

গিট

ডেভ সংকেত

গিট হচ্ছে ভার্সন কন্ট্রোল সিস্টেম। ডেভেলপাররা সাধারণত প্রজেক্টের ভার্সন কন্ট্রোল এবং একই প্রজেক্টে একের অধিক ডেভেলপার কাজ করার জন্য গিট ব্যবহার করে।

ইনস্টল (Setup)

গিট ইনস্টল (ম্যাক)	<code>brew install git</code>
গিট আনইনস্টল (ম্যাক)	<code>brew remove git</code>
গিট ইনস্টল (উবুন্টু)	<code>sudo apt-get install git</code>
গিট আনইনস্টল (উবুন্টু)	<code>sudo apt-get remove git</code>
গিট ভার্সন চেক	<code>git --version</code>
গিটটি কোথায় রয়েছে তা দেখুন	<code>which git</code>
গিট সাহায্য	<code>git help</code>

কনফিগ (Config)

গিট এর গ্লোবাল ইউজার নেম সেট করা
`git config --global user.name "username"`

গিট এর গ্লোবাল ইউজার নেম চেক করা
`git config --global user.name`

গিট এর গ্লোবাল ইউজার ইমেইল সেট করা
`git config --global user.email "user@name.com"`

গিট এর গ্লোবাল ইউজার ইমেইল চেক করা
`git config --global user.email`

গিট এর লোকাল/একটি নির্দিষ্ট রিপোজিটরির ইউজার নেম সেট করা
`git config user.name "username"`

গিট এর লোকাল/একটি নির্দিষ্ট রিপোজিটরির ইউজার নেম চেক করা
`git config user.name`

গিট এর লোকাল/একটি নির্দিষ্ট রিপোজিটরির ইউজার ইমেইল সেট করা
`git config user.email "user@name.com"`

কনফিগ (Config)

গিট এর লোকাল/একটি নির্দিষ্ট রিপোজিটরির ইউজার ইমেইল চেক করা
`git config user.email`

গিট এর গ্লোবাল ইউজার নেম বাতিল করা
`git config --global --unset-all user.name`

গিট এর গ্লোবাল ইউজার ইমেইল বাতিল করা
`git config --global --unset-all user.email`

গিট এর লোকাল/একটি নির্দিষ্ট রিপোজিটরির ইউজার নেম বাতিল করা
`git config --unset user.name`

গিট এর লোকাল/একটি নির্দিষ্ট রিপোজিটরির ইউজার ইমেইল বাতিল করা
`git config --unset user.email`

একটি নির্দিষ্ট ব্রাণের জন্য ইউজার নেম, পাসওয়ার্ড সেভ করা
`git config credential.helper store`

সকল ব্রাণের জন্য ইউজার নেম, পাসওয়ার্ড সেভ করা
`git config --global credential.helper store`

সাধারণ (General)	
নতুন রিপোজিটরি তৈরী করা	git init
কোনো রিপোজিটরি ক্লোন করা(লোকাল মেশিনে ডাউনলোড করা) git clone [url]	
রিপোজিটরি এর বর্তমান অবস্থা/পরিবর্তনগুলো দেখা (নতুন অথবা পুরোনো, কি কি ফাইল কমিট করতে হবে, ওয়ার্কিং ব্রাঞ্চ ইত্যাদি) git status	
কমিট এর জন্যে সব পরিবর্তিত ফাইল এড করা git add OR git add . OR git add --all OR git add -A	
কাস্টম ফাইল কমিট এর জন্যে এড করা git add index.html OR git add [file]	
একটি নির্দিষ্ট এক্সটেনশন এর সকল ফাইল এড করা git add *.ext	
একটি নির্দিষ্ট ফোল্ডারের সকল ফাইল এড করা git add /folder	

সাধারণ (General)	
একটি নির্দিষ্ট ফোল্ডারের একটি নির্দিষ্ট এক্সটেনশন এর সকল ফাইল এড করা git add folder/*.ext	
রিপোজিটরি তে কিছু কমিট করা (কমিট মেসেজসহ) git commit -m "YourCommitMessage"	
কমিট মেসেজ সংশোধন করা git commit --amend -m "your message"	
টাইটেল এবং ডেসক্রিপশন সহ কমিট করতে git commit -m "Title" -m "Description..."	
অ্যাড এবং কমিট এক স্টেপ এ করতে git commit -am "Message"	
নির্দিষ্ট কমিট দেখা	git show {commit-id}
সমস্ত পরিবর্তন আপডেট করতে	git add -u

