Engineer

Kodbas

all källkod för en app, samlad i ett enda repo

Vad det är

En kodbas, på engelska codebase, är hela den samling källkod som ett team redigerar, granskar och levererar som en enhet. När utvecklare talar om ”kodbasen” menar de stället där varje ändring av en applikation landar: själva koden, dess tester, skripten som bygger och kontrollerar den, ändringarna i databasen och dokumenten som förklarar hur delarna hänger ihop.

The Twelve-Factor App, en metod för att bygga webbtjänster, ger den mest användbara arbetsdefinitionen, eftersom den gör gränsen till något som går att testa. I ett centraliserat versionshanteringssystem som Subversion är en kodbas ett repo. I ett distribuerat system som git är det varje uppsättning repon som delar en rotcommit, alltså den första committen i historiken, den som saknar förälder. Två repon som har vuxit fram ur samma första commit är en och samma kodbas, hur långt isär de än har glidit. Två som började var för sig är två kodbaser, även om de innehåller nästan identiska filer.

Samma dokument parar ihop den definitionen med en andra regel: en kodbas, många driftsättningar. En driftsättning är här en körande instans av applikationen, och det finns oftast flera samtidigt. Produktion är en, staging en annan, och varje utvecklares lokala kopia räknas också. Alla kör samma kodbas, ofta vid olika commits: en utvecklare har arbete som staging inte har sett, och staging har arbete som produktion inte har sett.

Vad som hör hemma i kodbasen följer av det. Källkoden och allt som behövs för att göra den till ett fungerande program hör dit, inklusive en manifestfil som namnger varje beroende och dess version. Själva beroendena ligger oftast utanför och hämtas av en pakethanterare. Värden som skiljer sig från en

driftsättning till nästa ligger också utanför, och framför allt inloggningsuppgifter: ett lösenord som har committats en enda gång följer med varje kopia av historiken från den stunden.

En kodbas kan rymma flera program. Ett monorepo håller en frontend, ett API och databaskoden sida vid sida i ett och samma repo, så att en enda commit kan ändra alla tre. Enligt regeln om rotcommiten är det fortfarande en kodbas. Det motsatta valet sprider en applikation över repon med separata historiker, och då är varje repo en egen kodbas. Twelve-Factor är rakt på sak om följden: finns det flera kodbaser är det ett distribuerat system, och varje komponent är en app i sin egen rätt.

Använd begreppet i ett beslut

Använd ”Vad är en kodbas?” i ett konkret beslut. Vi kan knyta begreppet till systemet ni bygger eller köper. Systemets form utarbetad på papper först, så att delarna passar ihop och kostnaden är känd innan någon skriver kod.

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. En infrastrukturplan för 14 avdelningar — 14 forskningsavdelningar under en och samma infrastrukturplan

En infrastrukturplan för 14 avdelningar Hör av er →

Varför det spelar roll

Gränsen avgör vad som kan ändras tillsammans. Inom en kodbas kan en ändring av ett API och skärmen som anropar det skrivas, granskas och slås ihop som en enhet. Över två kodbaser blir samma ändring två pull requests, två releaser och en period då den ena sidan ligger före den andra. Båda sätten fungerar; priset bestäms av var linjen drogs.

Kodbasen är också den enhet som standarderna körs över. En testsvit, en linter, en formaterare, en regel för commit-meddelanden och en task runner tar alla ”kodbasen” som sitt område, och det som ligger utanför kontrolleras inte av någon av dem.

Tidiga beslut stelnar snabbare i en kodbas än någon annanstans. Varje rad som skrivs efter ett val förutsätter valet, så en regel om lagerindelning eller en kvalitetsribba som sätts under de första veckorna blir formen för allt som följer, och att vända på den senare innebär att skriva om koden som har vuxit upp runt den.

För människor är kodbasen det som en nykomling får i handen. Om den rymmer allt som behövs för att bygga, testa och köra applikationen räcker en klon och ett kommando för att komma igång. Om kunskapen ligger dels i koden och dels i någons minne går den första veckan åt till att ta reda på vad som är vad.

Så ser det ut i praktiken

Formen på en kodbas sitter i tre saker: ingången som alla använder, grundreglerna från de första veckorna och väggarna mellan lagren.

På en plattform är kodbasen ett polyglott monorepo i Go, TypeScript, SQL, Python och skalskript, och varje språk kommer med sitt eget sätt att bygga, testa och linta. Svaret blev en enda ingång: ett go-task-lager som växte till 578 namngivna mål, 50 i rotfilen och 528 i filer med egna namnrymder under den, och som täcker utvecklingslägen för nativt, Docker och HTTPS, kodkvalitet, databashantering och driftsättning. Vem som helst kan köra `task --list` och se hela verktygskedjan. Sidan läser också antalet som en varning: 578 mål är mer än någon kan hålla i huvudet, så det är namngivningen och namnrymderna som håller det hela användbart.

