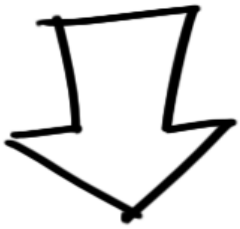
git - 간편 안내서

git을 시작하기 위한 간편 안내서. 어렵지 않아요 ;)

트윗

Roger Dudler가 만들었어요.
(@tfnico, @fhd와 Namics의 도움을 받았지요.)
번역은 Juntai Park과 Ardie Hwang이 담당했습니다.
문제 보고는 여기(github)로 해주세요!

다른 언어판도 읽어보세요!
English, Deutsch, Español, Français, indonesian, Italiano, 日本語,
မြန်မာ, Nederlands, polski, Português, Русский, Türkçe, 中文



설치

OS X용 git 다운로드

Windows용 git 다운로드

Linux용 git 다운로드

새로운 저장소 만들기

폴더를 하나 만들고, 그 안에서 아래 명령을 실행하세요.

```
git init
```

새로운 git 저장소가 만들어집니다.

저장소 받아오기

로컬 저장소를 복제(clone)하려면 아래 명령을 실행하세요.

```
git clone /로컬/저장소/경로
```

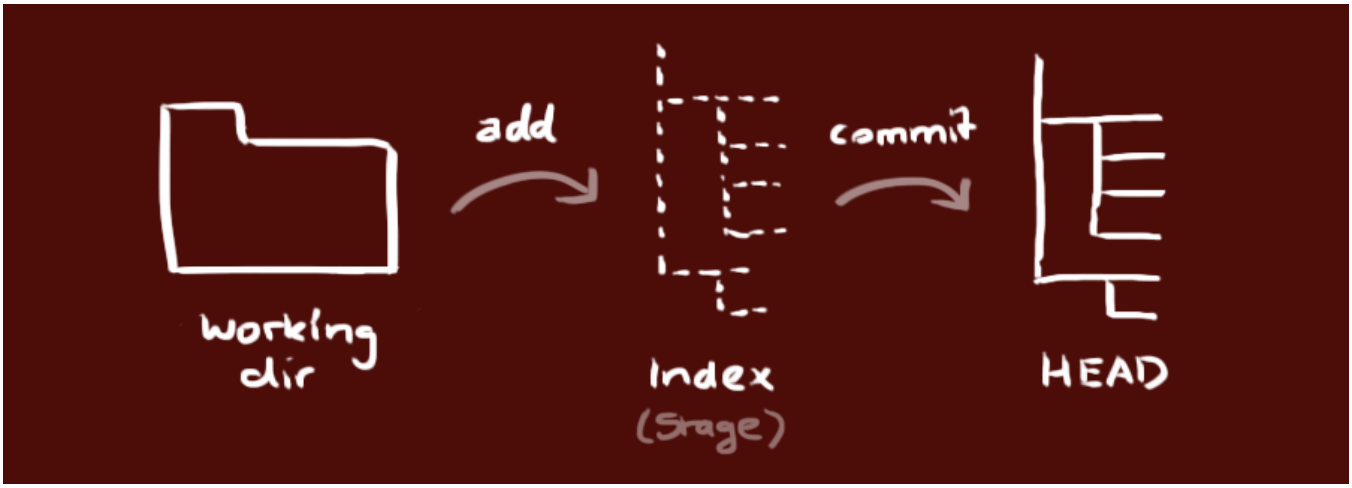
원격 서버의 저장소를 복제하려면 아래 명령을 실행하세요.

```
git clone 사용자명@호스트:/원격/저장소/경로
```

작업의 흐름

여러분의 로컬 저장소는 git이 관리하는 세 그루의 나무로 구성돼있어요.

첫번째 나무인 `작업 디렉토리(Working directory)` 는
실제 파일들로 이루어져있고, 두번째 나무인 `인덱스(Index)` 는
준비 영역(staging area)의 역할을 하며, 마지막 나무인 `HEAD` 는
최종 확정본(commit)을 나타내요.



추가와 확정(commit)

변경된 파일은 아래 명령어로 (**인덱스**에) 추가할 수 있어요.

```
git add <파일 이름>

git add *
```

이것이 바로 git의 기본 작업 흐름에서 첫 단계에 해당돼요.

하지만 실제로 변경 내용을 확정하려면 아래 명령을 내려야 한답니다.

```
git commit -m "이번 확정본에 대한 설명"
```

자, 이제 변경된 파일이 **HEAD**에 반영됐어요.

하지만, 원격 저장소에는 아직 반영이 안 됐답니다.

변경 내용 발행(push)하기

현재의 변경 내용은 아직 로컬 저장소의 **HEAD** 안에 머물고 있어요.