সাধারণ (General)	
ফাইল রিমুভ করতে	git rm index.html
রিপোজিটরিতে এড করা ফাইল আনট্র্যাক করা git rm --cached index.html OR git rm --cached filename	
gitignore এ থাকা সকল ফাইল আনট্র্যাক করা git rm -r --cached .	
ফাইল মুভ অথবা রিনেইম করতে git mv index.html dir/index_new.html	
নির্দিষ্ট কমিট পরিবর্তন না করা (ফাইল সর্বশেষ কমিট ভার্সন থেকে রিস্টোর করতে) git checkout -- index.html	
সকল কমিট পরিবর্তন না করা	git checkout -- .
ফাইল কাস্টম কমিট ভার্সন থেকে রিস্টোর করতে (কারেন্ট ব্রাঞ্চ এ) git checkout 6eb715d -- index.html	

রিসেট (Reset)	
কমিট প্রত্যাবর্তন করা git revert {commit-id} OR git revert <commit hash>	
সর্বশেষ কমিট প্রত্যাবর্তন করা	git revert HEAD
নির্দিষ্ট কমিট প্রত্যাবর্তন করা git revert -n <commit hash>	
সফট রিসেট (move HEAD only; neither staging nor working dir is changed) git reset --soft {commit-id}	
শেষ কমিট থেকে stage এরিয়া তে মুভ করা git reset --soft HEAD~	
মিক্সেড রিসেট (move HEAD and change staging to match repo; does not affect working dir) git reset --mixed {commit-id}	
হার্ড রিসেট (move HEAD and change staging dir and working dir to match repo) git reset --hard {commit-id}	

রিসেট (Reset)

শেষ কমিট ডিলেট করা `git reset --hard HEAD^`

শেষ দুটি কমিট ডিলেট করা `git reset --hard HEAD^^`

সিঙ্গেল ফাইল হার্ড রিসেট করতে (@ is short for HEAD)
`git checkout @ -- index.html`

আপডেট এবং ডিলিট (Update & Delete)

আনট্র্যাক ফাইলগুলো টেস্ট ডিলিট করতে `git clean -n`

আনট্র্যাক ফাইলগুলো ডিলিট করতে `git clean -f`

আনস্টেজড (undo adds)
`git reset HEAD index.html`

ব্রাঞ্চ (Branch)

সব লোকাল ব্রাঞ্চ এর নাম লিস্ট করা `git branch`

নতুন ব্রাঞ্চ তৈরী `git branch [branch_name]`

ব্রাঞ্চ চেঞ্জ করতে `git checkout [branch_name]`

ব্রাঞ্চ তৈরী এবং চেঞ্জ করা
`git checkout -b "branch name"`

রিপোজিটরি/ব্রাঞ্চ রিনেম করা
`git branch -m branchname new_branchname` OR
`git branch --move branchname new_branchname`

কারেন্ট ব্রাঞ্চ এর সাথে মার্জ করা সকল ব্রাঞ্চগুলো শো করতে
`git branch --merged`

ব্রাঞ্চ রিমুভ করা (only possible if not HEAD)
`git branch -d branchname` OR `git branch --delete branchname`

ব্রাঞ্চ (Branch)

রিমোট ব্রাঞ্চ রিমুভ করা
`git push --delete origin [branch-name]`

মার্জ করা না এমন ব্রাঞ্চ রিমুভ করতে
`git branch -D branch_to_delete`

মার্জ (Merge)

ট্রু মার্জ (fast forward) `git merge branchname`

মাস্টারে মার্জ করতে (only if fast forward)
`git merge --ff-only branchname`

মাস্টারে মার্জ করতে (force a new commit)
`git merge --no-ff branchname`

মার্জ বন্ধ করতে (in case of conflicts)
`git merge --abort`

মার্জ বন্ধ করতে (in case of conflicts)
`git reset --merge // prior to v1.7.4`

লোকাল মার্জ আন্ডু করতে যেই মার্জটি এখনো পুশ করা হইনাই
`git reset --hard origin/main`

স্পেসিফিক একটি কমিট মার্জ করতে
`git cherry-pick {commit-id}`

মার্জ (Merge)

রিবেস
`git checkout branchname » git rebase main`
OR `git merge main branchname`

রিবেস ক্যানসেল করতে `git rebase --abort`

মাল্টিপল কমিট একটি কমিট এ স্কোয়াশ করতে
`git rebase -i HEAD~3 (source)`

ফীচার ব্রাঞ্চ Squash-merge করতে (একটি কমিট এ)
`git merge --squash branchname (commit afterwards)`

