Engineer

YAML

ett inställningsformat som en människa fortfarande kan läsa

Vad det är

YAML är ett textformat för strukturerade data som markerar strukturen med indrag, kolon och bindestreck i stället för klamrar och citattecken, och det är därför det brukar väljas för konfiguration som människor skriver och granskar för hand. Namnet är en rekursiv akronym, ”YAML Ain’t Markup Language”. Filerna slutar på `.yaml` eller `.yml`, och RFC 9512, som publicerades i februari 2024, registrerade medietypen `application/yaml` och angav `.yaml` som föredragen filändelse.

Under den sparsamma interpunktionen finns en liten datamodell. Den består av mappningar, som parar ihop nycklar med värden, sekvenser, som är ordnade listor, och skalärer: strängar, tal, booleska värden och null. Runt den modellen lägger språket till det som en människa som redigerar en fil vill ha: kommentarer som börjar med `#`, blockskalärer som behåller radbrytningar med `|` eller viker ihop dem med `>`, flera dokument i en fil åtskilda av `---`, ankare och alias (`&name` och `*name`) som låter ett värde återanvändas på andra ställen, och taggar som anger en typ rakt ut.

Det första ramverket för YAML skrevs i Perl 2001. Version 1.1 av specifikationen följde 2005, och version 1.2 kom 2009 med huvudsyftet att göra YAML till en strikt övermängd av JSON, och med många av de äldre reglerna för implicita typer borttagna. Revision 1.2.2, som publicerades i oktober 2021, rättade fel och förtydligade formuleringar utan att ändra språket. Mycket programvara läser ändå fortfarande enligt de äldre reglerna: dokumentationen för Ansible noterar att PyYAML och libyaml implementerar YAML 1.1.

YAML står alltså mellan två grannar. JSON är en strikt delmängd av YAML, kortfattad och entydig, gjord för att program ska utbyta data. TOML är ett plattare konfigurationsformat där varje sträng står inom citattecken, så att ett

värdes typ aldrig gissas. YAML erbjuder mer än båda, och den extra friheten är just där problemen kommer ifrån.

Använd begreppet i ett beslut

Använd ”Vad är YAML?” i ett konkret beslut. Vi kan knyta begreppet till systemet ni bygger eller köper. Hela uppsättningen nedskriven som filer, så att den kan återskapas exakt och aldrig bara finns i en enda persons huvud.

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. Skrev 1 000 RBAC-regler som säkrar Linux-servrar — 1,000 åtkomstregler, skrivna för att täppa till en säkerhetslucka

Skrev 1 000 RBAC-regler som säkrar Linux-servrar Hör av er →

Varför det spelar roll

Det verkliga valet är vilket format en fil ska få när människor kommer att underhålla den i flera år: beskrivningar av driftsättningar, flödesdefinitioner för CI, serverkonfiguration. Playbooks i Ansible är YAML, och det är konfigurationsfilerna för många andra infrastrukturverktyg också, så för mycket av det arbetet är beslutet redan fattat högre upp i kedjan, och det som återstår att bestämma är hur strikt filen ska skrivas.

Argumentet för YAML är läsbarheten. En granskare kan läsa en diff av det utan att räkna klamrar, en kommentar kan förklara varför ett värde är som det är, och ett skript eller ett certifikat på flera rader får plats i en blockskalär utan att något behöver escapas. Mot det står att blanktecken bär betydelse, så att en nyckel som är indragen två mellanslag för långt kan hamna i en annan mappning medan filen ändå går att tolka. Tabbar är helt förbjudna för indrag. Värden utan citattecken får sin typ av hur de stavas, och stavningsreglerna skiljer sig mellan 1.1 och 1.2. Mallar lägger till en andra syntax ovanpå: i Ansible måste ett värde som börjar med `{{` stå inom citattecken, annars tar tolken det för en mappning.

Det som avgör beslutet är hur filen ska kontrolleras. YAML med ett lintverktyg och en vana att sätta strängar inom citattecken är ett bekvämt format för konfiguration som ett team redigerar tillsammans. YAML utan någotdera är ett format där en giltig fil kan säga något annat än vad författaren menade. För data som skickas mellan program är JSON det säkrare standardvalet, och för en kort inställningsfil med få nivåer av nästling är TOML enklare.

Så ser det ut i praktiken

Företagets infrastrukturrepo omfattar publika tjänster i drift och konfiguration av andra komponenter, däribland roller för databas, säkerhetskopiering, e-posttransport och DNS. Den dokumenterade implementeringen omfattade 19 Ansible-playbooks och 34 roller, 12 065 rader sammanlagt, en historisk siffra snarare än repots nuvarande storlek. Automatisering av säkerhetskopiering och e-post har ett separat steg för aktivering i drift utöver konfigurationen. Sammansättningen är data snarare än flaggor. Varje värddator tillhör en nivågrupp, och gruppens YAML-variabler listar de roller som värddatorn får, så en provisioneringskörning behöver inga argument för att föra varje värddator till det tillstånd som dess grupp deklarerar. Det är YAML som språk för deklarerat tillstånd, och resultatet kontrolleras i stället för att antas, eftersom en konvergerad värddator måste rapportera noll ändringar. YAML kostar tid: varje ändring tar längre tid än att redigera en fil direkt på servern, och eftersom en konvergering som bara tillämpas till hälften är värre än en som vägrar körs numera en preflight-play framför den.

