Engineer

YAML

et innstillingsformat et menneske fortsatt kan lese

Hva det er

YAML, på disk en fil som slutter på `.yaml` eller `.yml`, er et tekstformat for strukturerte data som markerer strukturen med innrykk, kolon og bindestreker i stedet for klammer og anførselstegn, og det er derfor det er det vanlige valget for konfigurasjon som folk skriver og gjennomgår for hånd. Navnet er et rekursivt akronym, «YAML Ain't Markup Language». RFC 9512, publisert i februar 2024, registrerte medietypen `application/yaml` og pekte ut `.yaml` som den foretrukne endelsen.

Under den lette tegnsettingen ligger en liten datamodell. Det finnes avbildninger (*mappings*), som kobler nøkler til verdier, sekvenser, som er ordnede lister, og skalarer: strenger, tall, boolske verdier og null. Rundt den modellen legger språket til det en person som redigerer en fil, vil ha: kommentarer som starter med `#`, blokkskalarer som beholder linjeskift med `|` eller bretter dem sammen med `>`, flere dokumenter i én fil skilt med `---`, ankere og aliaser (`&name` og `*name`) som lar én verdi gjenbrukes et annet sted, og tagger som navngir en type direkte.

Det første YAML-rammeverket ble skrevet i Perl i 2001. Versjon 1.1 av spesifikasjonen fulgte i 2005, og versjon 1.2 i 2009, med hovedvekt på å gjøre YAML til et strengt supersett av JSON og med mange av de eldre reglene for implisitt typing fjernet. Revisjon 1.2.2, publisert i oktober 2021, rettet feil og klargjorde ordlyden uten å endre språket. Mye programvare leser likevel fortsatt etter de eldre reglene: dokumentasjonen til Ansible påpeker at PyYAML og libyaml implementerer YAML 1.1.

Dermed står YAML mellom to naboer. JSON er en streng delmengde av det, kortfattet og entydig, laget for at programmer skal utveksle data. TOML er et flatere konfigurasjonsformat der hver streng står i anførselstegn, så typen til en

verdi aldri blir gjettet. YAML tilbyr mer enn begge, og den ekstra friheten er nettopp der problemene kommer fra.

Bruk begrepet i en beslutning

Bruk «Hva er YAML?» i en konkret beslutning. Vi kan knytte begrepet til systemet dere bygger eller kjøper. Hele oppsettet skrevet ned som filer, slik at det kan gjenskapes nøyaktig og aldri bare finnes i hodet på én person.

Se tjenestene våre Ta kontakt →

Undersøk antakelsene bak ordet

Bruk forklaringen og det tilknyttede arbeidet til å formulere spørsmål for prosjektet. Vi kan undersøke antakelsene sammen før et verktøy velges. Skrev 1 000 RBAC-regler for å sikre Linux-servere — 1,000 tilgangsregler skrevet for å tette et sikkerhetshull

Skrev 1 000 RBAC-regler for å sikre Linux-servere Ta kontakt →

Hvorfor det er viktig

Det egentlige valget er hvilket format en fil skal få når folk skal vedlikeholde den i årevis: beskrivelser av produksjonssetting, definisjoner av CI-løp, serverkonfigurasjon. Playbooks i Ansible er YAML, og det samme er konfigurasjonsfilene til mange andre infrastrukturverktøy, så for mye av det arbeidet er beslutningen allerede tatt lenger opp i kjeden, og det som gjenstår å avgjøre, er hvor strengt filene skal skrives.

Argumentet for YAML er lesbarhet. Den som gjennomgår en endring, kan lese en diff av det uten å telle klammer, en kommentar kan forklare hvorfor en verdi er som den er, og et skript eller et sertifikat over flere linjer får plass i en blokkskalar uten escaping. Mot det står at mellomrom bærer mening, så en nøkkel som er rykket inn to mellomrom for langt, kan havne i en annen avbildning mens filen fortsatt lar seg tolke. Tabulatorer er helt forbudt som innrykk. Verdier uten anførselstegn får type etter hvordan de staves, og reglene for stavingen er ulike i 1.1 og 1.2. Maler legger en syntaks til oppå: i Ansible må en verdi som begynner med `{{` , stå i anførselstegn, ellers tar tolkeren den for en avbildning.

Det som avgjør valget, er hvordan filen skal sjekkes. YAML med en linter og en vane med å sette strenger i anførselstegn er et behagelig format for konfigurasjon et team redigerer sammen. YAML uten noen av delene er et format der en gyldig fil kan si noe annet enn det forfatteren mente. For data som sendes mellom programmer, er JSON det tryggere standardvalget, og for en kort innstillingsfil med få nivåer av nesting er TOML enklere.

Slik ser det ut i praksis

Selskapets infrastrukturrepo dekker offentlig tilgjengelige tjenester som er satt i drift, og konfigurasjon for andre komponenter, blant annet roller for databaser, sikkerhetskopiering, e-posttransport og DNS. Den dokumenterte implementeringen besto av 19 Ansible-playbooks og 34 roller, til sammen 12 065 linjer, en historisk opptelling og ikke størrelsen på det nåværende repoet. Automatisering av sikkerhetskopiering og e-post har et eget trinn for aktivering i drift, atskilt fra konfigurasjonen. Sammensetningen er data og ikke flagg. Hver vert tilhører en nivågruppe, og gruppens YAML-variabler angir rollene verten får, slik at en klargjøringskjøring uten argumenter kan bringe hver vert til tilstanden gruppen deklarerer. Her brukes YAML som språket for deklarert tilstand, og vellykket resultat blir kontrollert og ikke antatt, fordi en konvergert vert må rapportere null endringer. YAML-en koster tid: hver endring går tregere enn å redigere en fil på serveren, og fordi en konvergering som bare delvis anvender endringene, er verre enn en som nekter å kjøre, kjører det nå en preflight-play foran den.

