and 331 contributors. Below this, a list of recent commits is shown:

Author	Commit Message	Time Ago
kevinsawicki	Merge pull request #4883 from atom/repl	12 hours ago
atom	Set _setDeprecatedOptionsCheck on exports	7 days ago
chromium_src	devtools: fix filesystem api usage and use prefs to track filesystem ...	11 days ago
default_app	Don't quit on window-all-closed when in repl mode	16 hours ago
docs-translations	fix link	6 days ago

# Commit 단위 Review, Discussion 예시

이 변화분 (commits) 을 통한 Review 와 Discussion 그리고 적용 (다수의 버그 / 오류 미리 차단)



#4892 opened 2 days ago by deepak1556	
[n] Improve error reporting when passing invalid argument types for dialog API methods ✓	0
#4887 opened 2 days ago by sergeybekrin	
[n] Added electron-release-server link to the docs ✓	2
#4885 opened 3 days ago by ArekSredzki	
[n] Run callback of setDestructor immediately when GC happens ✓	3
#4869 opened 4 days ago by zcbenz	
[n] Fix alert() ✓	5
#4861 opened 5 days ago by OctoHuman	
[n] Docs: Update Korean docs as upstream <a href="#">documentation</a>	0
#4854 opened 6 days ago by preco21	

# Commit 단위 Review, Discussion 예시

이 변화분 (commits) 을 통한 Review 와 Discussion 그리고 적용 (다수의 버그 / 오류 미리 차단)

The screenshot shows a GitHub pull request page for the 'atom/electron' repository. The pull request is titled 'Fix alert() #4861'. It has 13 pull requests and 322 issues. The 'Code' tab is selected. A red arrow points from the text above to the 'Conversation' section.

**OctoHuman commented 5 days ago**

Stop `alert()` from throwing an error when passing no parameters or when passing null or undefined as the message. For example if you run `alert()` in a browser it will display a blank dialog box, but in Electron it will throw a error because there is no `.toString()` method on `undefined`. Changing `alert()` like this will also make it more consistent with the major browsers implementations.

**OctoHuman added some commits 5 days ago**

- Fix alert() (9f65412)
- Refix alert() (116d611)

**DerNivel commented on the diff 3 days ago**

lib/renderer/override.js

```
... @@ -161,11 +161,14 @@ window.open = function(url, frameName, features) {  
161   161   // Use the dialog API to implement alert().  
162   162   window.alert = function(message, title) {
```

**DerNivel added a note 3 days ago**

Labels: None yet  
Milestone: No milestone  
Assignee: No one assigned  
Notifications: Subscribe (You're not receiving notifications from this thread)  
Participants: 3 participants (OctoHuman, DerNivel, SK)

# Commit 단위 Review, Discussion 예시

이 변화분 (commits) 을 통한 Review 와 Discussion 그리고 적용 (다수의 버그 / 오류 미리 차단)

Showing changes from 1 commit ▾ 1 changed file ▾

+1 -1 Diff options ▾

**Refix alert()**

OctoHuman committed 5 days ago commit 116d61185a887b50c2a214c0736e07dd6b806868

2 lib/renderer/override.js

OctoHuman committed 5 days ago commit 116d61185a887b50c2a214c0736e07dd6b806868

DerNivel added a note 3 days ago

Would it be possible to use default parameters. For example:

```
window.alert = function(message = '', title = '') {  
    // ...  
}
```

OctoHuman added a note 3 days ago

The problem with your code is if you passed `undefined` it still would be replaced by the default. So in your example passing `undefined` as the message would give me a blank `alert()` dialog. In a browser `alert()` will give me a blank dialog, but `alert(undefined)` will give me a dialog that says `undefined`

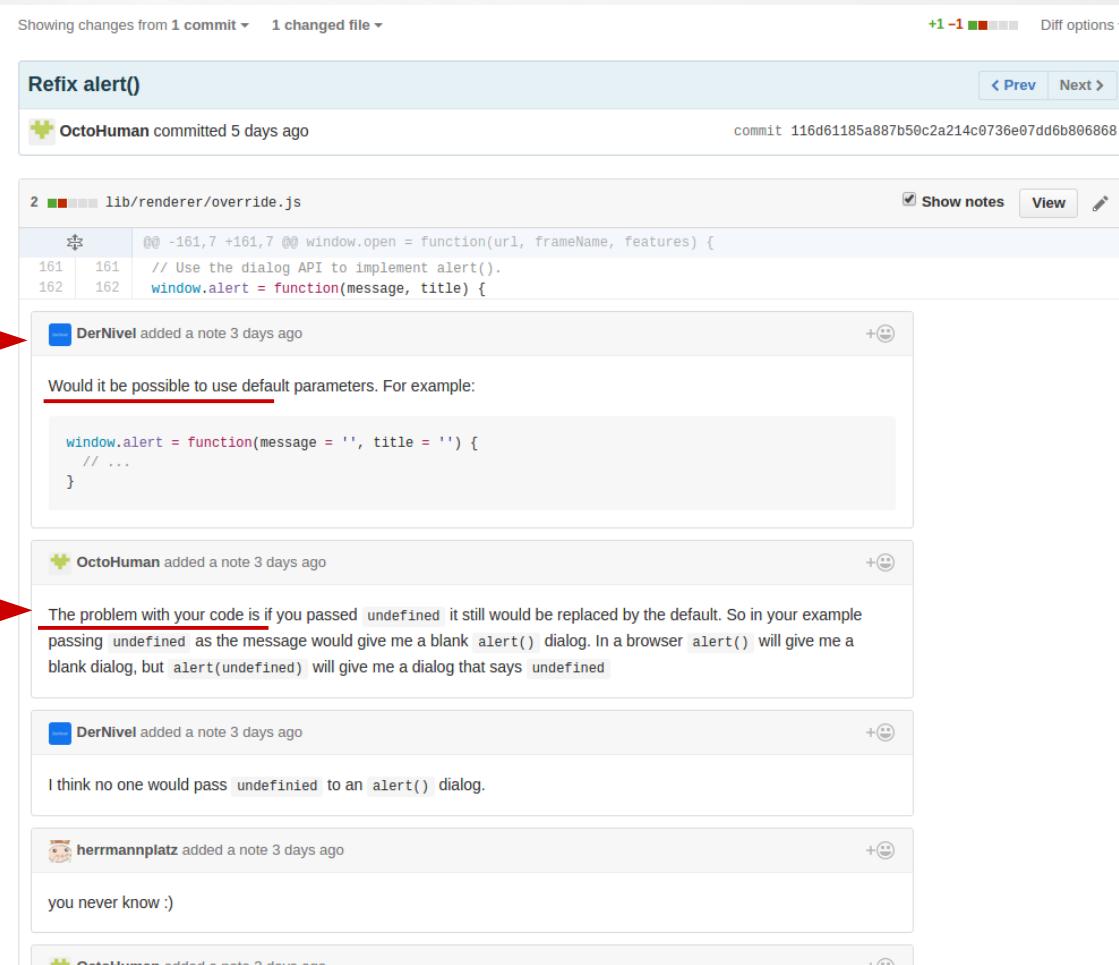
DerNivel added a note 3 days ago

I think no one would pass `undefined` to an `alert()` dialog.

herrmannplatz added a note 3 days ago

you never know :)

OctoHuman added a note 3 days ago



# Commit 단위 Review, Discussion 예시

이 변화분 (commits) 을 통한 Review 와 Discussion 그리고 적용 (다수의 버그 / 오류 미리 차단)

The screenshot shows a GitHub commit review interface. At the top, there are two notes:

- herrmannplatz** added a note 3 days ago:  
you never know :)
- OctoHuman** added a note 3 days ago:  
Maybe I'm weird, but I've used `alert(aBuggyVariable)` for debugging a decent amount, where knowing the exact value of `aBuggyVariable` is pretty important. Also it's only 3 lines of code, and then Electrons `alert()` will work the same as browsers `alert()`, which I imagine is what they are trying to emulate.

Below the notes is a "Add a line note" button. A red arrow points from the left margin to the "Add a line note" button. The main area shows a code diff with line numbers 163-167. Lines 164 and 165 have been modified:

```
163 163 var buttons;
164 - if (message === undefined) {
164 + if (arguments.length == 0) {
165 165   message = '';
166 166 +
167 167   if (title == null) {
```

## Git 을 쓰는 이유

Commits(변화분) 단위 관리를 통해 (코드를 수정한 이유가 명확)

▶ 개발과정 가시화 + 협업용이 + 유지보수 + 추적 용이 + 소스코드 품질 상승



Git 을 활용하는 오픈소스가 대표적인 증명 (IT 기업들의 높은 의존, 소프트웨어 역사를 이끄는)

▶ 카카오, NHN 엔터테이먼트 등 SW 기업들의 pull-request 방식도입

## Git 을 쓰는 이유

▶ Junior 개발자 교육 효과 (프로개발자의 코드를 볼수 있는 기회)

Commits( **변화분** ) 단위 관리를 통해 (코드를 수정한 이유가 명확)

▶ 개발과정 **가시화** + 협업용이 + 유지보수 + 추적 용이 + 소스코드 품질 상승

Git 을 활용하는 **오픈소스**가 대표적인 증명 (IT 기업들의 높은 의존 , 소프트웨어 역사를 이끄는)