Tolv tusen rader YAML behöver en maskin som läser dem innan en människa gör det. I det dokumenterade implementeringsskedet i augusti 2026 var YAML ett av språken som en kvalitetsgrind med 22 lintverktyg kontrollerade, tillsammans med HTML, CSS, JavaScript, Python, Markdown och shell. Grinden saknar varningsnivå, så varje diagnostik stoppar committen, och undertryckningar direkt i koden är helt förbjudna: en kommentar som ignorerar en regel eller ett direktiv som stänger av den är ingen nödutgång. För YAML spelar det roll, eftersom de flesta av fällorna nedan är helt giltig syntax, och bara en regel som vägrar dem fångar dem innan de når en server.

Var det går fel

Fällorna nedan har en sak gemensamt: filen tolkas utan fel och betyder något annat.

En landskod blir false. Enligt reglerna i YAML 1.1 är `yes`, `no`, `on`, `off`, `y` och `n` utan citattecken, i flera varianter av versaler och gemener, booleska värden. Kärnschemat i YAML 1.2 behåller bara `true` och `false`, i tre skrivsätt, och RFC 9512 varnar för att en tolk som följer den äldre syntaxen kan läsa `{insecure: n}` som false i stället för som bokstaven n. En lista över nordiska länder visar problemet:

```
countries:
    - DK
    - SE
    - NO # YAML 1.1 reads this as false
```

Sätt citattecken kring varje sträng som kan läsas som något annat, skriv booleska värden bara som `true` och `false`, och låt ett lintverktyg upprätthålla båda.

Ett versionsnummer tappar en siffra. Tal utan citattecken konverteras, så `version: 1.10` kommer fram som flyttalet `1.1`, och enligt reglerna i 1.1 markerar en inledande nolla ett oktalt tal, så en kod skriven som `010` blir 8. Dokumentationen för Ansible uppmanar sina användare att sätta citattecken kring en version som `"1.0"` av just det skälet. Sätt citattecken kring identifierare, koder och versioner, som är strängar som råkar innehålla siffror.

En nyckel hamnar under fel förälder. Indraget avgör vilken mappning en nyckel hör till, så en rad som flyttas två mellanslag flyttar en inställning någon annanstans utan något fel. Specifikationen kräver också att varje nyckel i en mappning är unik, och en upprepad nyckel långt ned i ett långt block är lätt att missa vid granskning, så yamllint har en regel för vart och ett av de två misstagen. Kör det innan filen används.

Att ladda en fil kör kod. Taggar kan namnge språkspecifika typer, och en laddare som respekterar dem konstruerar det objekt filen ber om. Dokumentationen för PyYAML jämför användning av `yaml.load` (med en osäker laddare) med `pickle.load`: den kan konstruera objekt och anropa Python-funktioner. Valet av laddare spelar roll; aktuella versioner kräver att den anges uttryckligen. `yaml.safe_load` känner bara igen grundläggande YAML-taggar. RFC 9512 begär att kodkörning i deserialiserare ska vara avstängd som standard. Använd den säkra laddaren för vanliga konfigurationsfiler.

En kort fil fyller minnet. Ankare och alias gör ett YAML-dokument till en graf snarare än ett träd. När varje ankare upprepar det föregående två gånger fördubblar varje ny rad storleken, så trettio korta rader beskriver mer än en miljard strängar när ett program går igenom resultatet som ett vanligt träd. RFC 9512 kallar det exponentiell dataexpansion, attacken billion laughs, och möter det med processorer som begränsar rekursionsdjupet för ankare. Ta bara emot YAML utifrån genom en sådan processor, och testa gränsen mot den tolk som faktiskt används.

Läs mer

- YAML 1.2.2 specification — själva språket, inklusive kärnschemat som avgör hur värden utan citattecken får sin typ.
- RFC 9512: YAML Media Type — registreringen av `application/yaml`, med säkerhetsanteckningar om booleska värden, kodkörning och expansion av alias.
- YAML specification changes — vad varje revision ändrade, från 1.2.1 till 1.2.2.
- Ansible: YAML Syntax — den delmängd som Ansible använder, med dess fällor kring citattecken.
- yamllint rules — kontrollerna för sanningsvärden (truthy), dubblerade nycklar, oktala tal och indrag.
- PyYAML documentation — skillnaden mellan `load` och `safe_load`, med bibliotekets egna ord.

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 YAML?” 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/yaml/