# YAML

D. Leeuw

27 mei 2024
v.0.3.0

# Over dit Document

## 0.1 Leerdoelen

Na het bestuderen van dit document heeft de lezer kennis van:

- wat YAML is
- hoe key/value pairs te definiëren in YAML
- hoe lists te definiëren in YAML
- hoe dictionaries te definiëren in YAML
- hoe YAML files geschreven moeten worden

## 0.2 Voorkennis

Voor dit document is geen specifieke voorkennis vereist.

# Inhoudsopgave

# Hoofdstuk 1

# Introductie

YAML is een taal die data serialiseert zodat deze makkelijk begrepen kan worden door mensen. YAML staat voor YAML Ain't Markup Language. Het is dus niet bedoelt als taal zoals HTML, SGML etc. die aangeeft hoe data (tekst) opgemaakt (Markup) moet worden. YAML kan beter beschreven worden als een taal om data een betekenis te geven. Het is een taal voor configuratie bestanden. Het lijkt qua functie op JSON en XML. Het voordeel van YAML is dat het makkelijk leesbaar is voor mensen en eenvoudig in elkaar zit. YAML stamt uit 2001, meer informatie over de beschikbaarheid van YAML kan gevonden worden op de YAML website: yaml.org.

YAML maakt gebruik van key/value pairs. Het programma dat YAML gebruikt in zijn configuratie bepaalt welke keys geldig zijn binnen een YAML bestand en dus ook welke waardes (values) er toegekend kunnen worden aan een key. In dit document hebben we wat keys en values verzonnen om YAML uit te kunnen leggen, maar de genoemde keys en de mogelijke waarden zijn volledig fictief.

YAML bestanden hebben de extensie .yaml of .yml.

# Hoofdstuk 2

# Syntax

## 2.1 Documents

YAML bestanden beginnen met 3 dashes en eindigen met 3 punten.

```
---
tekst: "Een YAML document"
...
```

Het eindigen met 3 punten is optioneel en wordt vaak weggelaten.

De drie dashes worden ook gebruikt om documenten te scheiden. Je kunt met YAML dus in een tekstbestand meerdere documenten zetten door deze steeds te scheiden met dashes.

```
---
tekst1: "Dit is document 1"
---
tekst2: "Dit is document 2"
...
```

Het stukken tussen de dashes en de eventuele punten noemen we velden. Velden beginnen aan het begin van de regel en eindigen als er een newline wordt gezien.

## 2.2 Commentaar

Het is binnen YAML mogelijk om commentaar te schrijven zodat er verdere uitleg gegeven kan worden bij de constructies. Het teken om aan te geven dat er commentaar komt is het hekje (#). Commentaar start bij het

hekje en loopt door tot de eerst volgende newline. Commentaar kan aan het begin van een regel gebruikt worden, maar ook na een veld:

```
---
# Dit is commentaar
ip: 192.168.42.1 # Ook commentaar
...
```

## 2.3 Inspringen en nesting

YAML gebruikt spaties om de verschillende structuren te groeperen (sequences, mappings, key/value pairs). Er mogen binnen YAML geen tabs gebruikt worden om in te springen en het aantal spaties is van belang om aan te gegeven wat er bij elkaar hoort.

```
---
- host: host01
  # Dit hoort bij elkaar
  ip: 1.1.1.1
  nm: 255.255.255.0
  ns:
    # dit is een ander stuk
    - 1.1.1.2
    - 1.1.1.3
...
```

Het is een goede gewoonte om in te springen per 2 spaties om de leesbaarheid te vergroten. Het eerste groepje met ip, nm en ns spingt 2 spaties in, het tweede groepje springt dan dus 4 spaties in ten opzichte van het begin van de regel. Dit groeperen van data heet nesting binnen YAML.

# Hoofdstuk 3

# Key/Value pairs

In veel configuratie bestanden gaat het voornamlijk om de toewijzing van een waarde aan een variabele zodat deze verder weer gebruikt kan worden. Zo ook in YAML:

```
---
ip: 192.168.1.1
...
```

In het Engels heet deze constuctie een key/value pair. Het ip gedeelte is de key, de waarde 192.168.1.1 is de value. Let op dat bij de toewijzing direct na de key een dubbele punt komt, daarna een spatie en dan de waarde al dan niet voorzien van quotes.

