گیت (Git)

معرفی گیت (Git)

«گیت» (Git)، یک سیستم کنترل نسخه متن‌باز «توزیع شده» (Distributed) و مسئول همه موارد مرتبط با گیت‌هاب است که روی کامپیوتر کاربر رخ می‌دهد. در این تقلب‌نامه، مهم‌ترین و متداول‌ترین دستورات Git ارائه شده‌اند.

نصب (Installation) و رابط گرافیکی کاربر (GUI)

گیت دارای نرم‌افزارهای نصبی است که برای پلتفرم‌های گوناگون طراحی شده‌اند. گیت‌هاب با ارائه جدیدترین نسخه‌های ابزار خط فرمان و همچنین رابط گرافیکی کاربری برای گیت که به منظور تعامل‌های روزانه، بررسی و همگام‌سازی مخازن استفاده می‌شود، باعث سهولت زیادی در به روز نگه داشتن کد شده است.

گیت‌هاب برای ویندوز: https://windows.github.com

گیت‌هاب برای مَک: https://mac.github.com

برای پلتفرم‌های «لینوکس» (Linux) و «سولاریس» (Solaris)، آخرین نسخه گیت در وب‌سایت رسمی آن موجود است.

گیت برای همه پلتفرم‌ها: http://git-scm.com

راه‌اندازی (Setup)

دستورهای مورد نیاز برای پیکربندی اطلاعات کاربر که در مخازن محلی مورد استفاده قرار می‌گیرند:

تعیین نام مالک مخزن که هنگام بررسی تاریخچه نسخه قابل شناسایی است.	git config --global user.name “[firstname lastname]”
تعیین آدرس ایمیلی که به هر «نشانگر» (Marker) تاریخچه تعلق می‌یابد.	git config --global user.email “[valid-email]”
تعیین رنگ‌بندی خودکار خط فرمان برای گیت که به منظور تسهیل بررسی صورت می‌گیرد.	git config --global color.ui auto

راه‌اندازی (Setup) و مقداردهی اولیه

دستورهای مورد نیاز برای پیکربندی اطلاعات کاربر و مقداردهی اولیه و کلون کردن:

مقداردهی اولیه یک دایرکتوری موجود به عنوان مخزن گیت	git init
بازیابی کل یک مخزن از یک موقعیت میزبانی شده با استفاده از URL	git clone [url]

SNAPSHOT و STAGE	
دستورهای مورد نیاز برای کار کردن با Snapshotها و ناحیه Staging در گیت:	
فایل‌های ویرایش شده در دایرکتوری که برای کامیت بعدی Stage شده‌اند نمایش می‌یابند.	git status
یک فایل را به همان شکلی که در حال حاضر قرار دارد به کامیت بعدی (Stage) اضافه می‌کند.	git add [file]
یک فایل را در حالی که تغییرات در دایرکتوری کاری بدون تغییر باقی می‌ماند، از Stageخارج می‌کند.	git reset [file]
مواردی که تغییر کرده‌اند ولی هنوز Stage نشده‌اند را مشخص می‌کند.	git diff
مواردی که Stage شده ولی هنوز کامیت نشده‌اند را مشخص می‌کند.	git diff --staged
محتوای Stage شده را به عنوان یک Snapshot از کامیت جدید، کامیت می‌کند.	git commit -m “[descriptive message]”
انشعاب (Branch) و ادغام (Merge)	
به وسیله این دو دستور می‌توان فرایند کاری را در شاخه‌های مختلف انشعاب داد، چارچوب کار را عوض کرد و تغییرات را یکپارچه‌سازی کرد:	
شاخه‌های مختلف مخزن را فهرست می‌کند. پس از شاخه فعال کنونی، یک علامت «*» قابل مشاهده است.	git branch
یک شاخه جدید در کامیت کنونی ایجاد می‌کند.	git branch [branch-name]
با این دستور می‌توان به یک شاخه دیگر رفت و آن را با دایرکتوری کاری مقایسه کرد.	git checkout
تاریخچه شاخه تعیین شده با تاریخچه انشعاب کنونی ادغام می‌شود.	git merge [branch]
همه کامیت‌ها در تاریخچه شاخه کنونی نمایش داده می‌شوند.	git log