▶ 카카오 , NHN 엔터테이먼트 등 SW 기업들의 pull-request 방식도입



Git / GitHub Training

Contents



# Chapter 6. Git 과 Github 통합구조

## Github 란 ?

# GitHub

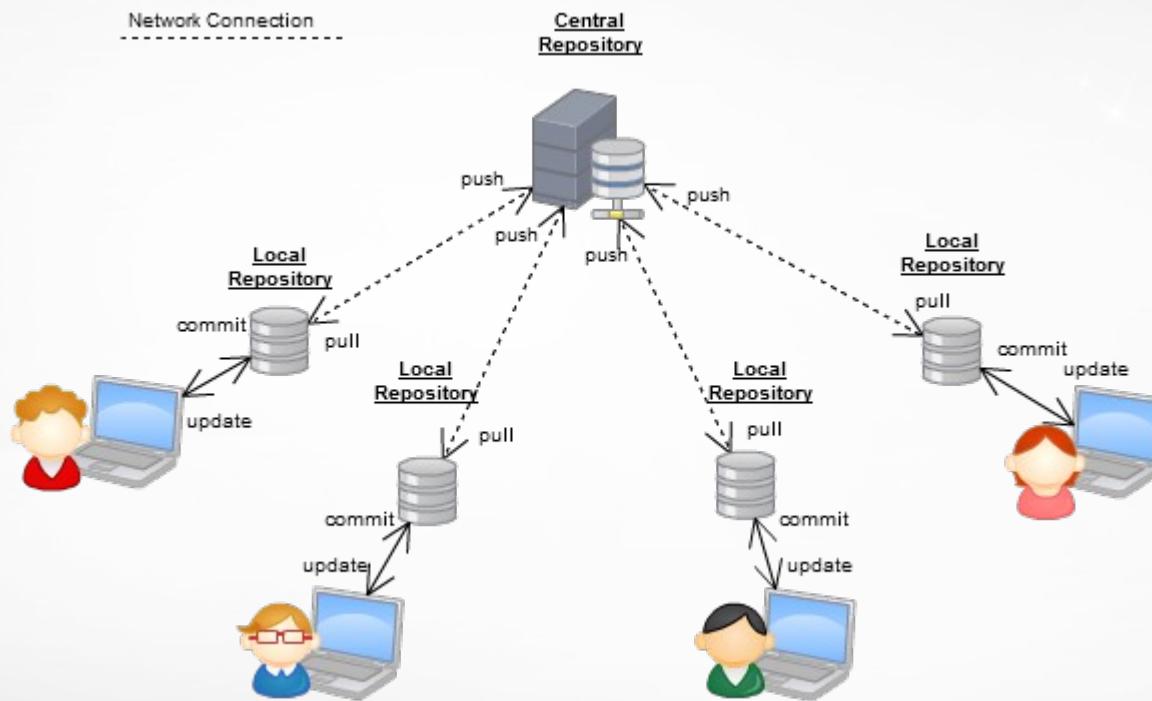


Git이라는 도구를 **응용**한 사이트

각종 **Remote repository** (원격저장소)들의 집합소

# Git 을 통한 작업의 흐름

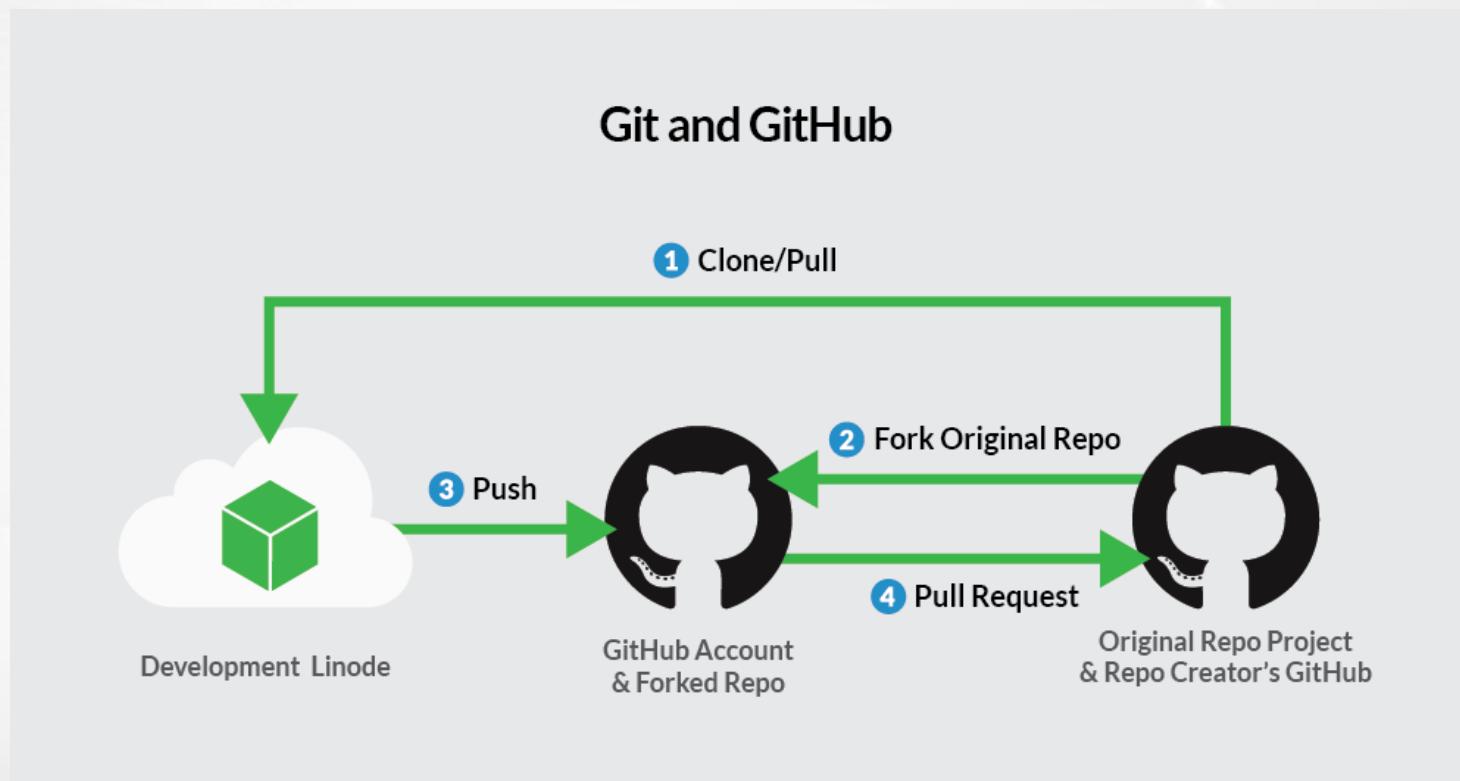
이제는 이 그림을 이해할 수 있다.



<http://jordankasper.com>

# Git pull-request 를 통한 작업의 흐름

이제는 이 그림도 이해할 수 있다.



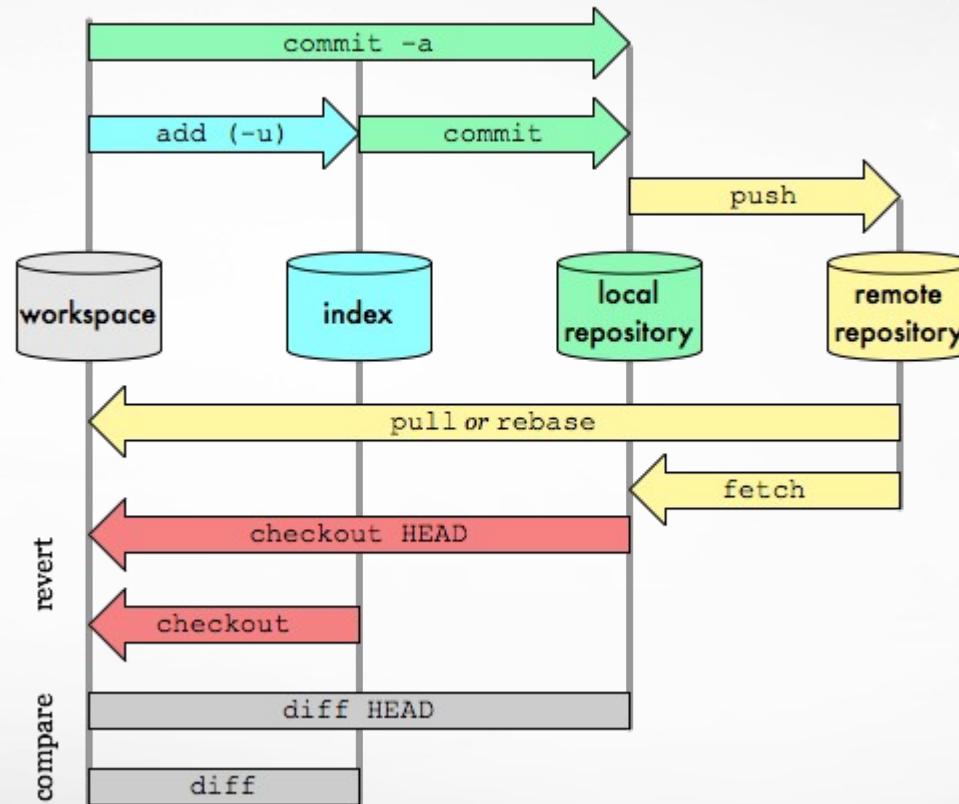
<https://www.linode.com/docs/applications/development/developing-git-github>

# Git 을 통한 작업의 흐름

이제는 이 그림도 이해할 수 있다.

Git Data Transport Commands

<http://cstealle.com>



# Opensource - Github / Not Github

## ➤ Not Github

- Apache (<http://git.apache.org>)
- Linux kernel (<http://git.kernel.org>)
- GNU (<http://git.savannah.gnu.org/cgit/>)
- Webkit (<git://git.webkit.org/WebKit.git>)
- ...

## ➤ Github

- Node.js (<https://github.com/nodejs/node>)
- Angular.js (<https://github.com/angular/angular.js>)
- Spring-boot (<https://github.com/spring-projects/spring-boot>)
- Rust (<https://github.com/rust-lang/rust>)
- Redis (<https://github.com/antirez/redis>)
- Flask (<https://github.com/mitsuhiko/flask>)