স্টাশ (Stash)	
স্টাশ এ রাখতে	git stash save "Message"
স্টাশ লিস্ট শো করতে	git stash list
স্টাশ স্ট্যাটাস শো করতে	git stash show stash@{0}
স্টাশ এর পরিবর্তনগুলি শো করতে	git stash show -p stash@{0}
স্টাশ এর সবগুলি আইটেম একসাথে ব্যবহার এবং ড্রপ করতে	git stash pop
স্টাশ লিস্ট এর স্পেসিফিক আইটেম ব্যবহার এবং ড্রপ করতে	git stash pop stash@{0}
স্টাশ লিস্ট এর স্পেসিফিক আইটেম শুধু ব্যবহার এবং ড্রপ না করতে	git stash apply stash@{0}

স্টাশ (Stash)	
কাস্টম স্টাশ আইটেম ব্যবহার এবং ইনডেক্স করতে	git stash apply --index
স্টাশ থেকে নতুন ব্রাঞ্চ করতে	git stash branch new_branch
স্টাশ লিস্ট এর প্রথম আইটেম বাদ দিতে	git stash drop
স্টাশ লিস্ট এর স্পেসিফিক আইটেম বাদ দিতে	git stash drop stash@{0}
সম্পূর্ণ স্টাশ ডিলিট করতে	git stash clear

লগ (Log)	
কমিট লগ বিস্তারিত দেখতে	git log
এক লাইন এ কমিট লগ এর সামারি শো করতে	git log --oneline
এক লাইন এ কমিট লগ এর সামারি ফুল SHA-1 ফরম্যাট এ শো করতে	git log --format=oneline
কমিট লগ সামারি আকারে দেখতে((৫টি)	git log --oneline -5
শুধুমাত্র কাস্টম কমিটগুলো শো করতে	git log --author="Sven" git log --grep="Message" git log --until=2013-01-01 git log --since=2013-01-01
শুধুমাত্র কাস্টম কমিট এর ডাটা শো করতে	git log --format=short git log --format=full git log --format=fuller git log --format=email git log --format=raw
পরিবর্তনগুলি শো করতে	git log -p

লগ (Log)	
শুধুমাত্র কাস্টম ফাইল এর সবগুলো কমিট শো করতে	git log 6eb715d.. index.html
শুধুমাত্র কাস্টম ফাইল এর সবগুলো কমিট এর পরিবর্তনগুলি শো করতে	git log -p 6eb715d.. index.html
কমিটগুলোর স্ট্যাটাস এবং সামারি শো করতে	git log --stat "," git log --stat --summary
কমিটগুলোর হিস্ট্রি গ্রাফ আকারে শো করতে	git log --graph
কমিটগুলোর হিস্ট্রি গ্রাফ আকারে সামারি শো করতে	git log --oneline --graph --all --decorate

তুলনা (Compare)	
শেষ কমিট এবং বর্তমান unstaged ভার্সনের পার্থক্য দেখা	git diff
মডিফাইড ফাইলস তুলনা এবং পরিবর্তনগুলি হাইলাইট করতে	git diff --color-words index.html
শেষ কমিট এবং বর্তমান staged ভার্সনের পার্থক্য দেখা	git diff --staged
সর্বশেষ কমিটের আগের কমিট ও সর্বশেষ কমিটের মাঝে তুলনা করতে	git diff HEAD^ HEAD
ব্রাঞ্চগুলো তুলনা করতে	git diff main..branchname
উপরের মত ব্রাঞ্চগুলো তুলনা করতে	git diff --color-words main..branchname^
কমিটগুলো তুলনা করতে	git diff {commit-id} git diff {commit-id}..HEAD git diff {commit-id}..{commit-id}

তুলনা (Compare)	
ফাইল এর কমিটগুলো তুলনা করতে	git diff {commit-id} index.html git diff {commit-id}..{commit-id} index.html
দরকারী তুলনা	git diff --stat --summary {commit-id}..HEAD
ব্ল্যাম	git blame -L10,+1 index.html

রিলিজ & ভার্সন (Releases & Version)	
ট্যাগ লিস্ট দেখতে	git tag
ট্যাগ তৈরি করা	git tag [tag-name]
প্রকাশিত সবগুলো ভার্সন কমিটগুলো সহ শো করতে	git tag -l -n1
রিলিজ ভার্সন তৈরী করতে	git tag v1.0.0
কমেন্টসহ রিলিজ ভার্সন তৈরী করতে	git tag -a v1.0.0 -m 'Message'
নির্দিষ্ট রিলিজ ভার্সনে চেকআউট করতে	git checkout v1.0.0

