Engineer

YAML

un format de paramètres lisible par une personne

Ce que c’est

YAML est un format texte pour les données imbriquées qui indique la structure par l’indentation, les deux-points et les tirets plutôt que par des crochets et des guillemets. C’est pourquoi il est courant pour les configurations que l’on écrit et relit à la main. Son nom est un acronyme récursif, « YAML Ain’t Markup Language ». Les fichiers se terminent par `.yaml` ou `.yml`, et la RFC 9512, publiée en février 2024, a enregistré le type de média `application/yaml` en désignant `.yaml` comme extension à privilégier.

Sous cette ponctuation légère se trouve un petit modèle de données. Il comprend des tables d’association, qui relient des clés à des valeurs, des séquences, qui sont des listes ordonnées, et des scalaires : chaînes, nombres, booléens et valeur nulle. Autour de ce modèle, le langage ajoute ce qu’attend une personne qui modifie un fichier : des commentaires commençant par `#`, des scalaires en bloc qui conservent les sauts de ligne avec `|` ou les replient avec `>`, plusieurs documents dans un fichier séparés par `---`, des ancres et des alias (`&name` et `*name`) qui permettent de réutiliser une valeur ailleurs, et des balises qui nomment directement un type.

Le premier framework YAML était écrit en Perl, en 2001. La version 1.1 de la spécification a suivi en 2005, puis la version 1.2 en 2009, avec pour objectif principal de faire de YAML un sur-ensemble strict de JSON et en supprimant nombre d’anciennes règles de typage implicite. La révision 1.2.2, publiée en octobre 2021, a corrigé des erreurs et clarifié la rédaction sans modifier le langage. Beaucoup de logiciels suivent pourtant encore les anciennes règles : la documentation d’Ansible indique que PyYAML et libyaml implémentent YAML 1.1.

YAML se situe donc entre deux voisins. JSON en est un sous-ensemble strict, concis et sans ambiguïté, conçu pour les échanges entre programmes. TOML est un format de configuration plus plat dans lequel toutes les chaînes sont entre guillemets, de sorte que le type d'une valeur n'est jamais deviné. YAML offre davantage que l'un ou l'autre, et cette liberté supplémentaire est précisément la source de ses difficultés.

Utilisez le concept pour prendre une décision

Utilisez « Qu'est-ce que YAML ? » pour éclairer une décision réelle. Nous pouvons vous aider à relier ce concept au système que vous construisez ou achetez. Toute la configuration décrite dans des fichiers, pour la recréer à l'identique sans qu'elle ne dépende de la mémoire d'une seule personne.

Découvrir nos prestations Contactez-nous →

Vérifiez les hypothèses derrière le terme

Appuyez-vous sur l'explication et les réalisations liées pour formuler les questions de votre projet. Nous pouvons examiner les hypothèses avec vous avant le choix d'un outil. 1 000 règles RBAC pour sécuriser Linux — 1,000 règles d'accès écrites pour combler une faille de sécurité

1 000 règles RBAC pour sécuriser Linux Contactez-nous →

Pourquoi c'est important

Le vrai choix est le format à donner à un fichier qui sera entretenu pendant des années : descriptions de déploiement, workflows de CI, configuration de serveurs. Les playbooks Ansible sont en YAML, comme les fichiers de configuration de nombreux autres outils d'infrastructure. Pour une grande partie de ce travail, le choix a donc déjà été fait en amont ; il reste à décider avec quelle rigueur l'écrire.

L'argument en faveur de YAML est la lisibilité. On peut en relire un diff sans compter les accolades, un commentaire peut expliquer la raison d'une valeur, et un script multiligne ou un certificat tient dans un scalaire en bloc sans échappement. En contrepartie, les espaces portent du sens : une clé indentée de deux espaces de trop peut aboutir dans une autre table d'association alors que le fichier reste analysable. Les tabulations sont entièrement interdites pour l'indentation. Les valeurs sans guillemets sont typées d'après leur écriture, et les règles diffèrent entre 1.1 et 1.2. Le recours à des gabarits ajoute une seconde syntaxe : dans Ansible, une valeur qui commence par `{{` doit être entre guillemets, sinon l'analyseur y voit une table d'association.

Ce qui fait pencher la décision est la manière dont le fichier sera vérifié. Avec un linter et l'habitude de mettre les chaînes entre guillemets, YAML est un format confortable pour une configuration modifiée en équipe. Sans l'un ni l'autre, un fichier YAML valide peut dire autre chose que ce que son auteur voulait exprimer.

Pour les données échangées entre programmes, JSON est le choix par défaut le plus sûr ; pour un petit fichier de paramètres avec peu de niveaux d'imbrication, TOML est plus simple.

À quoi cela ressemble en pratique

Le dépôt d'infrastructure de l'entreprise couvre des services déployés accessibles au public et la configuration d'autres composants, dont les rôles de base de données, de sauvegarde, de transport du courrier et de DNS. L'implémentation documentée comptait 19 playbooks Ansible et 34 rôles, soit 12 065 lignes au total, un décompte historique et non la taille actuelle du dépôt. L'automatisation des sauvegardes et du courrier comporte une étape d'activation opérationnelle distincte de la configuration. La composition est exprimée par des données plutôt que par des options. Chaque hôte appartient à un groupe de niveau, dont les variables YAML listent les rôles à lui attribuer : une exécution de provisionnement n'a donc besoin d'aucun argument pour amener chaque hôte dans l'état déclaré par son groupe. YAML sert ici de langage de l'état déclaré, et la réussite se vérifie au lieu de se supposer, puisqu'un hôte ayant convergé doit signaler zéro modification. YAML coûte du temps : chaque changement est plus lent qu'une modification directe d'un fichier sur le serveur. Comme une convergence qui n'applique que la moitié des changements est pire qu'une qui refuse de s'exécuter, un play de vérification préalable s'exécute désormais en amont.

