Engineer

YAML

et format til indstillinger et menneske kan læse

Hvad det er

YAML er et tekstformat til strukturerede data, hvor strukturen udtrykkes med indrykning, koloner og bindestreger i stedet for klammer og anførselstegn, og derfor er det det sædvanlige valg til konfiguration, som mennesker skriver og gennemgår i hånden. Navnet er et rekursivt akronym, »YAML Ain't Markup Language«. Filerne hedder `.yaml` eller `.yml`, og RFC 9512 fra februar 2024 registrerede medietypen `application/yaml` og gjorde `.yaml` til den foretrukne endelse.

Bag den sparsomme tegnsætning ligger en lille datamodel. Der er mappings, som parrer nøgler med værdier, sekvenser, som er ordnede lister, og skalarer: strenge, tal, sandhedsværdier og null. Omkring modellen lægger sproget det, en person med en fil åben i editoren har brug for: kommentarer, der starter med `#`, blokskalarer, der bevarer linjeskift med `|` eller folder dem sammen med `>`, flere dokumenter i én fil adskilt af `---`, ankre og aliasser (`&navn` og `*navn`), så én værdi kan genbruges andre steder, og tags, der navngiver en type direkte.

Det første YAML-bibliotek blev skrevet i Perl i 2001. Version 1.1 af specifikationen kom i 2005 og version 1.2 i 2009, hvor hovedformålet var at gøre YAML til en streng overmængde af JSON, og hvor mange af de gamle regler for implicit typning forsvandt. Revision 1.2.2 fra oktober 2021 rettede fejl og præciserede formuleringer uden at ændre selve sproget. Meget software læser stadig efter de ældre regler: Ansibles dokumentation oplyser, at PyYAML og libyaml implementerer YAML 1.1.

Dermed står YAML mellem to naboer. JSON er dets strenge delmængde, kompakt og entydig og lavet til, at programmer udveksler data. TOML er et fladere konfigurationsformat, hvor hver streng står i anførselstegn, så en værdis

type aldrig skal gættes. YAML kan mere end begge, og netop den ekstra frihed er kilden til dets problemer.

Brug begrebet i en beslutning

Brug »Hvad er YAML?« i en konkret beslutning. Vi kan knytte begrebet til det system, I bygger eller køber. Hele opsætningen skrevet ned som filer, så den kan genskabes præcist og aldrig kun findes i én persons hoved.

Se vores ydelser Kontakt os →

Undersøg antagelserne bag ordet

Brug forklaringen og det tilknyttede arbejde til at formulere spørgsmål til jeres projekt. Vi kan undersøge antagelserne sammen, før et værktøj vælges. Har skrevet 1.000 RBAC-regler til Linux-servere — 1,000 adgangsregler skrevet for at lukke et sikkerhedshul

Har skrevet 1.000 RBAC-regler til Linux-servere Kontakt os →

Hvorfor det betyder noget

Det egentlige valg handler om, hvilket format en fil skal have, når mennesker skal vedligeholde den i årevis: beskrivelser af udrulninger, definitioner af byggepipelines, serverkonfiguration. Ansible-playbooks er YAML, og det samme gælder konfigurationsfilerne i mange andre infrastrukturværktøjer, så for meget af det arbejde er valget allerede truffet. Tilbage står spørgsmålet om, hvor stramt filerne skal skrives.

Argumentet for YAML er læsbarhed. En reviewer kan læse en diff uden at tælle klammer, en kommentar kan forklare, hvorfor en værdi er, som den er, og et script på flere linjer eller et certifikat passer ind i en blokskalar uden escaping. Over for det står, at mellemrum bærer betydning: en nøgle, der er rykket to mellemrum for langt ind, kan havne i en anden mapping, mens filen stadig parser. Tabulatorer er helt forbudt til indrykning. Værdier uden anførselstegn får deres type ud fra stavemåden, og stavereglerne er forskellige i 1.1 og 1.2. Skabeloner lægger en ekstra syntaks ovenpå: i Ansible skal en værdi, der begynder med `{{`, stå i anførselstegn, ellers tager parseren den for en mapping.

Det, der afgør sagen, er, hvordan filen bliver tjekket. YAML med en linter og en vane med at sætte strenge i anførselstegn er et behageligt format til konfiguration, som et team redigerer i fællesskab. YAML uden nogen af delene er et format, hvor en gyldig fil kan sige noget andet, end forfatteren mente. Til data, der sendes mellem programmer, er JSON det sikrere udgangspunkt, og til en kort indstillingsfil med få niveauer er TOML enklere.

Sådan ser det ud i praksis

Virksomhedens infrastrukturrepository dækker offentligt tilgængelige tjenester i drift og konfiguration af andre komponenter, herunder roller til database, sikkerhedskopiering, mailtransport og DNS. Den dokumenterede implementering omfattede 19 Ansible-playbooks og 34 roller, i alt 12.065 linjer, en historisk optælling frem for repositoryets nuværende størrelse. Automatisering af sikkerhedskopiering og mail har et særskilt trin for aktivering i drift ud over konfigurationen. Sammensætningen er data frem for flag. Hver vært hører til en tier-gruppe, og gruppens YAML-variabler angiver de roller, værten får, så en provisioneringskørsel ikke behøver argumenter for at bringe hver vært i den tilstand, gruppen erklærer. Her er YAML sproget for erklæret tilstand, og resultatet bliver kontrolleret i stedet for antaget, for en konvergeret vært skal rapportere nul ændringer. YAML koster tid: hver ændring tager længere tid end at rette en fil direkte på serveren, og fordi en konvergering, der kun gennemføres halvt, er værre end en, der nægter at starte, kører et preflight-play nu foran den.