# Not Github - Linux Kernel git repo

 Kernel.org git repositories  
Git repositories hosted at kernel.org

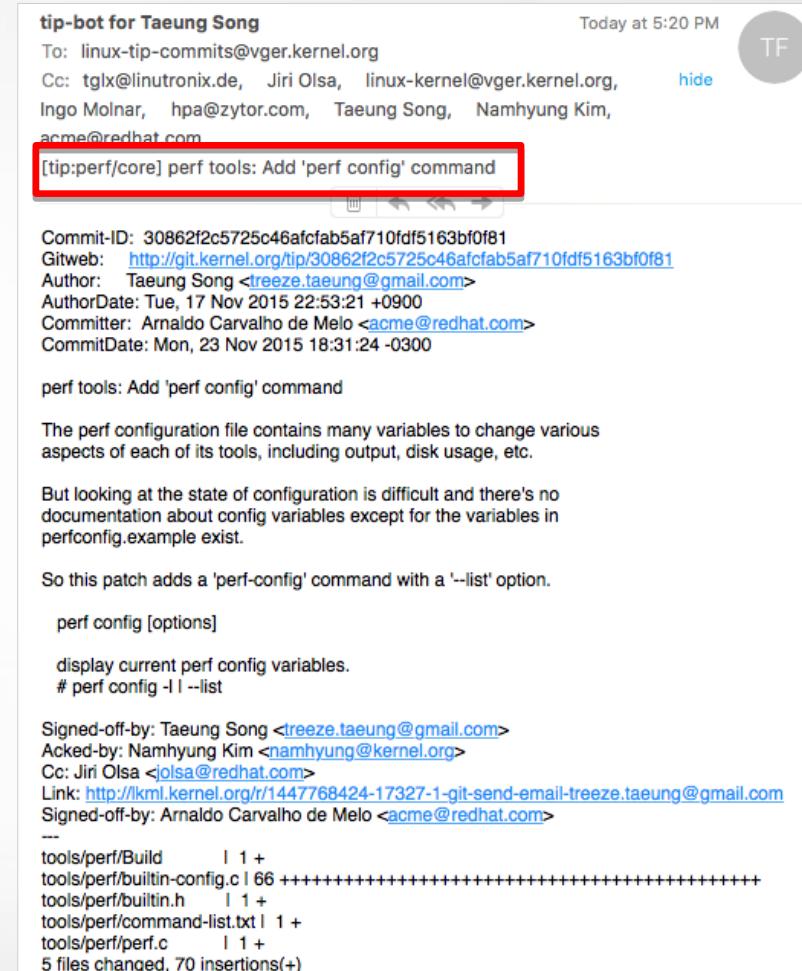
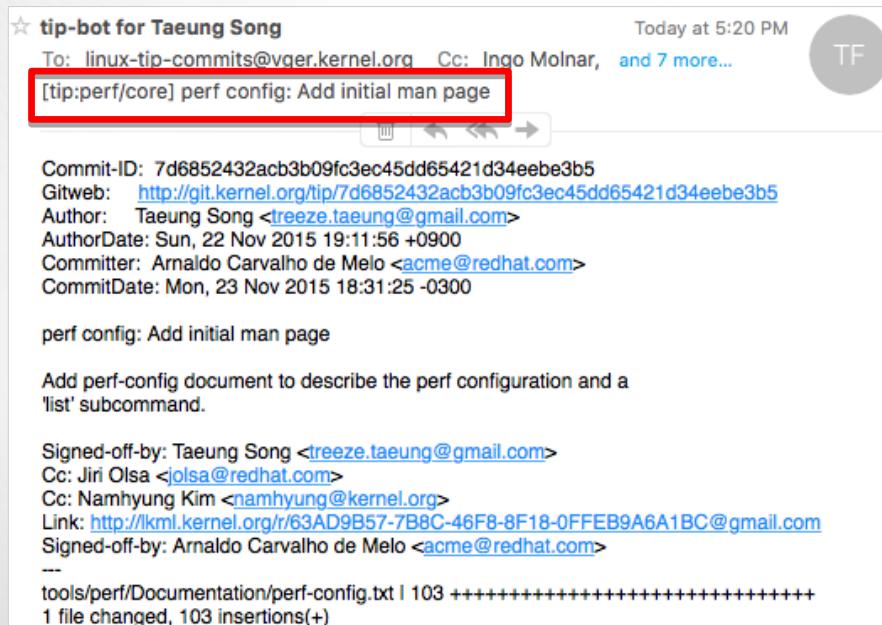
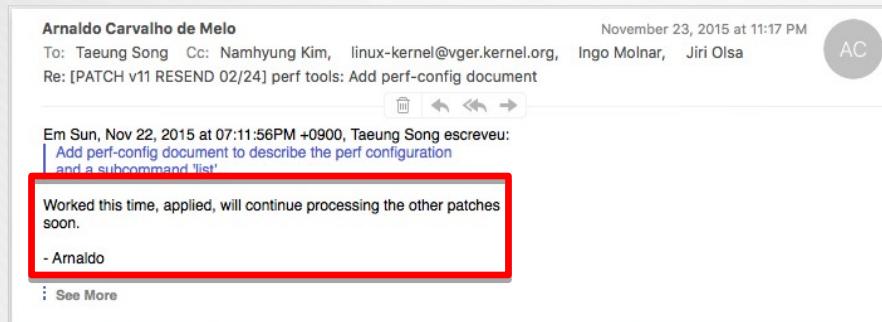
[index](#)  [search](#)

Name	Description	Owner	Idle	Links
<i>bluetooth</i>				
bluez-hcidump.git	Bluetooth packet analyzer	Marcel Holtmann	3 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
bluez.git	Bluetooth protocol stack for Linux	Marcel Holtmann	26 hours	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
obexd.git	OBEX Server	Marcel Holtmann	3 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
sbc.git	SBC library	holtmann	17 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
<i>boot</i>				
dracut/dracut.git	dracut - Initramfs generator using udev	Harald Hoyer	8 days	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
efilinux/efilinux.git	The efilinux UEFI boot loader	Matt Fleming	21 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
syslinux/syslinux.git	The Syslinux boot loader suite	Syslinux workgroup	18 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
<i>devel</i>				
pahole/pahole.git	Pahole and other DWARF utils	Arnaldo Carvalho de Melo	10 days	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
sparse/chris/sparse.git	Chris Li's sparse repository.	Christopher Li	5 weeks	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
sparse/sparse.git	C semantic parser	Christopher Li	13 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
<i>docs</i>				
kernel/kernel-docs.git	Kernel Documentation tree	Doc Group	2 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
kernel/ksmap.git	Kernel.org keysign map source	Kernel.org users	4 days	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
kernel/website.git	Kernel.org website source	Doc Group	13 days	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
man-pages/man-pages.git	Linux man pages Sections 2, 3, 4, 5, and 7	Michael Timothy Kerrisk	9 days	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
man-pages/website.git	Website files for /doc/man-pages	Michael Timothy Kerrisk	2 weeks	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
<i>editors</i>				
uemacs/uemacs.git	Micro-emacs	Linus Torvalds	16 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
<i>fs</i>				
ext2/e2fsprogs.git	Ext2/3/4 filesystem userspace utilities	Theodore T'so	3 days	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
ext2/xfstests-bld.git	Build framework and autorun scripts for xfstests	Theodore T'so	4 weeks	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
fat/fatattr/fatattr.git	FAT attribute set utility	H. Peter Anvin	7 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
fuse/dbfs.git	FUSE fs w/ Berkeley DB backend.	Jeff Garzik	7 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
fuse/fuse-ext2.git	FUSE ext2 filesystem driver.	Jeff Garzik	10 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
squashfs/squashfs-tools.git	squashfs tools development	Phillip Louher	4 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
xfs/dmapi-dev.git	Data Management API runtime environment	Christoph Hellwig	5 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
xfs/xfs-documentation.git	XFS AsciiDoc Documentation tree	XFS FS Group	3 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>

<https://git.kernel.org/cgit>

# PATCH mail 개발 방식 소개

## 이 변화분 (commits)을 통한 Review 와 Discussion 가능 (pull-request, PATCH mail)



# Opensource 각종 오픈소스

**BLACKDUCK | Open HUB**

PROJECTS PEOPLE ORGANIZATIONS TOOLS CODE BLOG

Follow @ OH SIGN IN JOIN NOW

Discover, Track and Compare Open Source

Search Projects... 

Join Now  Claim your contributions

Manage your project's data  TA/G/

Highlight your use of FOSS 

**Join Now**

What's New

**FREE BLACK DUCK VULNERABILITY PLUGIN** 

Identify open source software in your projects and discover known vulnerabilities.

**Most Popular Projects**

 Mozilla Firefox	13261 users
 Apache HTTP Server	9466 users
 MySQL	9180 users
 Apache Subversion	8680 users
 PHP	7789 users
 Linux Kernel	7142 users

**Most Active Projects**

 Alfresco Content Management Community Edition	19909 commits
 Arch Linux Packages	7343 commits
 Gentoo Linux	6280 commits
 azure-content	4740 commits
 Mozilla Firefox	4558 commits

**Most Active Contributors**

 Felix Yan	7389 commits
 Translation updater bot	1440 commits
 Luca suriano	1159 commits
 Manfred Touron	546 commits
 Mischa ter Smitten	501 commits
 Ori Livneh	501 commits

전세계 각종 오픈소스 통계 사이트 (<https://www.openhub.net/>)

# Opensource 각종 오픈소스에 기여 통계



오픈소스 커미터 개인별 통계 뷰 (<https://www.openhub.net/accounts/namhyung>)

# Opensource – Contributor, Committer, Maintainer

오픈소스 컨트리뷰션의 효과

실력, 프로개발자, 코어개발자, 커미터

행복한 개발자



(참고)

<https://wiki.kldp.org/wiki.php/KoreanOpenSourceCommitter>

<http://dirkriehle.com/publications/2014-2/the-open-source-software-developer-career-and-its-benefits/>

[http://www.payscale.com/research/US/Skill=Open\\_Source/Salary](http://www.payscale.com/research/US/Skill=Open_Source/Salary)

<http://nolongernew.blogspot.kr/2010/02/economic-motivation-of-open-source.html>



Git / GitHub Training

Contents



# Chapter 7. 프로젝트와 Git 운영전략

# 프로젝트와 Git 운용전략

프로젝트명 : salady

팀원 : Taeung, Irvanda ...

salady

# 프로젝트와 Git 운용전략

프로젝트명 : salady

팀원 : Taeung, Irvanda ...

salady

Fork !!

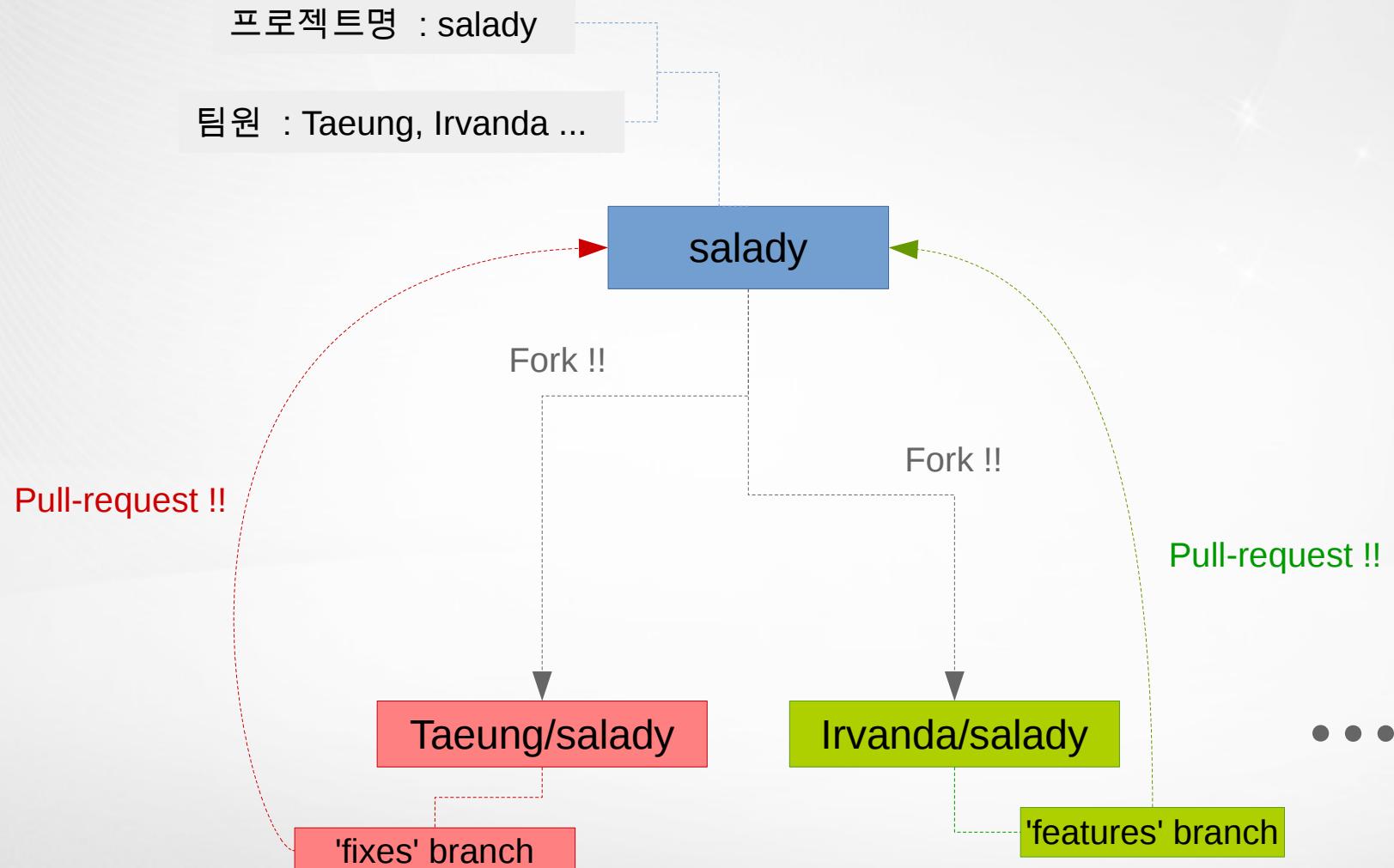
Fork !!