Douze mille lignes de YAML doivent être lues par une machine avant de l'être par une personne. Au stade d'implémentation d'août 2026 décrit dans la documentation, YAML était l'un des langages vérifiés par un contrôle de commit à 22 linters, aux côtés de HTML, CSS, JavaScript, Python, Markdown et shell. Ce contrôle n'a pas de niveau d'avertissement : chaque diagnostic bloque le commit, et les suppressions dans le code sont formellement interdites. Dans le fichier, ni un commentaire demandant d'ignorer une règle ni une directive de désactivation ne permettent de contourner le contrôle. C'est important pour YAML : la plupart des pièges ci-dessous sont parfaitement valides sur le plan syntaxique, et seule une règle qui les refuse les intercepte avant qu'ils n'atteignent un hôte.

Où les problèmes surviennent

Les pièges ci-dessous ont un point commun : le fichier s'analyse sans erreur, mais signifie autre chose.

Un code pays devient faux. Selon les règles de YAML 1.1, `yes`, `no`, `on`, `off`, `y` et `n`, sans guillemets et sous plusieurs combinaisons de majuscules et de minuscules, sont des booléens. Le schéma central de YAML 1.2 ne conserve que `true` et `false`, sous trois combinaisons de casse, et la RFC 9512 avertit qu'un analyseur suivant l'ancienne syntaxe peut lire `{insecure: n}` comme faux plutôt que comme la lettre n. Une liste de pays nordiques illustre le problème :

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

Mettez entre guillemets toute chaîne qui pourrait être interprétée autrement, écrivez les booléens uniquement sous la forme `true` et `false`, et faites respecter ces deux règles par un linter.

Un numéro de version perd un chiffre. Les nombres sans guillemets sont convertis : `version: 1.10` devient le nombre à virgule flottante 1,1, et selon les règles de 1.1 un zéro initial indique un nombre octal, si bien qu'un code écrit `010` devient 8. La documentation d'Ansible recommande de mettre une version telle que `"1.0"` entre guillemets pour cette raison précise. Faites-le pour les identifiants, les codes et les versions : ce sont des chaînes qui se trouvent contenir des chiffres.

Une clé se retrouve sous le mauvais parent. L'indentation détermine à quelle table d'association appartient une clé : décaler une ligne de deux espaces déplace un paramètre ailleurs sans provoquer d'erreur. La spécification exige aussi que chaque clé d'une table d'association soit unique. Une clé répétée au fond d'un long bloc passe facilement inaperçue à la relecture : yamllint possède donc une règle pour chacune de ces deux erreurs. Exécutez-le avant d'utiliser le fichier.

Charger un fichier exécute du code. Les balises peuvent nommer des types propres à un langage, et un chargeur qui les respecte construit tout objet que le fichier demande. La documentation de PyYAML compare l'utilisation de `yaml.load` (avec un chargeur non sûr) à `pickle.load` : cet appel peut construire des objets et invoquer des fonctions Python. Le choix du chargeur compte ; les versions actuelles exigent qu'il soit précisé explicitement. `yaml.safe_load` ne

reconnaît que les balises YAML de base. La RFC 9512 demande que l'exécution de code dans les désérialiseurs soit désactivée par défaut. Utilisez le chargeur sûr pour les fichiers de configuration ordinaires.

Un petit fichier remplit la mémoire. Les ancres et les alias font d'un document YAML un graphe plutôt qu'un arbre. Lorsque chaque ancre répète deux fois la précédente, chaque ligne ajoutée double la taille : trente lignes courtes décrivent ainsi plus d'un milliard de chaînes dès qu'un programme parcourt le résultat comme un simple arbre. La RFC 9512 nomme cela une expansion exponentielle des données, l'attaque billion laughs, et y répond par des processeurs qui limitent la profondeur de récursion des ancres. N'acceptez du YAML extérieur qu'au travers d'un tel processeur et testez la limite avec l'analyseur réellement utilisé.

Pour aller plus loin

- YAML 1.2.2 specification — le langage lui-même, dont le schéma central qui détermine le typage des valeurs sans guillemets.
- RFC 9512: YAML Media Type — l'enregistrement d' `application/yaml` , avec des notes de sécurité sur les booléens, l'exécution de code et le développement des alias.
- YAML specification changes — les changements de chaque révision, de 1.2.1 à 1.2.2.
- Ansible: YAML Syntax — le sous-ensemble utilisé par Ansible et ses pièges de mise entre guillemets.
- yamllint rules — les contrôles des valeurs interprétées comme vraies ou fausses, des clés en double, des nombres octaux et de l'indentation.
- PyYAML documentation — la différence entre `load` et `safe_load` , selon la propre documentation de la bibliothèque.

Glossaire

Choisissez selon vos contraintes réelles

Présentez le parcours de déploiement, les risques opérationnels et les personnes d'astreinte. Nous pouvons définir un changement avec des vérifications, une procédure de reprise et une responsabilité claire.

Consulter les tarifs Contactez-nous →

Rendez la terminologie concrète

Demandez comment « Qu'est-ce que YAML ? » s'applique à votre organisation. Un exemple concret tiré de votre système actuel suffit pour engager la discussion.

Découvrir nos prestations

welcome@engineer.company +45 50 10 61 84 Copenhague, Danemark

Emportez cette page avec vous

Fiche de contact

Créé avec soin et amour par D.

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

Droit d'auteur © 2025 – aujourd'hui · Engineer ApS · Tous droits réservés.

Imprimé depuis https://engineer.company/fr/glossary/yaml/