সহযোগিতা (Collaborate)	
রিমোট শো করতে	git remote
সবগুলো রিমোটের লিস্ট নাম এবং URL সহ দেখা	git remote -v
নতুন রিমোট এড করা	git remote add <RemoteName> <RemoteURL>
রিমোট পরিবর্তন করা	git remote set-url <RemoteName> <RemoteURL>
Fork রিপোজিটরি ক্ষেত্রে আপস্ট্রিম রিমোট এড করা	git remote add upstream <RemoteURL>
Fork রিপোজিটরি ক্ষেত্রে আপস্ট্রিম রিমোট পরিবর্তন করা	git remote set-url upstream <RemoteURL>
সার্ভারে আপস্ট্রিম রিমোট কনফিগার করা	git remote add upstream ssh://root@123.123.123.123/path/to/repository/

সহযোগিতা (Collaborate)

অন্য রিপোজিটরি থেকে ব্রাঞ্চ/রেফ/অবজেক্ট fetch করা
`git fetch RepositoryName`

Fetch/Fork রিপোজিটরি কে আপস্ট্রিম(upstream) এর সাথে up-to-date রাখা
`git fetch upstream`

fetch'd কমিটগুলো মার্জ করতে
`git merge upstream/main` OR `git merge upstream/branchname`

কাস্টম ব্রাঞ্চ Fetch করতে
`git fetch upstream`
`branchname:local_branchname`

অরিজিন রিমুভ করতে `git remote rm origin`

সব রিমোট ব্রাঞ্চ এর নাম লিস্ট করা `git branch -r`

সব লোকাল এবং রিমোট ব্রাঞ্চ এর নাম লিস্ট করা
`git branch -a`

সহযোগিতা (Collaborate)

রিমোট ব্রাঞ্চ থেকে ব্রাঞ্চ তৈরি এবং চেকআউট করতে
`git checkout -b local_branchname`
`upstream/remote_branchname`

রিপোজিটরি তে লোকাল ব্রাঞ্চ থেকে আপলোড করা
`git push -u origin main` OR `git push -u origin [branchName]`

পুশ `git push origin main`

বর্তমান/ Checkout/ Current ব্রাঞ্চ এ পুশ করতে
`git push origin HEAD`

ফোর্স পুশ `git push origin main --force`

রিপোজিটরি থেকে নতুন চেক্স গুলো পুল করা `git pull`

স্পেসিফিক ব্রাঞ্চ পুল করতে
`git pull origin branchname`

সহযোগিতা (Collaborate)

কোনো রিপোজিটরি ক্লোন করা(লোকাল মেশিনে ডাউনলোড করা)

`git clone`
`https://github.com/user/project.git` OR `git clone ssh://user@domain.com/~dir/.git`

লোকাল ফোল্ডার এ ক্লোন করতে

`git clone`
`https://github.com/user/project.git`
`~/dir/folder`

স্পেসিফিক ব্রাঞ্চ লোকাল এ ক্লোন করতে

`git clone -b branchname`
`https://github.com/user/project.git`

অথেনটিকেশন টোকেন ব্যবহার করে ক্লোন করতে

`git clone https://oauth2:`
`<token>@gitlab.com/username/repo.git`

রিমোট ব্রাঞ্চ ডিলিট করতে

`git push origin :branchname` OR `git push origin --delete branchname`

কন্ট্রিবিউটরদের নামের লিস্ট দেখা `git shortlog -sn`

কন্ট্রিবিউটরদের নাম এবং তাঁদের সকল কমিট লিস্ট আকারে দেখা

`git shortlog`

গিটহাব ইস্যু (Github issues)
resolves
resolved

এই চিটশিটে কন্ট্রিবিউট করেছেনঃ



Iqbal Hossain Rony

/iqbalrony



Sayedul Sayem

/sayedulsayem



Shaon Kabir

/shaonkabir8



Md. Tahmid Hossain

/tahmid02016



sabbirshawon

/sabbirshawon



Fahim Faisaal

/fahimfaisaal



ASIF ADIB

/asifadib



Ali Siddique

/imalisiddique



Imran Fakhrul

/imranfakhrul



MD. Mehedi Hasan

/Mehedi61



Tanbir chowdhury

/totanbir



mirajehossain

/mirajehossain



Niyaz Bin Hashem

/niyazed



Abu Sakib

/selectiveduplicate



Ariful

/arif98741



Zonayed Ahmed

/zonayedpcadotcom



Afrida Aesha

/afrida67



সংকেত

বাংলা চিটশিটের ভান্ডার
devsonket.com