Taeung/salady

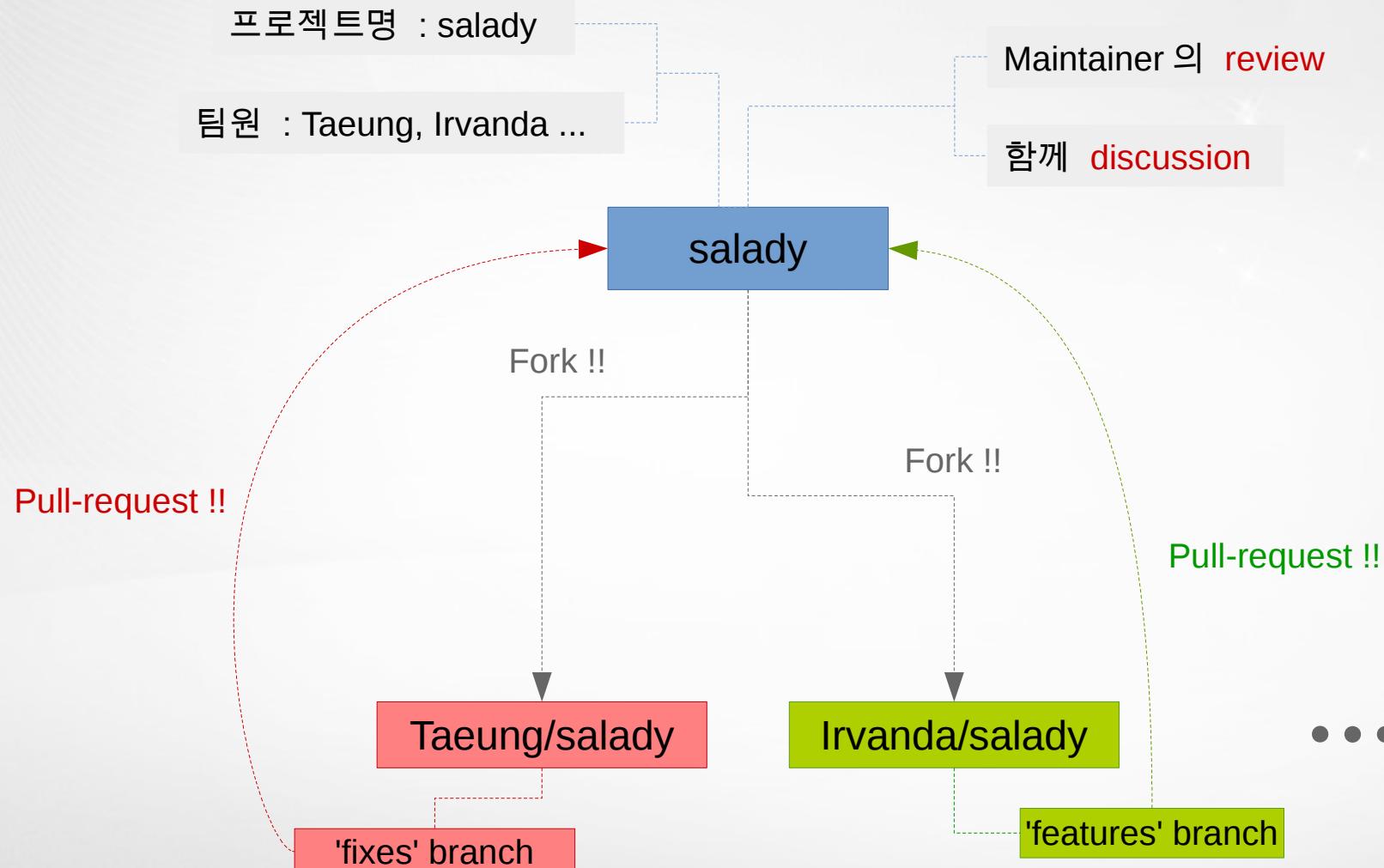
Irvanda/salady

• • •

# 프로젝트와 Git 운용전략



# 프로젝트와 Git 운용전략





Git / GitHub Training

Contents



# Chapter 8. Git 관련 자주 묻는 질문과 대답

## Git 관련 자주 묻는 질문과 대답

### 1) Git 과 Github 의 차이는 ?

Git 은 각 컴퓨터 (**local**) 에 설치되어 소스코드관리가 가능한 프로그램이고  
Github 는 **remote** 저장소가 있는 외부서버를 지칭한다 .

### 2) Commit 과 Push 의 차이는 ?

commit 은 **local** 작업폴더에 history 를 쌓는것이어서 외부망 (internet) 을 안쓰고  
Push는 **remote** 저장소 (Github 등 ) 에 history 를 쌓는것이어서 외부망 (intenet) 이 필요하다 .

### 3) Fetch 와 Pull 의 차이는 ?

Remote 저장소 (Github 등 ) 로 부터 최신 commit 정보들을 가져오는것은 매한가지이나  
Fetch 는 가져와서 임시폴더 (.git) 에 저장하고  
Pull 은 바로 현재 branch 에 merge 작업을 동반한다 .

## Git 관련 자주 묻는 질문과 대답

### 4) Rebase 와 Merge 의 차이는 ?

둘다 두 branch 의 차이점 (commits) 을 합치는 것은 매한가지나  
Rebase 는 합치기 전에 **되감기** (rewinding) 를 하고  
Merge 는 **안하고** 합친다 .

### 5) Branch 명령엔 브랜치 변경 기능이 왜 없을까 ? Checkout 의 의미 ?

Checkout 이라는 의미는 이전 VCS 에서 부터 쓰이던 용어로 '도서관 대출' 의미를 갖고  
이것을 통해서 브랜치를 Git 시스템으로 부터 가져온다 .

## Git 관련 자주 묻는 질문과 대답

### 6) Reset 과 Stash 의 차이 ?

Stash 는 특정 add 한 목록 (index) 를 **임시 저장**을 해두는 기능이고  
Reset 은 add 한 목록 (index) 를 **삭제**할때 이용할 수 있다 .

### 7) Revert 와 Reset 의 차이 ?

Revert 는 특정 commit 에 대한 **취소 commit** 을 만들어내는것이고  
Reset 은 **특정 commit** 삭제할때 이용할 수 있다 .

### 8) Checkout -- <file name> 과 Reset 의 차이 ?

Checkout 은 도서관에서 대출받듯 git 이라는 시스템을 통해서 파일 또는 브랜치 등을  
가져오는 기능을 한다 . Reset 은 add 되어 만들어진 리스트 (index) 를 지울 수 있다 .



Git / GitHub Training

How to use git



Git 간단한 정의 / 기능



Git 기본 실습  
(How)



Git 이해하기  
(Why, What)



Git 고급  
(Advanced)



Opensource  
(with Git)



Git / GitHub Training

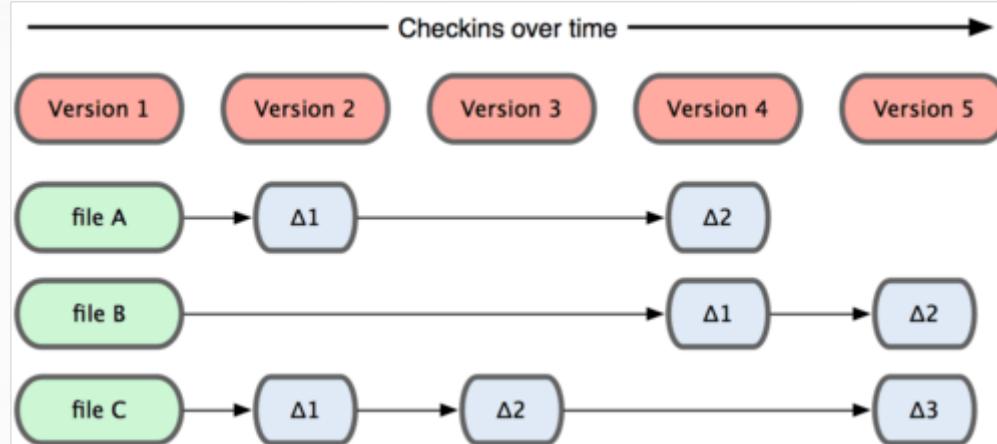
Contents



# Chapter 9. Branch 운용방식과 Checkout 의 의미

## Git에서 파일을 관리하는 원리

기존 VCS 시스템은 시간순으로 변화(delta)분을 저장, 관리하는 방식

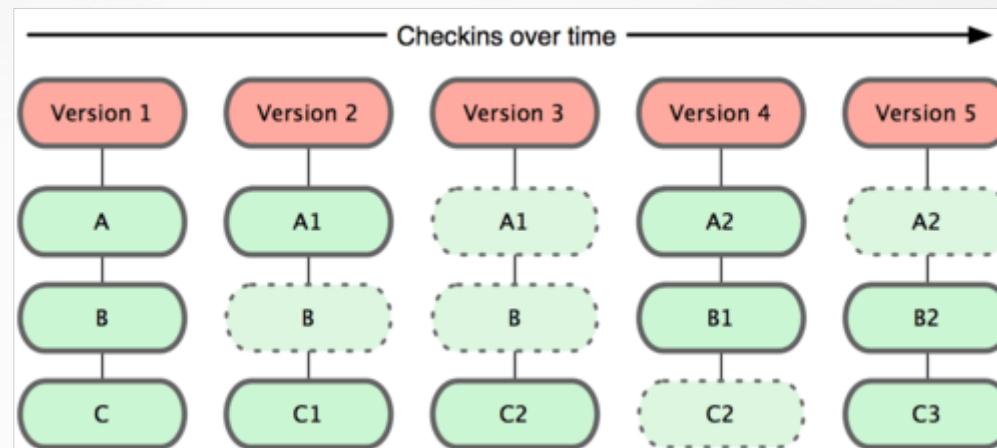


<https://git-scm.com/book/ko/v2/%EC%8B%9C%EC%91%ED%95%98%EA%B8%B0-Git-%EA%B8%B0%EC%B4%88>

