Engineer

Git

versionshanteringen där den här webbplatsens historik finns

Vad det är

Git är ett distribuerat versionshanteringssystem: ett program som registrerar tillståndet hos en samling filer som en kedja av ögonblicksbilder, så att varje tidigare tillstånd kan granskas, jämföras med nuläget eller tas tillbaka. Projektet skriver namnet som Git och kommandot som `git`, och båda avser samma verktyg.

Varje ögonblicksbild är en commit. En commit lagrar hur varje spårad fil såg ut i just det ögonblicket, vem som gjorde ändringen och när, ett meddelande som säger varför, och en pekare till den eller de commits som kom före. En fil som git redan har sett lagras bara en gång: är filen oförändrad mellan två commits pekar den nya ögonblicksbilden på kopian som redan finns på disken i stället för att skriva den på nytt, så en lång historik tar betydligt mindre plats än en fullständig kopia per commit skulle göra.

Allt där inne namnges efter sitt innehåll. Innan git lagrar en fil, en kataloglistning eller en commit beräknar det en kontrollsumma över innehållets byte, i det klassiska formatet en SHA-1-hash skriven som 40 hexadecimala tecken, och kontrollsumman blir objektets namn. Ändras en enda byte ändras namnet med den, liksom namnet på varje commit som bygger ovanpå. Det är så git upptäcker korruption, och det är därför en commits identifierare fortfarande pekar på exakt samma kod flera år senare.

Distribuerat betyder att varje kopia är komplett. Att klona ett repo hämtar hela historiken och inte bara de senaste filerna, så att läsa loggen, jämföra två versioner eller göra en commit sker på den lokala disken utan att någon server är inblandad. Att dela arbete är ett separat och medvetet steg: du pushar dina commits till en annan kopia, eller hämtar commits från en och slår ihop dem med

dina egna. Vilken kopia som helst kan fungera som server, och i de flesta team spelar en överenskommen kopia, ofta hos en värdtjänst, den rollen av konvention snarare än av konstruktion.

Det dagliga arbetet rör sig genom tre platser. Arbetskatalogen håller filerna du redigerar. Förberedelseområdet, som också kallas indexet, håller det som nästa commit ska innehålla, vilket låter dig committa en del av dina ändringar och spara resten till senare. Git-katalogen, `.git` högst upp i projektet, håller objektdatabasen och varje referens. En gren är inget annat än ett flyttbart namn som pekar på en enda commit, och därför går det på ett ögonblick att skapa en.

Git skrevs 2005 av Linus Torvalds och gemenskapen kring Linux-kärnan, sedan den kostnadsfria licensen för BitKeeper, det proprietära verktyg de dittills hade använt, hade dragits in. Målen de satte upp var snabbhet, en enkel design, starkt stöd för tusentals parallella grenar, full distribution och förmågan att hantera ett projekt lika stort som kärnan.

Använd begreppet i ett beslut

Använd ”Vad är git?” i ett konkret beslut. Vi kan knyta begreppet till systemet ni bygger eller köper. Testade, släppta och möjliga att backa, så att ändringar når kunderna en tisdagseftermiddag i stället för sent en fredagskväll.

Se våra tjänster Hör av er →

Granska antagandena bakom ordet

Använd förklaringen och det länkade arbetet för att formulera frågor till projektet. Vi kan undersöka antagandena tillsammans innan ett verktyg väljs. Automatiserade GIS-driftsättning i GitHub Actions — driftsättning, data och rapportering, helt utan manuellt arbete

Automatiserade GIS-driftsättning i GitHub Actions Hör av er →

Varför det spelar roll

Versionshantering besvarar de frågor som dyker upp så fort mer än en person, eller en person under mer än en vecka, rör samma filer: vad som ändrades, när, av vem och varför. Förs svaren för hand lever de i minnet, i filnamn som `final-v3` och i e-posttrådar. Med git lever de bredvid koden, fästa vid varje ändring, där vem som helst kan söka i dem.

Den distribuerade designen ändrar hur ett team arbetar från dag till dag. Varje klon håller hela historiken, så ingen behöver vänta på en central server för att läsa loggen eller committa, och ett avbrott på servern låter alla fortsätta arbeta. Grenar är billiga, så varje funktion, rättning eller experiment kan få en egen, granskas för sig och slås ihop när den är klar, medan huvudlinjen hela tiden går att leverera.

Samma design gör git försiktigt med data. Nästan varje operation lägger bara till i objektdatabasen, så arbete som har committats är svårt att förlora: en gren som raderats av misstag, en rebase som gick snett eller en reset till fel commit går oftast att ångra. Skyddsnätet är reflog, en lokal förteckning över vart varje gren pekade och när, som enligt standardinställningen sparas i 90 dagar för commits som fortfarande går att nå och i 30 dagar för dem som inte gör det.

Git är också den gemensamma grunden under de flesta andra verktyg i en leveranskedja. En forge, ett granskningsverktyg, en byggpipeline, ett skript för driftsättning och en statisk webbplatsgenerator läser alla från det, och det gör historiken till en källa till fakta långt bortom koden: dagen då en sida senast ändrades, den commit som en release byggdes från, den ändring som en bugg kom in med.

Så ser det ut i praktiken

Företagets egen kod visar tre sidor av det.