이제 이 변경 내용을 원격 서버로 올려봅시다. 아래 명령을 실행하세요.

```
git push origin master
```

(다른 가지를 발행하려면 *master*를 원하는 가지 이름으로 바꿔주세요.)

만약 기존에 있던 원격 저장소를 복제한 것이 아니라면,

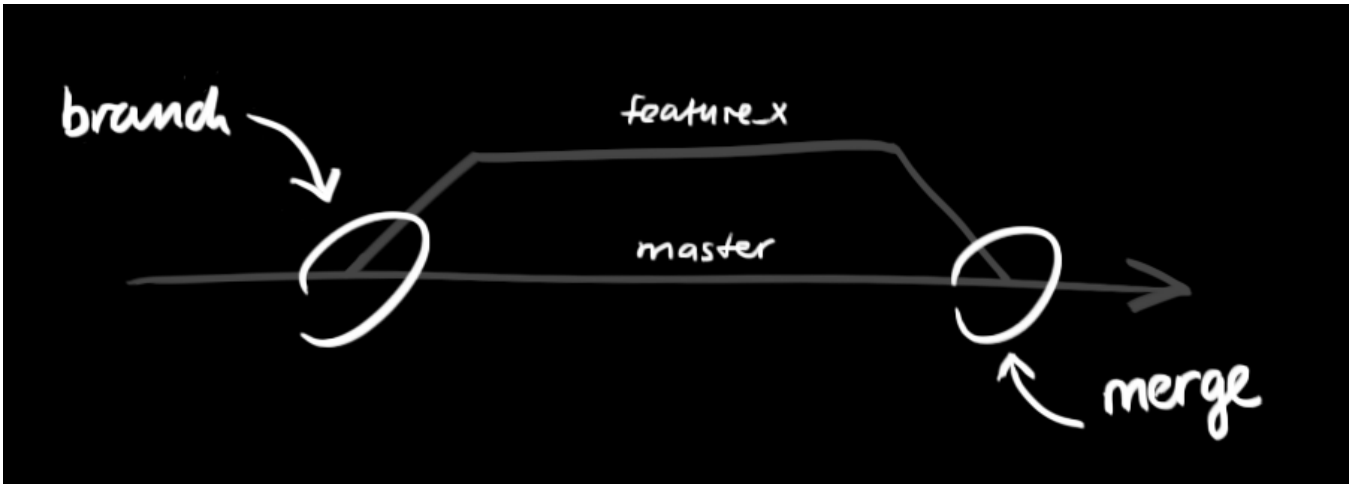
원격 서버의 주소를 git에게 알려줘야 해요.

```
git remote add origin <원격 서버 주소>
```

이제 변경 내용을 원격 서버로 발행할 수 있어요.

가지(branch)치기

가지는 안전하게 격리된 상태에서 무언가를 만들 때 사용해요.
여러분이 저장소를 새로 만들면 기본으로 *master 가지*가 만들어집니다.
이제 다른 가지를 이용해서 개발을 진행하고, 나중에 개발이 완료되면
master 가지로 돌아와 병합하면 돼요.



아래 명령으로 "feature_x"라는 이름의 가지를 만들고 갈아탑니다.

```
git checkout -b feature_x
```

아래 명령으로 master 가지로 돌아올 수 있어요.

```
git checkout master
```

아래 명령으로는 가지를 삭제할 수 있어요.

```
git branch -d feature_x
```

여러분이 새로 만든 가지를 원격 저장소로 전송하기 전까지는
다른 사람들이 접근할 수 없어요.

```
git push origin <가지 이름>
```

갱신과 병합(merge)

여러분의 로컬 저장소를 원격 저장소에 맞춰 갱신하려면
아래 명령을 실행하세요.

```
git pull
```

이렇게 하면 원격 저장소의 변경 내용이 로컬 작업 디렉토리에
받아지고(fetch), *병합(merge)*된답니다.
다른 가지에 있는 변경 내용을 현재 가지(예를 들면, master 가지)에
병합하려면 아래 명령을 실행하세요.

```
git merge <가지 이름>
```

첫번째 명령이든 두번째 명령이든, git은
자동으로 변경 내용을 병합하려고 시도해요.
문제는, 항상 성공하는 게 아니라 가끔
*충돌(conflicts)*이 일어나기도 한다는 거예요.
이렇게 충돌이 발생하면, git이 알려주는 파일의 충돌 부분을
여러분이 직접 수정해서 병합이 가능하도록 해야 하죠.
충돌을 해결했다면, 아래 명령으로 git에게
아까의 파일을 병합하라고 알려주세요.

```
git add <파일 이름>
```

변경 내용을 병합하기 전에, 어떻게 바뀌었는지 비교해볼 수도 있어요.

```
git diff <원래 가지> <비교 대상 가지>
```

꼬리표(tag) 달기

소프트웨어의 새 버전을 발표할 때마다 꼬리표를 달아놓으면 좋아요.