Subversion 또는 비슷한 기타 VCS 와 Git 의 가장 큰 차이점은 데이터를 저장 방법에 있다

## Git에서 파일을 관리하는 원리

Git의 데이터는 파일 시스템의 **스냅샷**이라 할 수 있다



<https://git-scm.com/book/ko/v2/%EC%8B%9C%EC%9E%91%ED%95%98%EA%B8%B0-Git-%EA%B8%B0%EC%B4%88>

파일이 **달라지지 않았으면** Git은 성능을 위해서 파일을 **저장하지 않고**  
단지 이전 상태의 파일에 대한 **링크만 저장한다**

# Branch 란 ?

Git 은 데이터를 Change Set 이나 변경사항 (Diff) 으로  
기록하지 않고 일련의 **스냅샷으로 기록**

Git 의 브랜치는 커밋 사이를 **가볍게 이동**할 수 있는 어떤 **포인터** 같은 것

# Checkout 의 의미

“Check out” 도서관에서 대출의미로 이해 해보자

\* 참고 URL (<http://english.stackexchange.com/questions/211634/what-checkout-is-supposed-to-mean-in-git-checkout-sentence>)

1) 현재 브랜치의 최신 commit 역사까지의 기준으로 특정파일을 가져온다 (특정파일복구)

```
# git checkout -- <file name>
```

2) git 시스템을 이용하여 develop이라는 브랜치를 가져온다 (브랜치 변경)

```
# git checkout develop
```

# Merge로 2개의 브랜치 합치기

1) 방금작업한 develop 브랜치가 현재 브랜치인지 확인하자 (**status** 로도 확인 가능)

```
# git branch
```

2) 추가 브랜치 만들어보자

```
# git checkout -b test
```

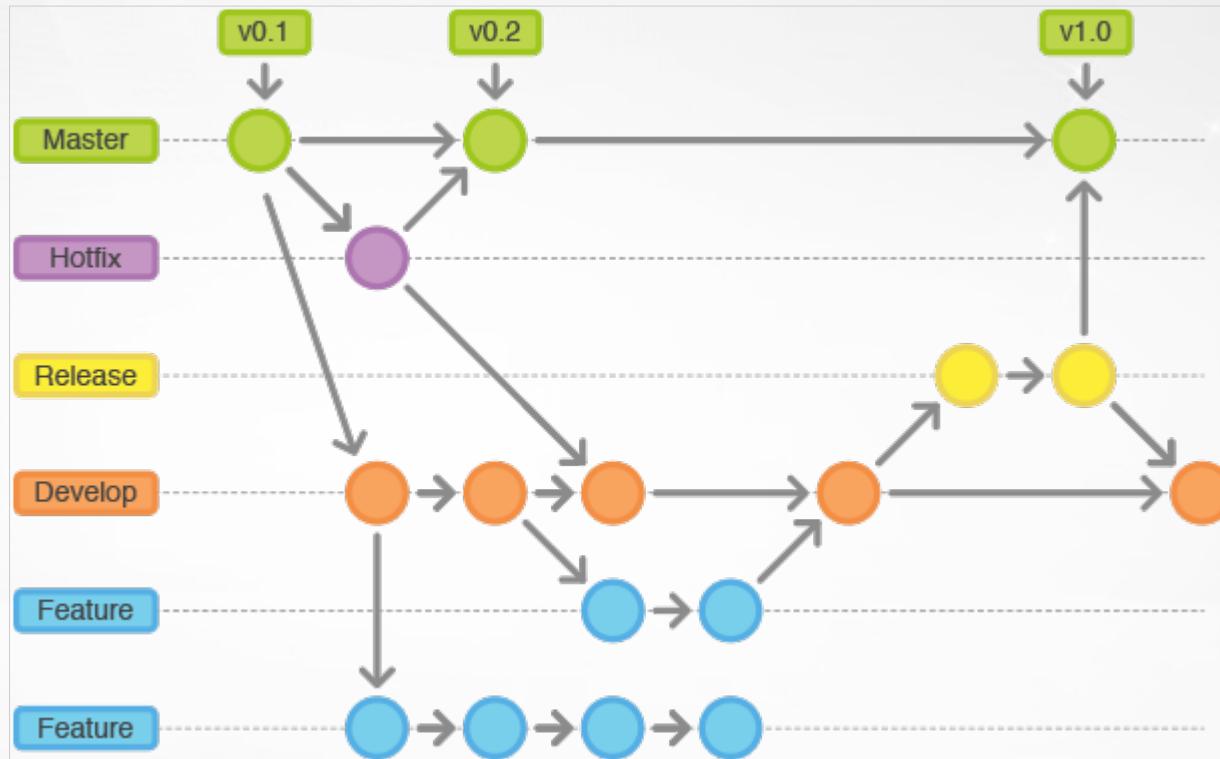
3) touch로 빈파일하나 만들어서 commit 만들어보자 (;로 명령어들을 연속적 실행 가능)

```
# touch test; git add test; git commit -sm "test"
```

4) 현재브랜치(develop)을 기준으로 추가브랜치(test)을 합치자

```
# git checkout develop; git status; git merge test
```

# Branch 운용관리 Git Flow



<http://stackoverflow.com/questions/35419185/git-workflow-example>



Git / GitHub Training

Contents



# Chapter 10. Rebase

# Rebase 기능

Rebase 사용하는 시나리오 !!

commit 을 역사의 한단위 '블럭' 이라 하고 블럭들의 모임을 'tree' 라 할때

내가 쌓은 블럭을 잠시 빼고

(뺀 나머지) 기준이 되는 tree 를 최신 업데이트 한 후에

그 위에 다시 내 블럭을 쌓아 올릴때 쓸 수 있다.

# Stage 10. Rebase 하기

1) Chapter 4. 의 Stage 8. 에서 fork 후 clone 했던 프로젝트경로로 이동하여 upstream 을 추가하자

```
# git remote add upstream https://github.com/Taeung/git-training.git
```

2) upstream 의 dev 브랜치를 가져오자

```
# git fetch upstream dev
```

3) 현재 내 브랜치가 develop 인지 확인하자

```
# git status; git branch
```

4) rebase 하자

```
# git rebase upstream/dev
```

# Stage 11. 중간에 낸 Commit 수정하기

고난이도 기능 중 하나 rebase --interactive

- 1) commit 최초기록 부터 2 번째 commit 을 수정해보자

```
# git rebase -i --root
```

- 2) vi 에디터가 열리면 수정하려는 commit(2 번째) 앞에 edit 이라 적자

- 3) 상태확인해서 rebase 진행 정상적인지 보고

```
# git status
```

- 4) commit 정보 수정( “packing knapsack:” commit 메시지에 추가) 하고 --continue 로 마무리

```
# git commit --amend -sm "packing knapsack: Add knapsack problem PDF"
```

```
# git rebase --continue
```

# 안전한 Rebase 방법

- 1) 먼저 upstream 으로 부터 최신 commit 들을 .git 으로 받아온다

```
# git fetch upstream
```

- 2) 현재 작업한 commit 이 7 개라면 8 번째를 edit 지정하고 되감는다

```
# git rebase -i HEAD~8
```

- 3) 작업을 시작지점을 강제 업데이트한다

```
# git reset --hard upstream/master
```

- 4) 작업 시작지점이 업데이트 되었다면 continue 로 되감았던 내용을 풀어낸다

```
# git rebase --continue
```



Git / GitHub Training

Contents



# Chapter 11. Stash, Clean, Config

# Stash 기능

Stash 란 add 하지 않은 변경사항 임시저장 해두는 기능

```
taeung ~/git/linux-perf/tools/perf
:> git status
On branch perf/config/default-value-v6
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
    (use "git checkout -- <file>..." to discard changes in working
     directory)

      modified:   util/config.c
      modified:   util/config.h

no changes added to commit (use "git add" and/or "git commit -a")
```

# Stash 실습

1) add 작업이 이루어 지지 않은 변경사항에 대해서 임시저장을 한다

```
# git stash
```

2) 현재 임시저장된 사항에 대한 리스트를 확인한다

```
# git stash list
```

3) 특정 stash 기록을 적용한다

```
# git stash apply stash@{0}
```

4) 특정 stash 기록을 삭제한다

```
# git stash drop stash@{0}
```

# Clean 기능

추적되지 않거나 필요없는 파일 모두 **청소**하는 기능

```
taeung ~/git/linux-perf/tools/perf
:> git status
On branch perf/config/default-value-v6
Untracked files:
  (use "git add <file>..." to include in what will be committed)

    needless/
    unnecessary

nothing added to commit but untracked files present (use "git add" to track)
```

# Clean 실습

1) 추적되지 않는 불필요한 파일을 삭제한다

```
# git clean -n
```

2) 강제옵션을 주고 삭제할 수도 있다

```
# git clean -f
```

3) 파일들 뿐만아니라 디렉토리(폴더)도 삭제를 원할때는 -d 옵션을 포함한다

```
# git clean -f -d
```

# Config 기능과 설정파일 적용순서

Local Config : .git/config

Global Config : \$HOME/.gitconfig

System Config : /etc/gitconfig

1) 모든 설정 내용을 출력한다

```
# git config -a
```

2) system 기준 config 파일 내용만 확인한다

```
# git config --system --list
```

3) global 기준 (\$HOME/.gitconfig) 에 user 이름 설정한다

```
# git config --global user.name "Taeung Song"
```



Git / GitHub Training

Contents



# Chapter 12. Blame, Cherry-pick, Submodule

해당 소스라인 대해서 누가 마지막으로 수정을 했는지

commit ID 추적이 가능하다.

```

07800601 Documentation/perf_counter/config.c (Ingo Molnar
5f9273d6 tools/perf/util/config.c          (Namhyung Kim
5f9273d6 tools/perf/util/config.c          (Namhyung Kim
5f9273d6 tools/perf/util/config.c          (Namhyung Kim
5f9273d6 tools/perf/util/config.c          (Namhyung Kim
07800601 Documentation/perf_counter/config.c (Ingo Molnar
4b6ab94e tools/perf/util/config.c          (Josh Poimboeuf
0b93da17 tools/perf/util/config.c          (Namhyung Kim
aa61fd05 tools/perf/util/config.c          (Wang Nan
20105ca1 tools/perf/util/config.c          (Taeung Song
07800601 Documentation/perf_counter/config.c (Ingo Molnar

```

```

2009-04-20 15:00:56 +0200 1) /*
2011-12-13 22:52:03 +0900 2) * config.c
2011-12-13 22:52:03 +0900 3) *
2011-12-13 22:52:03 +0900 4) * Helper functions for p..
2011-12-13 22:52:03 +0900 5) * Originally copied from..
2009-04-20 15:00:56 +0200 6) *
2009-04-20 15:00:56 +0200 7) * Copyright (C) Linus Torval..
2009-04-20 15:00:56 +0200 8) * Copyright (C) Johannes S..
2009-04-20 15:00:56 +0200 9) *
2009-04-20 15:00:56 +0200 10) */
2009-04-20 15:00:56 +0200 11) #include "util.h"
2009-04-20 15:00:56 +0200 12) #include "cache.h"
2015-12-15 09:39:39 -0600 13) #include <subcmd/exec-cmd.h>
2014-01-14 12:02:15 +0900 14) #include "util/hist.h"
2015-07-21 11:13:34 +0000 15) #include "util/llvm-utils.h"
2016-04-14 16:53:18 +0900 16) #include "config.h"
2009-04-20 15:00:56 +0200 17)

```

1) 어떤 파일이든 누가 어느 라인을 수정했는지 파악해보자

```
# git blame report_card.c
```

2) 해당 commit ID를 이용하여 그 당시 commit 정보확인

```
# git show <commit ID>
```

## Cherry-pick 기능

특정 commit 혹은 여러개의 commit 을 가져올때 사용하는 기능

주로 다른 브랜치에 존재하는 commit 을 가져올때 사용한다 .