Källkoden låg tidigare hos en värdtjänst som drevs av tredje part. Nu finns den på en git-server som företaget kör på sin egen hårdvara: en enda binärfil som hålls igång av operativsystemet, vald för att den är byggd i första hand för SSH och saknar administrativt webbgränssnitt. Den som klonar eller pushar behöver en nyckel, eftersom både anonym åtkomst och åtkomst utan nyckel nekas, och varje deklarerat repo är privat. Trafiken går via en hoppserver till port 23231, där servern bara lyssnar på loopback. Sidan säger rakt ut vad det kostar: att driva sin egen forge betyder att också äga autentisering, åtkomstkontroll, lagring och säkerhetskopior.

Fullständiga kopior av historiken på flera maskiner är lätta att ta för en säkerhetskopia. Företagets implementering av säkerhetskopiering och återställningstest utformades för att utöka återställningsberedskapen bortom en ögonblicksbild hos leverantören. Dess restic-skript och systemd-enheter stöder en databasdump där det är konfigurerat, krypterade, deduplicerade ögonblicksbilder via SFTP till lagring hos en annan leverantör, gallring enligt bevarandepolicyn och integritetskontroller. Ett separat månatligt arbetsflöde återställer en känd fil till ett tillfälligt arbetsutrymme och jämför dess byte. Det testet måste faktiskt köras innan det ger belägg för en återställningsberedskap: implementeringsbeskrivningen daterad 2026-10-27 anger att timerenheterna var

inaktiverade utan ett konfigurerat repo och redovisar ingen genomförd återställning i drift. Hur många kloner det än finns på bärbara datorer ger de inte de belägg som saknas.

Historiken matar också själva webbplatsen. Webbplatsens kolofon noterar att W3C:s Feed Validator varnar för att de flesta posterna i flödena har samma ändringsdatum, och att varningen stämmer: datumen kommer från git-historiken, och de flesta av sidorna skrevs verkligen i ett fåtal commits. Webbplatsen visar hellre varningen än hittar på ett datum. Historiken säger vad som hände, även att mycket av det hände på en gång.

Ett enda kommando läser tillbaka ett sådant faktum. Det här skriver ut den korta identifieraren, commitdatumet i formatet ISO 8601 och ämnesraden för den senaste commit som rörde en viss fil:

```
git log -1 --format='%h %cI %s' -- README.md
```

Var det går fel

- **En operativ hemlighet som har committats en gång stannar i historiken.** Att radera filen i nästa commit tar bort den ur den senaste ögonblicksbilden men lämnar den kvar i den föregående, som varje klon redan bär på. Behandla åtkomstuppgiften som röjd och byt ut den. Att skriva om historiken i efterhand begränsar spridningen, men kopior som gjordes under tiden behåller de gamla commits de redan har.
- **En tvingad push ersätter någon annans arbete.** `git push --force` stänger av kontrollen av att fjärrgrenen är en förfader till det du pushar, så commits som en kollega pushade under tiden faller bort från grenen. `--force-with-lease` uppdaterar fjärrgrenen bara så länge den fortfarande pekar dit du senast såg den.
- **Kloner behandlas som säkerhetskopian.** En kopia på en laptop ligger efter, håller bara de grenar som ägaren råkade hämta och har aldrig medvetet använts för en återställning. Ärenden, åtkomstregler och resten av de data som forgen själv håller ligger helt utanför repot.
- **En stor binärfil gör varje klon tyngre för gott.** Git behåller varje version av varje fil och varje klon laddar ned alla, så en video som committades av misstag och raderades dagen efter följer ändå med till varje maskin som klonar projektet efteråt.

- **Arbete som bara finns på ett ställe försvinner med det stället.** Den reflog som räddar en misslyckad reset finns på en enda maskin och löper ut. Commits som aldrig har pushats finns ingen annanstans, och en gren som lämnas orörd i månader glider så långt isär att det blir ett eget projekt att slå ihop den igen.

Läs mer

- Git documentation — referensmanualen, boken och videorna, direkt från projektet.
- Pro Git — hela boken på nätet, från en första commit till objektdatabasen därunder.
- What is Git? — ögonblicksbilder, lokala operationer, kontrollsummor och de tre tillstånden i ett och samma kapitel.
- A Short History of Git — varför gemenskapen kring Linux-kärnan skrev ett eget verktyg 2005 och vad det skulle klara.
- git-reflog — hur man hittar vart en gren pekade före misstaget, och hur länge den förteckningen finns kvar.
- git-push — alternativen för tvingad push och vad vart och ett av dem skyddar.



Ordlista

Välj utifrån era faktiska förutsättningar

Ta med driftsättningsvägen, driftsriskerna och dem som har jour. Vi kan avgränsa en förändring med kontroller, en väg tillbaka och tydligt ansvar.

Se vad det kostar Hör av er →

Gör begreppen praktiska

Fråga hur ”Vad är git?” gäller i er organisation. Ett konkret exempel från ert nuvarande system räcker för att börja samtalet.

Se våra tjänster

welcome@engineer.company +45 50 10 61 84 Köpenhamn, Danmark

Ta med dig den här sidan

Visitkort

Omsorgsfullt skapad med kärlek av D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Upphovsrätt © 2025 – nu · Engineer ApS · Alla rättigheter förbehållna.

Utskrivet från https://engineer.company/sv/glossary/git/