## 3.1 Een Key zonder waarde

Om een key een lege waarde te geven kunnen we twee constructies gebruiken:

```
---
leeg1: ~
leeg2: null
...
```

## 3.2 Strings

Het werken met strings kent verschillende manieren om deze op een zo leesbaar mogelijke manier weer te geven binnen YAML. Allereerst zijn er

de quotes om aan te geven of speciale characters weergegeven moeten worden als speciale characters of als letterlijke tekst.

De enkele quotes worden gebruikt om aan te geven dat iets letterlijk is:

```
---
tekst: 'deze tekst is letterlijk\n'
...
```

De backslash n zal dus letterlijk geschreven worden en zal niet tot gevolg hebben dat er een nieuwe regel gestart wordt. Dubbele quotes zorgen voor een nieuwe regel:

```
---
tekst: "deze tekst eindigt met een nieuwe regel\n"
...
```

We kunnen aan een key ook heel veel value geven. Er zijn twee manieren om dit toch leesbaar te houden:

```
---
tekst1: >
  dit is een string die
  meerdere regels omvat
  maar eigenlijk een lange
  string is.

tekst2: |
  dit is een string die
  daadwerkelijk uit meerdere
  regels bestaat.
...
```

Het groterdan en het pipe teken zorgen voor de verschillende manieren waarop deze YAML data wordt geïnterpreteert.

## 3.3 Nummers

```
---
integer: 42
float: 3.14159
...
```

## 3.4 Booleans

Binnen YAML kan een boolean value voor iets dat waar is op drie manieren worden gegeven: True, On of Yes. Voor iets dat niet waar is is de value dan False, Off of No. De case maakt niets uit dus true, True of TRUE is allemaal hetzelfde.

# Hoofdstuk 4

# Sequences

Een lijst met elementen heet in YAML een “sequence”. In andere talen wordt een sequence ook wel een list of een array genoemd.

Een sequence begint met een dash, min-teken, en daarna een spatie, waarna de waarde volgt.

Zo kunnen we bijvoorbeeld een sequence aanmaken met servers in ons netwerk:

```
---
servers:
  - srv01
  - srv02
  - srv03
...
```

Je mag een sequence ook in 1 regel beschrijven:

```
---
servers: [ "srv01", "srv02", "srv03" ]
...
```

# Hoofdstuk 5

# Maps

Een "Map" is niets ander dan een lijst van key/value-pairs. Maps worden ook wel dictionaries genoemd. Aan onze lijst van machines voegen we een dictionary toe met de IP-netwerk configuratie:

```
---
servers:
  - srv01:
    ip: 192.168.42.11
    subnetmask: 255.255.255.0
    def_gateway: 192.168.42.1
  - srv02:
    ip: 192.168.42.12
    subnetmask: 255.255.255.0
    def_gateway: 192.168.42.1
clients:
  - clt01:
    ip: 192.168.42.101
    subnetmask: 255.255.255.0
    def_gateway: 192.168.42.1
  - clt02:
    ip: 192.168.42.102
    subnetmask: 255.255.255.0
    def_gateway: 192.168.42.1
...
```

Aan het eind van de naam van de machine is nu een dubbele punt gekomen om aan te geven dat er meer informatie komt die bij de machine hoort. Die extra informatie is gegeven als een set mappings tesamen vormen ze de dictionary.

We kunnen lijsten en dictionaries eindeloos combineren:

```
---
servers:
  - srv01:
    ip: 192.168.42.11
    subnetmask: 255.255.255.0
    def_gateway: 192.168.42.1
    nameservers:
      - 192.168.42.2
      - 192.168.42.3
  - srv02:
    ip: 192.168.42.12
    subnetmask: 255.255.255.0
    def_gateway: 192.168.42.1
    nameservers:
      - 192.168.42.2
      - 192.168.42.3
clients:
  - clt01:
    ip: 192.168.42.101
    subnetmask: 255.255.255.0
    def_gateway: 192.168.42.1
    nameservers:
      - 192.168.42.2
      - 192.168.42.3
  - clt02:
    ip: 192.168.42.102
    subnetmask: 255.255.255.0
    def_gateway: 192.168.42.1
    nameservers:
      - 192.168.42.2
      - 192.168.42.3
...
```

Aan de lijst van machines hebben we in de dictionary voor elke machine een nameservers list toegevoegd. De lijst bevat de IP-adressen van de door ons gebruikt nameservers.