# Cherry-pick 실습

1) 하나의 커밋을 가져오기

```
# git cherry-pick <commit ID>
```

2) 범위 지정하여 여러개의 커밋 가져오기 “git cherry-pick (더오래된커밋)..(기준커밋)”

```
# git cherry-pick A..B
```

3) 범위 지정하여 여러개의 커밋 가져오기 (브랜치 명으로 예시)

```
# git cherry-pick perf-infra-v9~2..perf-infra-v9
```

## Submodule 기능

프로젝트를 수행하다 보면 다른 프로젝트를 함께 사용해야 하는 경우가 종종 있다.

Git 저장소 안에 다른 Git 저장소를 디렉토리로 분리해 넣는 것이 서브모듈이다.

# Submodule 실습

1) 서브모듈 추가하기

```
# git submodule add <추가 repository URL>
```

2) 추가한 모듈에 대해서 확인해보자

```
# cat .gitmodules
```

# Submodule 실습

1) 서브모듈을 포함하는 프로젝트 clone 하기

```
# git clone <해당 프로젝트 repository URL>
```

2) 포함된 서브모듈정보를 확인해본다

```
# cat .gitmodules
```

3) 원래 포함해야하는 서브모듈에 대한 초기화

```
# git submodule init
```

4) 비어있던 서브모듈 소스 받아오기 (업데이트)

```
# git submodule update
```



Git / GitHub Training

## How to use git



Git 간단한 정의 / 기능

Git 기본 실습  
(How)

Git 이해하기  
(Why, What)

Git 고급  
(Advanced)

Opensource  
(with Git)



Git / GitHub Training

Contents



# Chapter 13. Git 을 활용해 여러가지 오픈소스 살펴보기

# Openhub 로 다양한 오픈소스 찾아보기

The screenshot shows the homepage of Black Duck Open Hub. At the top, there's a navigation bar with links for PROJECTS, PEOPLE, ORGANIZATIONS, TOOLS, and BLOG. On the right side of the header are buttons for 'Follow @ OH', 'SIGN IN', and 'JOIN NOW'. Below the header, a large blue banner features the text 'Discover, Track and Compare Open Source' and a search bar with the placeholder 'Search Projects...'. Below the search bar, it says 'Connecting 3,764,597 open source contributors'. On the left, under 'Join Now', there are three options: 'Claim your contributions' (with a green icon), 'Manage your project's data' (with a blue icon), and 'Highlight your use of FOSS' (with a yellow icon). A large green 'Join Now' button is at the bottom. On the right, under 'What's New', there's a dark blue box with the text 'What's going on with Open Hub in 2016' and a 'Learn more...' link.

<https://www.openhub.net/>

# Openhub 로 다양한 오픈소스 찾아보기

**BLACKDUCK | Open HUB**

[PROJECTS](#) [PEOPLE](#) [ORGANIZATIONS](#) [TOOLS](#) [BLOG](#)

[Follow @ OH](#) [SIGN IN](#) [JOIN NOW](#)

[Projects](#)

**Apache HTTP Server**

[Settings](#) | [Report Duplicate](#)

88%  
240 Project Vulnerability Report  
Activity Not Available  
9,340 I Use This!

**Project Summary**

The Apache HTTP Server Project is a collaborative software development effort aimed at creating a robust, commercial-grade, feature-rich, and freely-available source code implementation of an HTTP (Web) server. The project is jointly managed by a group of volunteers located around the world, using the Internet and the Web to communicate, plan, and develop the server and its related documentation. This project is part of the Apache Software Foundation. In addition, hundreds of users have contributed ideas, code, and documentation to the project.

**Tags**

gateway, dynamic\_content, ssl, plugin, intranet, ftpserver, cgi, http\_server, http, webserver, dav, authorization, xml, apache, webdav, ftpd, web, scgi, server, https, fastcgi, sni, httpd, isapi, proxy, ftp, modular, caching, ldap, authentication, tls, html, internet

**Quick Reference**

**Organization:** Apache Software Foundation

**Project Links:** [Homepage](#), [Documentation \(2 Links\)](#), [Download](#), [Forums](#), [Issue Trackers \(2 Links\)](#)

**Code Locations:** (3 Locations)

**Licenses:** Apache-2.0

**Similar Projects:** NGINX, nginx, Nanoweb - Th..., Hiawatha, web..., Roxen

**Managers:** Eric Covener, Jim Jagielski, and William A. Rowe Jr.

<https://www.openhub.net/p/apache>

# Openhub 로 다양한 오픈소스 찾아보기

## In a Nutshell, Apache HTTP Server...

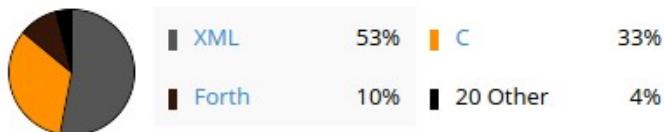
... has had 76,803 commits made by 120 contributors representing 1,808,624 lines of code

... is mostly written in C with an average number of source code comments

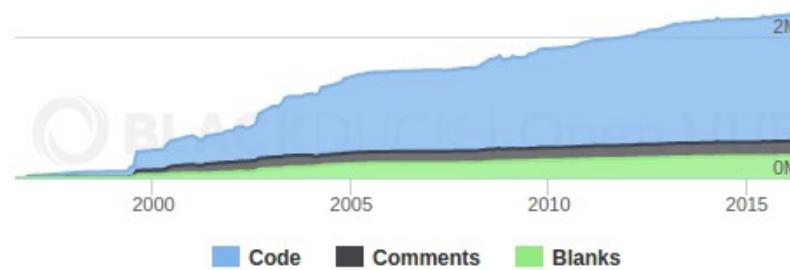
... has a well established, mature codebase maintained by a very large development team with increasing Y-O-Y commits

... took an estimated 508 years of effort (COCOMO model) starting with its first commit in July, 1996 ending with its most recent commit about 2 months ago

## Languages



## Lines of Code



## Activity

### 30 Day Summary

Apr 24 2016 — May 24 2016

257 Commits

16 Contributors  
including 1 new contributor

### 12 Month Summary

May 24 2015 — May 24 2016

2774 Commits

Up + 450 (19%) from previous 12 months

28 Contributors

Down 0 (0%) from previous 12 months

## Commits per Month

Zoom 1yr 3yr 5yr 10yr All



<https://www.openhub.net/p/apache>

# Github에서 다양한 오픈소스 찾아보기

Explore GitHub

Showcases Integrations Trending Stars

## Trending in open source

See what the GitHub community is most excited about today.

Repositories Developers Trending: today ▾ All languages

zeit/hyperterm Unknown languages

JavaScript • 400 stars today • Built by  C

 Star 

goldfire/howler.js ProTip! Looking for most forked JavaScript repositories? Try this search

Javascript audio library for the modern web.

JavaScript • 293 stars today • Built by 



<https://github.com/trending/javascript>

# Git 을 활용한 다양한 오픈소스 분석

1) 해당 프로젝트 clone 을 한다

```
# git clone <repository URL>
```

2) shortlog 명령을 이용해서 통계를 내보자

```
# git shortlog
```

3) tig, gitk, log 명령등으로 commit 들을 살펴보자

```
# tig
```

4) 궁금하거나 영향력있는 개발자의 commit 만 살펴보자

```
# tig --author="Taeung Song"
```

# Linux Kernel 오픈소스 살펴보기

 Kernel.org git repositories  
Git repositories hosted at kernel.org

<a href="#">index</a>		<a href="#">search</a>		
Name	Description	Owner	Idle	Links
<i>bluetooth</i>				
bluez-hcidump.git	Bluetooth packet analyzer	Marcel Holtmann	3 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
bluez.git	Bluetooth protocol stack for Linux	Marcel Holtmann	26 hours	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
obexd.git	OBEX Server	Marcel Holtmann	3 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
sbc.git	SBC library	holtmann	17 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
<i>boot</i>				
dracut/dracut.git	dracut - Initramfs generator using udev	Harald Hoyer	8 days	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
efilinux/efilinux.git	The efilinux UEFI boot loader	Matt Fleming	21 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
syslinux/syslinux.git	The Syslinux boot loader suite	Syslinux workgroup	18 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
<i>devel</i>				
pahole/pahole.git	Pahole and other DWARF utils	Arnaldo Carvalho de Melo	10 days	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
sparse/chrisl/sparse.git	Chris Li's sparse repository.	Christopher Li	5 weeks	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
sparse/sparse.git	C semantic parser	Christopher Li	13 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
<i>docs</i>				
kernel/kernel-docs.git	Kernel Documentation tree	Doc Group	2 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
kernel/ksmap.git	Kernel.org keysign map source	Kernel.org users	4 days	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
kernel/website.git	Kernel.org website source	Doc Group	13 days	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
man-pages/man-pages.git	Linux man pages Sections 2, 3, 4, 5, and 7	Michael Timothy Kerrisk	9 days	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
man-pages/website.git	Website files for /doc/man-pages	Michael Timothy Kerrisk	2 weeks	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
<i>editors</i>				
uemacs/uemacs.git	Micro-emacs	Linus Torvalds	16 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
<i>fs</i>				
ext2/e2fsprogs.git	Ext2/3/4 filesystem userspace utilities	Theodore T'so	3 days	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
ext2/xfstests-bld.git	Build framework and autorun scripts for xfstests	Theodore T'so	4 weeks	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
fat/fatattr/fatattr.git	FAT attribute set utility	H. Peter Anvin	7 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
fuse/dbfs.git	FUSE fs w/ Berkeley DB backend.	Jeff Garzik	7 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
fuse/fuse-ext2.git	FUSE ext2 filesystem driver.	Jeff Garzik	10 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
squashfs/squashfs-tools.git	squashfs tools development	Phillip Louher	4 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
xfs/dmapi-dev.git	Data Management API runtime environment	Christoph Hellwig	5 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
xfs/xfs-documentation.git	XFS AsciiDoc Documentation tree	XFS FS Group	3 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>