Ett annat projekt började med ett tomt repo den 25 november 2025, och tre beslut som fattades under de första veckorna höll fortfarande nio månader och flera tusen commits senare: en stack i lager, dataåtkomst bakom lagrade funktioner och en ribba med noll varningar som infördes innan det fanns mycket kod att hålla till den. Resonemanget på sidan är kärnan i den här artikeln: en standard som införs vid commit 5 000 är ett städprojekt, medan samma standard vid commit 50 helt enkelt är hur repot fungerar.

Väggarna inuti en kodbas kan vara själva designen. I grundarens arkitekturbidrag till NextMariner, en maritim plattform byggd i tre lager, sköter frontenden i Next.js presentationen och ingenting annat, ett medvetet tunt Go-API byggt med Huma ovanpå Fiber routar, validerar och orkestrerar, och affärslogiken ligger i PostgreSQL-funktioner. När en regel ändras sker det i ett lager, i SQL. Gränserna skrevs ned och upprätthölls i granskningen, eftersom en konvention som ingen bevakar slutar vara en konvention.

Den här webbplatsens egen kodbas drar en annan sorts gräns. Projektsidorna skrivs i ett separat repo, exporteras hit av ett skript och committas, så att webbplatsen kan byggas på egen hand. Inom den här kodbasen är de sidorna

genererade filer, och instruktionerna för arbetet här säger att de aldrig redigeras för hand.

Var det går fel

Kopierad kod glider isär

Två applikationer behöver samma hjälpfunktion, och det snabbaste draget är att kopiera den. Ett år senare finns det två versioner med olika rättelser, och en bugg som har lagats i den ena lever kvar i den andra. Twelve-Factor kallar flera appar som delar samma kod för ett brott mot principen och pekar ut lösningen: flytta den gemensamma delen till ett bibliotek, dra in det via beroendehanteraren och låt varje kodbas ange vilken version den använder.

En hemlighet hamnar i historiken

En anslutningssträng eller en API-nyckel committas i en konfigurationsfil, och från den stunden ligger den i historiken som varje klon bär med sig, även efter att en senare commit har tagit bort raden. Testet från Twelve-Factor är värt att köra före varje commit: skulle den här kodbasen kunna publiceras som öppen källkod i dag utan att en enda inloggningsuppgift läcker? Allt som skulle fälla det testet hör hemma i miljön för varje driftsättning.

En genererad fil redigeras för hand

En fil som har skrivits av ett skript ser ut som vilken källfil som helst, så någon rättar ett stavfel direkt i den. Nästa körning av skriptet skriver tillbaka den gamla texten och rättelsen försvinner utan ett ljud. Märk genererade sökvägar tydligt, låt generatorn vara deras enda skribent och skicka varje rättelse till generatorns indata.

Väggarna inuti nöts ner

Under tidspress hamnar en databasfråga direkt i en hanterare för förfrågningar, eftersom det är den kortaste vägen, och lagerindelningen som projektet började med slutar beskriva koden. Varje genväg är liten. Tillsammans gör de en ändring på två rader till en som rör vid varje lager. En nedskriven regel hjälper, granskning hjälper mer, och en kontroll som fäller bygget när ett lager griper in i ett annat hjälper mest.

En standard täcker halva koden

En linter som sattes upp för frontenden lämnar skalskripten och SQL-koden i fred, och just de filerna samlar på sig exakt de problem som lintern skulle stoppa. En ribba för hela kodbasen måste namnge varje språk i kodbasen, och det är det dolda priset för ett polyglott repo.

En fork blir en andra produkt

Ett team kopierar ett repo för att starta en närliggande produkt. Båda kopiorna behåller samma rotcommit, så enligt Twelve-Factor är de fortfarande en kodbas, som nu betjänar två appar, och glidningen som beskrevs ovan börjar i samma skala som ett helt repo. En jämförelse av rotcommits visar vilken situation man befinner sig i:

```
git -C app-a rev-list --max-parents=0 HEAD
git -C app-b rev-list --max-parents=0 HEAD
```

Identisk utdata betyder en gemensam rot och en gemensam kodbas; olika utdata betyder två separata historiker.

Läs mer

- I. Codebase — regeln om en kodbas och många driftsättningar, och testet med den gemensamma rotcommiten bakom den.
- II. Dependencies — att deklarera varje beroende i en manifestfil och isolera det, så att en kodbas anger exakt vad den drar in.
- III. Config — lackmustestet med öppen källkod för vad som måste hållas utanför en kodbas.
- gitsubmodules — hur git bäddar in ett repo i ett annat medan vart och ett behåller sin egen historik.
- git-rev-list — referensen för att lista commits, inklusive rotcommits i en historik.

Ordlista

Välj utifrån era faktiska förutsättningar

Beskriv arbetsflödet, befintliga gränssnitt och användarna som behöver förändringen. Vi kan definiera en testbar leverans och en överlämning som gör att teamet kan underhålla lösningen.

Se vad det kostar Hör av er →

Gör begreppen praktiska

Fråga hur ”Vad är en kodbas?” 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/codebase/