بررسی (Inspect) و مقایسه (Compare)	
با استفاده از این دو دستور می‌توان به بررسی لاگ‌ها (logs)، diffها و اطلاعات شی پرداخت:	
این دستور لاگ‌های کامیت را در شاخه فعال کنونی نمایش می‌دهد.	git log
کامیت‌های روی شاخه A را که روی شاخه B نیستند نمایش می‌دهد.	git log branchB..branchA
کامیت‌هایی که فایل را تغییر داده‌اند را حتی با وجود تغییر یافتن نام نمایش می‌دهد.	git log --follow [file]
مواردی که در شاخه A قرار دارند اما در شاخه B نیستند را نمایش می‌دهد.	git diff branchB...branchA
همه موارد موجود در گیت را به صورتی که برای خواندن انسان مناسب است نمایش می‌دهد.	git show [SHA]
ردگیری تغییرات مسیر (Tracking Path Changes)	
با استفاده از این دستورها می‌توان به نسخه‌بندی حذف شدن فایل‌ها و تغییرات رخ داده در مسیر فایل‌ها پرداخت:	
یک فایل را از پروژه حذف کرده و آنچه حذف شده را برای کامیت شدن به ناحیه stage می‌برد.	git rm [file]
فایل را از پروژه حذف کرده و این جابه‌جایی را Stage می‌کند.	git mv [existing-path] [new-path]
همه لاگ‌های کامیت‌ها را با تاکید روی مسیرهایی که جابه‌جا شده‌اند نمایش می‌دهد.	git log --stat -M
الگوهای نادیده‌گیری (Ignoring Patterns)	
یک فایل با الگوی مورد نظر که بر اساس تطبیق رشته‌ای مستقیم یا متغیرهای سراسری وایلدکارد تعریف می‌شود را به صورت gitignore ذخیره می‌کند.	logs/ .*notes pattern/*
یک الگوی نادیده‌گیری را روی همه سیستم برای تمام مخازن محلی تعریف می‌کند.	git config --global core.excludesfile [file]

اشتراک‌گذاری و به روز رسانی	
به وسیله این دستورها، به روز رسانی‌های صورت گرفته در مخازن دیگر بازیابی شده و مخزن محلی بر همین اساس به روز رسانی می‌شود:	
git remote add [alias] [url]	یک URL گیت به عنوان یک نام «مستعار» (alias) اضافه می‌شود.
git fetch [alias]	همه شاخه‌ها از Git ریموت «واکشی» (Fetch) می‌شوند.
git merge [alias]/[branch]	یک شاخه ریموت جهت به روز رسانی شاخه جاری، در آن ادغام می‌شود.
git push [alias] [branch]	کامیت‌های شاخه محلی به شاخه مخزن ریموت منتقل می‌شوند.
git pull	همه کامیت‌های شاخه ریموت مورد ردگیری قرار می‌گیرند و در شاخه محلی واکشی و ادغام می‌شوند.
بازنویسی تاریخچه (Rewrite History)	
به وسیله این دستورها، شاخه‌ها بازنویسی، کامیت‌ها به روزرسانی و تاریخچه شاخه‌ها پاک می‌شود.	
git rebase [branch]	همه کامیت‌های شاخه جاری که قبل از کامیت تعیین شده قرار دارند، اعمال می‌شوند.
git reset --hard [commit]	ناحیه staging پاکسازی، و درخت کاری از کامیت تعیین شده بازنویسی می‌شود.
کامیت‌های موقت (Temporary Commits)	
با این مجموعه دستورها، فایل‌های ویرایش و ردگیری شده به منظور ایجاد تغییر در شاخه‌ها به صورت موقت ذخیره می‌شوند.	
git stash	تغییرات ویرایش و stage شده ذخیره می‌شوند.
git stash list	تغیرات فایل Stash شده با ترتیب پشته‌ای فهرست می‌شوند.
git stash pop	نوشتن از بالای پشته Stash صورت می‌گیرد.
git stash drop	تغییرات صورت گرفته در بالای پشته Stash لغو می‌شوند.

همگام‌سازی تغییرات (Synchronize Changes)	
با استفاده از این دستورها می‌توان «بوکمارک» (Bookmark) یک مخزن را ثبت و تاریخچه نسخه را مبادله کرد:	
همه تاریخچه را از بوکمارک مخزن دانلود می‌کند.	$git fetch [bookmark]
شاخه بوکمارک را با شاخه محلی کنونی ادغام می‌کند.	$git merge [bookmark]/[branch]
همه کامیت‌های شاخه محلی را در گیت‌هاب آپلود می‌کند.	$git push [alias] [branch]
تاریخچه بوکمارک را دانلود می‌کند و تغییرات را در بر می‌گیرد.	$git pull

مجموعه آموزش‌های برنامه‌نویسی فرادرس (+کلیک کنید)

برای مشاهده دیگر «تقلب‌نامه‌های» مجله فرادرس، به این لینک مراجعه فرمایید.

جهت آگاهی از آخرین تقلب‌نامه‌های منتشر شده، در کانال تلگرام مجله فرادرس عضو شوید.

تهیه و تنظیم: مجله فرادرس

منبع

برای مشاهده دیگر «تقلب‌نامه‌های» مجله فرادرس، به این لینک مراجعه فرمایید.