<https://git.kernel.org/cgit>

# Git 으로 Linux Kernel 오픈소스 살펴보기

1) Linux Kernel 공식 repository URL로 프로젝트 받기

```
# git clone git://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git
```

2) shortlog 명령을 이용해서 통계를 내보자

```
# git shortlog
```

3) tig, gitk, log 명령등으로 commit들을 살펴보자

```
# tig
```

4) 궁금하거나 영향력있는 개발자의 commit만 살펴보자

```
# tig --author="Namhyung Kim"
```

# Node.js 오픈소스 살펴보기

nodejs / node

Watch 1,743 Star 25,038 Fork 3,790

Code Issues 482 Pull requests 253 Wiki Pulse Graphs

Node.js JavaScript runtime 🎉🐢🚀🎉 <https://nodejs.org>

14,464 commits 124 branches 377 releases 960 contributors

Branch: master New pull request Create new file Upload files Find file Clone or download

thefourtheye committed with Trott test: make import common as the first line ... Latest commit 6123075 3 days ago

.github doc: minor rewording to the GitHub issue/pr templates 25 days ago

benchmark buffer: optimize hex\_decode 5 days ago

<https://github.com/nodejs/node>

# Git 으로 Node.js 오픈소스 살펴보기

1) Linux Kernel 공식 repository URL로 프로젝트 받기

```
# git clone git://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git
```

2) shortlog 명령을 이용해서 통계를 내보자

```
# git shortlog
```

3) tig, gitk, log 명령등으로 commit들을 살펴보자

```
# tig
```

4) 궁금하거나 영향력있는 개발자의 commit만 살펴보자

```
# tig --author="Ben Noordhuis"
```

# Springboot 오픈소스 살펴보기

The screenshot shows the GitHub repository page for `spring-projects/spring-boot`. The page includes navigation links for Code, Issues (469), Pull requests (42), Wiki, Pulse, and Graphs. Key statistics are displayed: 8,459 commits, 7 branches, 61 releases, and 279 contributors. A pull request button and file management buttons (Create new file, Upload files, Find file, Clone or download) are also present. The repository's history shows recent activity from `wilkinsona`, including a merge pull request and upgrades to .github and .mvn files.

Commit	Description	Time Ago
wilkinsona Merge pull request #6437 from Johnny Lim	Merge pull request #6437 from Johnny Lim	16 hours ago
.github	Merge branch '1.3.x'	2 days ago
.mvn	Upgrade to Maven 3.3.9	3 months ago

<https://github.com/spring-projects/spring-boot>

# Git 으로 Springboot 오픈소스 살펴보기

1) Linux Kernel 공식 repository URL로 프로젝트 받기

```
# git clone https://github.com/spring-projects/spring-boot.git
```

2) shortlog 명령을 이용해서 통계를 내보자

```
# git shortlog
```

3) tig, gitk, log 명령등으로 commit들을 살펴보자

```
# tig
```

4) 궁금하거나 영향력있는 개발자의 commit만 살펴보자

```
# tig --author="Phil Webb"
```

# React 오픈소스 살펴보기

 facebook / react

Watch 3,277 Star 45,823 Fork 7,977

Code Issues 504 Pull requests 96 Wiki Pulse Graphs

A declarative, efficient, and flexible JavaScript library for building user interfaces. <https://facebook.github.io/react/>

6,979 commits 22 branches 48 releases 747 contributors

Branch: master ▾ New pull request Create new file Upload files Find file Clone or download ▾

keyanzhang committed with gaearon improved warning in ReactUpdateQueue (#7326) Latest commit 3fd5826 20 hours ago

.github Add basic issue and PR templates (#6597) 2 months ago

docs Finish Jekyll 3 Upgrade 2 days ago

<https://github.com/facebook/react>

# Git 으로 React 오픈소스 살펴보기

1) Linux Kernel 공식 repository URL로 프로젝트 받기

```
# git clone https://github.com/facebook/react.git
```

2) shortlog 명령을 이용해서 통계를 내보자

```
# git shortlog
```

3) tig, gitk, log 명령등으로 commit들을 살펴보자

```
# tig
```

4) 궁금하거나 영향력있는 개발자의 commit만 살펴보자

```
# tig --author="Ben Alpert"
```

# Nuclide 오픈소스 살펴보기

 [facebook / nuclide](#)

[Watch ▾ 255](#) [Star 4,743](#) [Fork 317](#)

[Code](#) [Issues 175](#) [Pull requests 4](#) [Wiki](#) [Pulse](#) [Graphs](#)

An open IDE for web and native mobile development, built on top of Atom <http://nuclide.io>

 5,581 commits  3 branches  69 releases  89 contributors

Branch: master ▾ [New pull request](#) [Create new file](#) [Upload files](#) [Find file](#) [Clone or download ▾](#)

 Derrick J. Bonafilia committed with Facebook Github Bot 7 Added the ability to create datatips without a service/provider ... Latest commit bce3294 an hour ago

 docs Add troubleshooting information for module not found errors a day ago

 flow-typed Update defs from flow-typed 15 days ago

<https://github.com/facebook/nuclide>

# Git 으로 Nuclide 오픈소스 살펴보기

1) Linux Kernel 공식 repository URL로 프로젝트 받기

```
# git clone https://github.com/facebook/nuclide.git
```

2) shortlog 명령을 이용해서 통계를 내보자

```
# git shortlog
```

3) tig, gitk, log 명령등으로 commit들을 살펴보자

```
# tig
```

4) 궁금하거나 영향력있는 개발자의 commit만 살펴보자

```
# tig --author="Chen Shen"
```



Git / GitHub Training

Contents



# Chapter 14. Linux Kernel Contribution

# 해당되는 Linux Kernel 공식 repository

 Kernel.org git repositories  
Git repositories hosted at kernel.org

Name	Description	Owner	Idle	Links
<i>bluetooth</i>				
bluez-hcidump.git	Bluetooth packet analyzer	Marcel Holtmann	3 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
bluez.git	Bluetooth protocol stack for Linux	Marcel Holtmann	26 hours	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
obexd.git	OBEX Server	Marcel Holtmann	3 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
sbc.git	SBC library	holtmann	17 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
<i>boot</i>				
dracut/dracut.git	dracut - Initramfs generator using udev	Harald Hoyer	8 days	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
efilinux/efilinux.git	The efilinux UEFI boot loader	Matt Fleming	21 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
syslinux/syslinux.git	The Syslinux boot loader suite	Syslinux workgroup	18 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
<i>devel</i>				
pahole/pahole.git	Pahole and other DWARF utils	Arnaldo Carvalho de Melo	10 days	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
sparse/chrisl/sparse.git	Chris Li's sparse repository.	Christopher Li	5 weeks	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
sparse/sparse.git	C semantic parser	Christopher Li	13 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
<i>docs</i>				
kernel/kernel-docs.git	Kernel Documentation tree	Doc Group	2 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
kernel/ksmap.git	Kernel.org keysign map source	Kernel.org users	4 days	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
kernel/website.git	Kernel.org website source	Doc Group	13 days	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
man-pages/man-pages.git	Linux man pages Sections 2, 3, 4, 5, and 7	Michael Timothy Kerrisk	9 days	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
man-pages/website.git	Website files for /doc/man-pages	Michael Timothy Kerrisk	2 weeks	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
<i>editors</i>				
uemacs/uemacs.git	Micro-emacs	Linus Torvalds	16 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
<i>fs</i>				
ext2/e2fsprogs.git	Ext2/3/4 filesystem userspace utilities	Theodore T'so	3 days	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
ext2/xfstests-bld.git	Build framework and autorun scripts for xfstests	Theodore T'so	4 weeks	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
fat/fatattr/fatattr.git	FAT attribute set utility	H. Peter Anvin	7 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
fuse/dbfs.git	FUSE fs w/ Berkeley DB backend.	Jeff Garzik	7 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
fuse/fuse-ext2.git	FUSE ext2 filesystem driver.	Jeff Garzik	10 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
squashfs/squashfs-tools.git	squashfs tools development	Phillip Louher	4 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
xfs/dmapi-dev.git	Data Management API runtime environment	Christoph Hellwig	5 years	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>
xfs/xfs-documentation.git	XFS AsciiDoc Documentation tree	XFS FS Group	3 months	<a href="#">summary</a> <a href="#">log</a> <a href="#">tree</a>

<https://git.kernel.org/cgit>

# 해당되는 Linux Kernel 공식 repository

1) Mainline에서 최상위 공식 Repository인 리누스토발즈의 linux 프로젝트를 clone 한다

```
# git clone git://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git
```

2) 생성된 linux 폴더에서 MAINTAINERS 파일을 열어서 작업할 프로젝트의 공식 repository 주소를 확인하자

## List of maintainers and how to submit kernel changes

Please try to follow the guidelines below. This will make things easier on the maintainers. Not all of these guidelines matter for every trivial patch so apply some common sense.

...(omitted)

### PERFORMANCE EVENTS SUBSYSTEM

M: Peter Zijlstra <peterz@infradead.org>  
M: Ingo Molnar <mingo@redhat.com>  
M: Arnaldo Carvalho de Melo <acme@kernel.org>  
R: Alexander Shishkin <alexander.shishkin@linux.intel.com>  
L: linux-kernel@vger.kernel.org  
T: git git://git.kernel.org/pub/scm/linux/kernel/git/tip/tip.git perf/core  
S: Supported  
F: kernel/events/\*  
...(omitted)  
F: tools/perf/

# PATCH 작업 (contrubution) 거리 찾기

▶ 메인테이너가 쉽게 받아줄 거라  
예상되는 PATCH/Pull-request 는 ?