Tolv tusind linjer YAML skal læses af en maskine, før et menneske ser dem. På den dokumenterede implementeringsetape i august 2026 var YAML et af de sprog, en commit-gate med 22 lintere tjekkede, sammen med HTML, CSS, JavaScript, Python, Markdown og shell. Gaten har intet advarselsniveau, så hver diagnostik standser committet, og inline-undertrykkelser er direkte forbudt: en ignorér-kommentar eller et deaktiveringsdirektiv i filen fører ingen steder hen. Det betyder meget for YAML, fordi de fleste af fælderne nedenfor er fuldt gyldig syntaks, og kun en regel, der afviser dem, fanger dem, inden de når en vært.

Hvor det går galt

Fælderne nedenfor har ét træk til fælles: filen parser fint og betyder noget andet.

En landekode bliver til false. Efter YAML 1.1-reglerne er `yes`, `no`, `on`, `off`, `y` og `n` uden anførselstegn sandhedsværdier, i flere skrivemåder med store og små bogstaver. Kerneskemaet i YAML 1.2 beholder kun `true` og `false` i tre skrivemåder, og RFC 9512 advarer om, at en parser efter den gamle syntaks kan læse `{insecure: n}` som false i stedet for bogstavet n. En liste over nordiske lande viser problemet:

```
lande:
    - DK
    - SE
    - NO # YAML 1.1 læser dette som false
```

Sæt anførselstegn om enhver streng, der kan læses som noget andet, skriv kun sandhedsværdier som `true` og `false`, og lad en linter håndhæve begge dele.

Et versionsnummer mister et ciffer. Tal uden anførselstegn bliver konverteret, så `version: 1.10` ankommer som kommatallet 1.1, og efter 1.1-reglerne markerer et foranstillet nul et oktaltal, så en kode skrevet som `010` bliver til 8. Ansibles dokumentation beder netop derfor brugerne om at skrive en version som `"1.0"`. Sæt anførselstegn om id'er, koder og versioner, for de er strenge, der tilfældigvis indeholder cifre.

En nøgle havner under den forkerte forælder. Indrykningen afgør, hvilken mapping en nøgle hører til, så én linje, der er flyttet to mellemrum, flytter en indstilling et andet sted hen uden nogen fejlmeddelelse. Specifikationen kræver desuden, at hver nøgle i en mapping er unik, og en gentaget nøgle langt nede i en lang blok er let at overse i en review, så yamllint har en regel for hver af de to fejl. Kør den, før filen tages i brug.

Indlæsning af en fil afvikler kode. Tags kan navngive sprogspecifikke typer, og en loader, der følger dem, bygger det objekt, filen beder om. PyYAMLs dokumentation sammenligner brugen af `yaml.load` (med en usikker loader) med `pickle.load`: den kan bygge objekter og kalde Python-funktioner. Valget af loader betyder noget; nuværende versioner kræver, at den angives eksplicit. `yaml.safe_load` kender kun grundlæggende YAML-tags. RFC 9512 beder om, at kodeafvikling i deserialisering som standard er slået fra. Brug den sikre loader til almindelige konfigurationsfiler.

En kort fil fylder hukommelsen op. Ankre og aliasser gør et YAML-dokument til en graf frem for et træ. Når hvert anker gentager det foregående to gange, fordobler hver ny linje størrelsen, så tredive korte linjer beskriver over en milliard strenge, når et program gennemløber resultatet som et almindeligt træ. RFC 9512 kalder det eksponentiel dataudvidelse, et billion laughs-angreb, og svarer med processorer, der begrænser dybden af ankerrekursion. Tag kun imod YAML udefra gennem sådan en processor, og afprøv grænsen mod den parser, der faktisk er i brug.

Læs mere

- YAML 1.2.2 specification — selve sproget, herunder kerneskemaet, der afgør, hvordan værdier uden anførselstegn får deres type.
- RFC 9512: YAML Media Type — registreringen af `application/yaml` med sikkerhedsnoter om sandhedsværdier, kodeafvikling og aliasudvidelse.
- YAML specification changes — hvad hver revision ændrede, fra 1.2.1 til 1.2.2.
- Ansible: YAML Syntax — den delmængde, Ansible bruger, med faldgruberne omkring anførselstegn.
- yamllint rules — reglerne for sandhedsværdier, dublerede nøgler, oktaltal og indrykning.
- PyYAML documentation — forskellen på `load` og `safe_load` med bibliotekets egne ord.

Ordliste

Vælg ud fra jeres faktiske vilkår

Medbring udrulningsforløbet, driftsrisiciene og dem, der har vagten. Vi kan afgrænse en ændring med kontroller, en vej tilbage og tydeligt ansvar.

Se, hvad det koster Kontakt os →

Gør begreberne praktiske

Spørg, hvordan »Hvad er YAML?« gælder i jeres organisation. Et konkret eksempel fra jeres nuværende system er nok til at begynde samtalen.

Se vores ydelser

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

Tag denne side med dig

Visitkort

Lavet med kærlighed af D.

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

Copyright © 2025 – nu · Engineer ApS · Alle rettigheder forbeholdes.

Udskrevet fra https://engineer.company/da/glossary/yaml/