(물론 꼬리표는 SVN 등에 이미 존재하는 기능이지요.)

아래 명령을 실행하면 새로운 꼬리표인 *1.0.0*을 달 수 있어요.

```
git tag 1.0.0 1b2e1d63ff
```

위 명령에서 *1b2e1d63ff* 부분은 꼬리표가 가리킬 확정본 식별자입니다.

아래 명령으로 확정본 식별자를 얻을 수 있어요.

```
git log
```

확정본 식별자의 앞부분 일부만 입력해도 꼬리표를 붙일 수 있지만,

그 일부분이 반드시 고유하다는 조건이 필요해요.

로컬 변경 내용 되돌리기

만약 여러분이 (물론 그럴 일은 없겠지만 ;) 실수로 무언가 잘못한 경우,

아래 명령으로 로컬의 변경 내용을 되돌릴 수 있어요.

```
git checkout -- <파일 이름>
```

위 명령은 로컬의 변경 내용을 변경 전 상태(HEAD)로 되돌려줘요.

다만, 이미 인덱스에 추가된 변경 내용과

새로 생성한 파일은 그대로 남는답니다.

만약, 로컬에 있는 모든 변경 내용과 확정본을 포기하려면,

아래 명령으로 원격 저장소의 최신 이력을 가져오고,

로컬 master 가지가 저 이력을 가리키도록 할 수 있어요.

```
git fetch origin
git reset --hard origin/master
```

유용한 힌트

git의 내장 GUI

```
gitk
```

콘솔에서 git output을 컬러로 출력하기

```
git config color.ui true
```

이력(log)에서 확정본 1개를 딱 한 줄로만 표시하기

```
git config format.pretty oneline
```

파일을 추가할 때 대화식으로 추가하기

```
git add -i
```

링크 & 자료

그래픽 클라이언트

GitX (L) (OS X용, 오픈 소스 소프트웨어)
Tower (OS X용)
Source Tree (OS X용, 무료)
GitHub for Mac (OS X용, 무료)
Gitbox (OS X용, App Store)

한글 안내서

Pro Git
A Visual Git Guide
Git 작업 흐름
Git 브랜치 배우기

영문 안내서

Pro Git
Think like a git
GitHub Help
A Visual Git Guide

댓글

43 Comments git - the simple guide

Recommend 46 Tweet Share Sort by Newest

Join the discussion…

sukbu • a month ago
심플
Reply • Share ›

TeraClaNsky • 2 months ago
작성 된지 몇 년은 지난 게시글이지만 초보자가 깃을 이해하는데는 이것보다 쉬운 설명이 없습니다. 감사합니다.
Reply • Share ›

DongHyeok Lee • 2 months ago
깔끔해요!! 사랑합니다
Reply • Share ›

mingguk • 3 months ago
감사합니다!
Reply • Share ›

김수현 • 5 months ago
감사합니다. 정말 많은 도움이 되었어요ㅠㅠ
Reply • Share ›

Seo Donggu • 6 months ago
감사합니당 ㅠ_ㅠ
Reply • Share ›

mststst33 • 7 months ago
깔끔하고 알기 쉽게 잘 만들었네요~
감사합니다 :)
Reply • Share ›

김영준 • 7 months ago
잘 보고 갑니다. 감사합니다.
Reply • Share ›

이건일 • 10 months ago
아 어렵다
Reply • Share ›

박준호 • a year ago
다 좋은데 질문이 하나 있습니다.
git push origin master
하면 원본이 바뀌길 기대했는데, 그대로 이더군요. server에서 git checkout master 라고 하면 수정된 것들이 나옵니다.
그럼 다음에 다시 부르면, git pull, 처음 상태 것이 옵니까? master에 수정된 것이 올까요? 아니면 선택을 할 수 있는 것입니까?
Reply • Share ›

김선호 ➚ 박준호 • 9 months ago
뭔가 저도 잘 모르지만,,
git add 파일명
하시고
git commit -m "커밋에 대한 설명"
을 하시고
git push origin master 를 하셔야 반영이 되더라구여

git commit 을 안하면 안되는거 같습니당
Reply • Share ›

Kim • a year ago
ㅠㅜ 간단명료하고 시각적인 설명 감사합니다!!
Reply • Share ›

Soon-gang Heo • a year ago
고맙습니다.
덕분에 깃헙에 처음으로 파일들 올려봤네요 :)
Reply • Share ›

이정욱 • a year ago
좋은 정보 알려주셔서 감사해요~ Merge내용 조금만 추가해주세요 ㅠㅠ
아니면 example이라도 하나 달아주세요 ㅠㅠ
Reply • Share ›