1 위 - Refactoring ( 가장 부담 zero)

2 위 - Bugfix ( 명확한 버그라면 )

3 위 - Documentation ( 첫 접근성은 좋으나 기술적인 description 이 문제 )

4 위 - Minor features

그 이후 new feature 나 core 부분은 신뢰가 필요하다고 봄 ( 다수의 commits )

# PATCH 파일 만들고 Code Style 검사

1) 해당 프로젝트 repository 주소로 clone 하거나 upstream 으로 등록한다

```
# git clone git://git.kernel.org/pub/scm/linux/kernel/git/tip/tip.git
```

2) 소스파일 추가 및 수정에 대해서 서명을 포함해서 commit 만든다

```
# git add <file> && git commit -s
```

3) 완성된 commit 을 다시한번 확인해본다

```
# git show
```

4) 가장 최신 커밋 1 개를 patch 파일로 만들어낸다

```
# git format-patch -1
```

5) 완성된 .patch 파일의 Code Style 을 검사 한다

```
# $LINUX/scripts/checkpatch.el something.patch
```

# PATCH 세트와 Cover letter

1) 여러개의 commit 을 PATCH 세트를 cover letter 와 함께 생성한다 ( 일곱번째 보내는것을 가정 )

```
# git format-patch -7 --cover-letter --subject-prefix="PATCH v7"
```

2) 함께 생성된 cover letter 에 대해 작성한다

```
# emacs 0000-cover-letter.patch
```

# Maintainer 찾기 및 PATCH mail 전송

1) MAINTAINERS 파일을 토대로 해당 프로젝트의 maintainer, reviewer 등을 찾는다

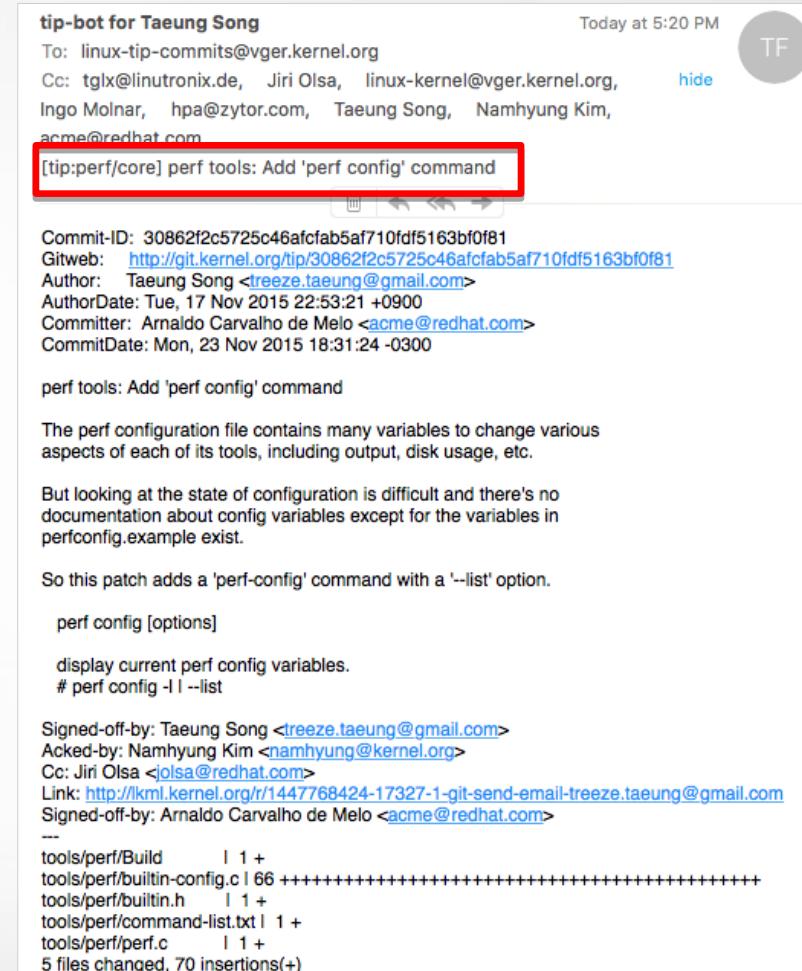
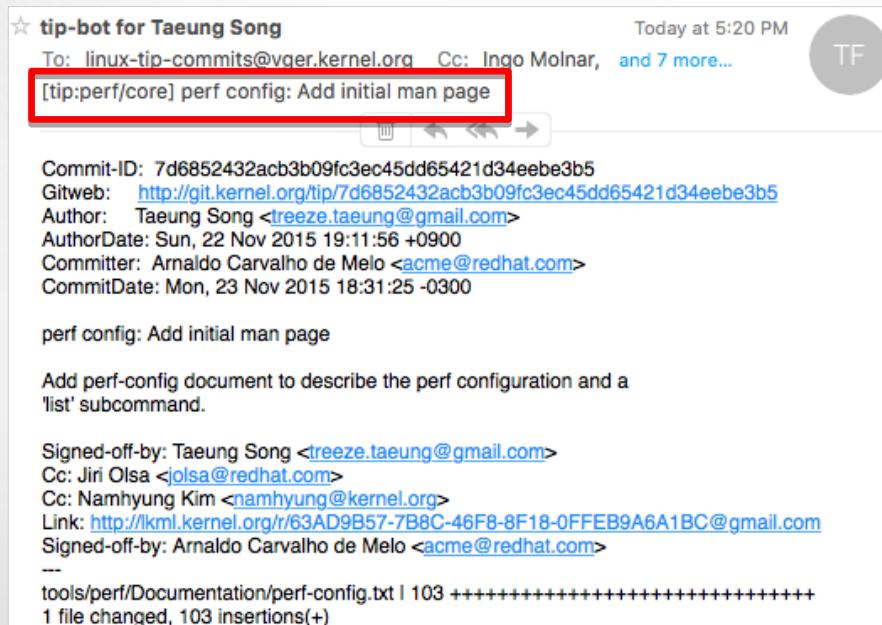
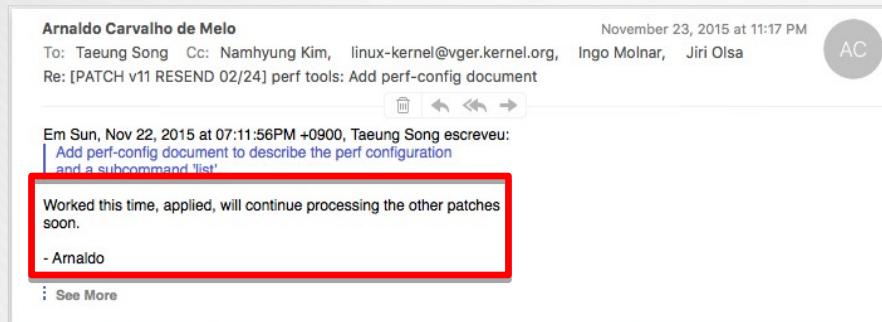
```
# emacs $LINUX/MAINTAINERS
```

2) PATCH 세트 혹은 파일을 메인테이너등에게 메일로 전송한다

```
# git send-email --confirm=never \
--to "Arnaldo Carvalho de Melo <acme@kernel.org>" \
--cc "linux-kernel@vger.kernel.org" \
--cc "Jiri Olsa <jolsa@kernel.org>" \
--cc "Namhyung Kim <namhyung@kernel.org>" \
--cc "Ingo Molnar <mingo@kernel.org>" \
--cc "Peter Zijlstra <peterz@infradead.org>" \
--cc "Alexander Shishkin <alexander.shishkin@linux.intel.com>" \
--cc "Masami Hiramatsu <mhiramat@kernel.org>" \
--cc "Wang Nan <wangnan0@huawei.com>" *.patch
```

# PATCH mail 개발 방식 소개

## 이 변화분 (commits)을 통한 Review 와 Discussion 가능 (pull-request, PATCH mail)





Git / GitHub Training

Contents



# Chapter 15. GitHub 기반 오픈소스 Contribution

# 오픈소스 ax5ui

The screenshot shows the GitHub repository page for 'ax5ui'. The header includes the 'AX5UI' logo, a status icon, and the text 'Javascript UI Plugins, Bootstrap & jQuery based'. Below the header are links to the website (<http://ax5.io>) and email (tom@axisj.com). The main navigation bar has 'Repositories' selected, showing a count of 3 people. A search bar with 'Filters' and a 'Find a repository...' placeholder is present. Two repositories are listed: 'ax5ui-kernel' and 'ax5docs'. 'ax5ui-kernel' is a JavaScript repository with 12 stars, 6 forks, and was updated 16 hours ago. 'ax5docs' is an HTML repository with 10 stars, 0 forks, and was updated 4 days ago. To the right, a 'People' sidebar lists three contributors: 'brant-hwang' (Brant), 'geminiKim' (Gemini), and 'thomasJang' (Thomas), each with a small profile picture.

**Repositories** **People 3**

**Filters** Find a repository...

**ax5ui-kernel**  
ax5ui Development Repository  
Updated 16 hours ago

JavaScript ★ 12 ⚡ 6

**ax5docs**  
ax5ui website - <http://ax5.io/>  
Updated 4 days ago

HTML ★ 10 ⚡ 0

**People** 3 >

- brant-hwang**  
Brant
- geminiKim**  
Gemini
- thomasJang**  
Thomas

<https://github.com/ax5ui>

# ax5ui 프로젝트 Fork 하기

ax5ui / ax5ui-kernel

Code Issues 2 Pull requests 0 Wiki Pulse Graphs

ax5ui Development Repository <http://ax5.io>

804 commits 1 branch 0 releases 6 contributors

Branch: master New pull request Create new file Upload files Find file Clone or download

thomasJang	ax5grid	Latest commit c83e993 16 hours ago
build	ax5combobox ~	27 days ago
dist	ax5media-viewer@0.3.0 반영	24 days ago
src	ax5grid	16 hours ago

Fork 버튼을 눌러서 프로젝트가 생성되었는지 확인하자 (본인 Github)

# Pull-request로 작업내용 제출

1) Fork 한 프로젝트를 내려받는다

```
# git clone https://github.com/<본인 ID>/ax5ui-kernel.git
```

2) develop 브랜치를 만든다

```
# cd ax5ui-kernel && git checkout -b develop
```

3) 소스파일 추가 및 수정에 대해서 서명을 포함해서 commit 만든다

```
# git add <file> && git commit -s
```

4) 완성된 commit 을 다시한번 확인해본다

```
# git show
```

5) 본인 fork 한 프로젝트에 push 를 하고 pull-request 진행

```
# git push origin develop
```

# Pull-request로 작업내용 예시

The screenshot shows a GitHub repository page for `ax5ui / ax5ui-kernel`. The main navigation bar includes links for Code, Issues (2), Pull requests (0), Wiki, Pulse, and Graphs. The Pull requests tab is currently selected. A prominent message "please check #8" is displayed above the pull request details. The pull request itself is titled "Merged" and describes merging 3 commits from `90MaJH:master` into `ax5ui:master` on Jun 8. The commit details show a check comment by `90MaJH` and two commits to `README.md`. The pull request has +76 additions and -48 deletions, with a green progress bar. On the right side, there are sections for Labels (None yet), Milestone (No milestone), Assignees (No one assigned), and Participants (2 participants, showing icons for `90MaJH` and another user).

<https://github.com/ax5ui/ax5ui-kernel/pull/8>

# Issue tracking 기능 활용

ax5ui / ax5ui-kernel

Watch 4 Star 12 Fork 6

Code Issues 2 Pull requests 0 Wiki Pulse Graphs

## dialog.alert ignorance on double call #11

New issue

Open EastskyKang opened this issue 17 days ago · 0 comments

EastskyKang commented 17 days ago • edited

+ 😊

```
dialog.alert([...], function() {
    dialog.alert([...], [...]);
});
```

위와 같이 dialog.alert 의 callback에서 다시 한번 dialog.alert 이 호출되면 두번째 alert 은 무시됩니다.

As a snippet above, when dialog.alert is called in a callback function of another dialog.alert, it does not work.

Labels  
None yet

Milestone  
No milestone

Assignees  
No one assigned

1 participant

<https://github.com/ax5ui/ax5ui-kernel/issues/11>

# Q & A

---

Git / Github Training