Tolv tusen linjer YAML trenger en maskin som leser dem før et menneske gjør det. På det dokumenterte implementeringsstadiet i august 2026 var YAML ett av språkene en commit-port med 22 lintere sjekket, sammen med HTML, CSS, JavaScript, Python, Markdown og shell. Porten har ikke noe advarselsnivå, så hver diagnostikk stopper commiten, og undertrykking i selve koden er forbudt uten unntak: en ignore-kommentar eller et disable-direktiv i filen er ingen fluktvei. For YAML betyr det noe, fordi de fleste fellene nedenfor er helt gyldig syntaks, og bare en regel som avviser dem, fanger dem før de når en vert.

Hvor det går galt

Fellene nedenfor har ett trekk felles: filen lar seg tolke uten feil og betyr noe annet.

En landkode blir til false. Etter reglene i YAML 1.1 er `yes`, `no`, `on`, `off`, `y` og `n` uten anførselstegn boolske verdier, i flere varianter av store og små bokstaver. Kjerneskjemaet i YAML 1.2 beholder bare `true` og `false`, i tre varianter av store og små bokstaver, og RFC 9512 advarer om at en tolker som følger den eldre syntaksen, kan lese `{insecure: n}` som false og ikke som bokstaven n. En liste over nordiske land viser problemet:

```
countries:
    - DK
    - SE
    - NO # YAML 1.1 leser dette som false
```

Sett hver streng som kan leses som noe annet, i anførselstegn, skriv boolske verdier bare som `true` og `false`, og la en linter håndheve begge deler.

Et versjonsnummer mister et siffer. Tall uten anførselstegn blir konvertert, så `version: 1.10` kommer fram som flyttallet `1.1`, og etter reglene i 1.1 markerer en innledende null et oktalt tall, så en kode skrevet som `010` blir 8. Dokumentasjonen til Ansible sier at en versjon som `"1.0"` skal stå i anførselstegn, av nettopp denne grunnen. Sett identifikatorer, koder og versjoner i anførselstegn; de er strenger som tilfeldigvis inneholder sifre.

En nøkkel havner under feil forelder. Innrykket avgjør hvilken avbildning en nøkkel tilhører, så én linje som er forskjøvet to mellomrom, flytter en innstilling et annet sted uten noen feilmelding. Spesifikasjonen krever også at hver nøkkel i en avbildning er unik, og en gjentatt nøkkel langt nede i en lang blokk er lett å overse i en gjennomgang, så yamllint har en regel for hver av de to feilene. Kjør den før filen tas i bruk.

Å laste inn en fil kjører kode. Tagger kan navngi typer som hører til ett bestemt programmeringsspråk, og en laster som respekterer dem, bygger det objektet filen ber om. Dokumentasjonen til PyYAML sammenligner bruk av `yaml.load` (med en utrygg laster) med `pickle.load`: den kan bygge objekter og kalle Python-funksjoner. Valget av laster betyr noe; gjeldende versjoner krever at den angis uttrykkelig. `yaml.safe_load` gjenkjenner bare grunnleggende YAML-tagger. RFC 9512 ber om at kjøring av kode i deserialiserere er slått av som standard. Bruk den trygge lasteren til vanlige konfigurasjonsfiler.

En kort fil fyller minnet. Ankere og aliaser gjør et YAML-dokument til en graf heller enn et tre. Når hvert anker gjentar det forrige to ganger, dobler hver ny linje størrelsen, så tretti korte linjer beskriver mer enn en milliard strenger når et program går gjennom resultatet som et vanlig tre. RFC 9512 kaller dette eksponentiell dataekspansjon, angrepet kjent som *billion laughs*, og svarer med prosessorer som begrenser rekursjonsdybden for ankere. Ta bare imot YAML utenfra gjennom en slik prosessor, og test grensen mot tolkeren som faktisk er i bruk.

Les mer

- YAML 1.2.2 specification — selve språket, også kjerneskjemaet som avgjør hvilken type verdier uten anførselstegn får.
- RFC 9512: YAML Media Type — registreringen av `application/yaml`, med sikkerhetsmerknader om boolske verdier, kjøring av kode og utvidelse av aliaser.
- YAML specification changes — hva hver revisjon endret, fra 1.2.1 til 1.2.2.
- Ansible: YAML Syntax — delmengden Ansible bruker, med fellene rundt anførselstegn.
- yamllint rules — sjekkene for sannhetsverdier, dupliserte nøkler, oktale tall og innrykk.
- PyYAML documentation — forskjellen på `load` og `safe_load`, med bibliotekets egne ord.

Ordliste

Velg ut fra deres faktiske rammer

Ta med utrullingsløpet, driftsrisikoene og dem som har vakt. Vi kan avgrense en endring med kontroller, en vei tilbake og tydelig ansvar.

Se hva det koster Ta kontakt →

Gjør begrepene praktiske

Spør hvordan «Hva er YAML?» gjelder i organisasjonen deres. Et konkret eksempel fra dagens system er nok til å starte samtalen.

Se tjenestene våre

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

Ta med deg denne siden

Visittkort

Laget med kjærlighet av D.

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

Opphavsrett © 2025 – nå · Engineer ApS · Alle rettigheter forbeholdt.

Skrevet ut fra https://engineer.company/nb/glossary/yaml/