dnter • a year ago
'Git 브랜치 배우기' 링크가 깨졌네요.
Reply • Share ›

글렌 • 2 years ago
감사합니다. ㅠㅠㅠ
진짜 설명 너무 잘 되어있어서 쉽게 이해했습니다.
Reply • Share ›

goodjob • 2 years ago
정말 감사합니다. 정말 많은 도움이 되었습니다.
Reply • Share ›

bigman • 2 years ago
뭔가 이쁘게 잘 만들었네요 ㅋㅋㅋ 굿입니다.
Reply • Share ›

Nam • 2 years ago
많은 도움이 되었습니다. 감사합니다.
Reply • Share ›

Eugene Chu • 2 years ago
감사합니다 그냥 줄글로 된것 보다 이해가 쉬워요
Reply • Share ›

강수지 • 2 years ago
유용한 정보 얻고 갑니다.
Reply • Share ›

houdini • 2 years ago
잘보고갑니다.
Reply • Share ›

Hyeonil Choi • 2 years ago
정보 잘 보고 갑니다. github~.
Reply • Share ›

Emily Lee • 2 years ago
감사합니다!
Reply • Share ›

Junwha Hong • 2 years ago
감사합니다. 갈피를 못잡고 있었는데 머리속에 확 들어오네요
Reply • Share ›

SeungKwon Park • 2 years ago
처음 깃을 접한 2015년에 이 글을 수도없이 읽은거 같아요
정리해주신 내용덕분에 지금은 git flow도 적용하고
git에 대한 두려움이 없습니다

정말 감사합니다
Reply • Share ›

david kim • 2 years ago
좋은 문서 감사합니다
Reply • Share ›

Rocky • 2 years ago
갱신과 병합(merge) 항목에서

충돌을 해결했다면, 아래 명령으로 git에게
아까의 파일을 병합하라고 알려주세요.
git add <파일 이름>

이라고 되어있는데

git commit <파일 이름>

이라고 해야 하지 않나요 ?
Reply • Share ›

하드코어GIT ➚ Rocky • 2 years ago
충돌(conflict)난 파일을 수정(충돌해결)해서 저장했다면 modified 상태인데 이것을 staged 상태로 만들라는 의미입니다. 물론 staged 한 다음에 최종적으로 이상이 없다면 바로 commit 을 해야죠.
Reply • Share ›

1 • 3 years ago
good!
Reply • Share ›

revie • 3 years ago
감사합니다 잘보고 갑니다
Reply • Share ›

Heechan Lee • 3 years ago
링크 & 자료에 OS X용만 보이는데 Windows와 Linux용도 적어주시면 좋겠습니다.
Reply • Share ›

PARK-JOO-SEOK • 3 years ago
정말 잘 정리되어 있네요.감사합니다~!
Reply • Share ›

benant • 3 years ago
잘 정리해주셔서 쉽게 이해가 됐습니다. 감사합니다.
Reply • Share ›

Rhee Jaeseok • 3 years ago
쉽게 설명을 잘 해주셔서 정말 감사합니다.^^.. 다음번에 실제 적용사례를 가지고 해주시면 정말 좋을것 같네요.
Reply • Share ›

Null • 3 years ago
잘보고 갑니다
Reply • Share ›

Jongho Kim • 3 years ago
스테이지에 올라가있는 리스트를 보는 bash 명령어와 스테이지에 올라와있는 리스트 중 몇가지만 커밋하려면 이에 해당하는 명령어는 뭘까요?
도움 부탁드립니다.
Reply • Share ›

Jeonghoon Seo (Scott) ➚ Jongho Kim • 3 years ago
git status 명령어로 현재 stage에 있는 파일들을 볼 수 있구요.
git add <파일명> 명령어로 원하는 파일들만 stage에 추가한 후
git commit 명령어로 커밋하면 되겠네요.

git commit 명령어를 까먹었어요...

^ | ∨ • Reply • Share ›

hanseong park • 3 years ago
정말 설명이 친절하고 쉽게 잘 되어 있는것 같습니다. 디자인도 좋아서 가독성도 매우 좋아요 감사합니다~
^ | ∨ • Reply • Share ›

LILIS • 3 years ago
많은 도움이 되었습니다. 멋진 설명 감사합니다. :)
^ | ∨ • Reply • Share ›

Juyong Woo • 3 years ago
많은 도움이 되었습니다. 감사합니다.(__)
^ | ∨ • Reply • Share ›

잉여 호박고구마 • 3 years ago
매우 잘 설명되있네요 ㅅㅅ
^ | ∨ • Reply • Share ›

개발자 • 3 years ago
잘보고 갑니다. 정말 간단하게 설명 잘 되어있네요
^ | ∨ • Reply • Share ›

Subscribe **Add Disqus to your siteAdd DisqusAdd**

Disqus' Privacy PolicyPrivacy PolicyPrivacy