YAML이란?

YAML(YAML Ain't Markup Language)이란 JSON을 대체하고 있는 차세대 superset으로 docker 설치 등등에 많이 쓰이고 있습니다.

아래는 "Watchtower" Docker 컨테이너 설치를 위한 YAML 예제 입니다.

```
version: "3"
services:
  watchtower:
    image: containrrr/watchtower
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
    environment:
      TZ: Asia/Seoul
      WATCHTOWER_POLL_INTERVAL: 86400
    restart: unless-stopped
```

아래는 Redhat에서 설명하고 있는 내용입니다.

YAML은 사람이 읽을 수 있는 데이터 직렬화 언어로서, 구성 파일 작성에 자주 사용됩니다. YAML을 yet another markup language로 생각하는 사람도 있고, YAML ain't markup language(재귀 약어)로 생각하는 사람도 있습니다. 후자는 YAML이 문서가 아닌 데이터용임을 강조하는 말입니다.

YAML은 쉽게 읽고 이해할 수 있도록 설계되어 프로그래밍 언어 중에서도 인기가 높습니다. 또한 다른 프로그래밍 언어와 함께 사용할 수도 있습니다. YAML은 그 유연성과 접근성으로 인해 Ansible 자동화 툴이 Ansible Playbook 형태로 자동화 프로세스를 생성하는 데 사용합니다.

YAML 구문

YAML 파일은 .yml 또는 .yaml 확장자를 사용하며 특정 구문 규칙을 따릅니다.

YAML에는 Perl, C, XML, HTML, 기타 프로그래밍 언어에서 유래한 기능이 있습니다. 또한 YAML은 JSON의 상위 집합이므로 YAML에서 JSON 파일을 사용할 수 있습니다.

중괄호, 대괄호, 닫기 태그 또는 따옴표와 같은 통상적인 형식 기호는 없습니다. 또한 YAML 파일은 Python 스타일의 들여쓰기를 사용해 구조를 결정하고 중첩을 표시하므로 더 읽기 쉽습니다. 시스템 전반에서 이식성을 유지하기 위해 의도적으로 탭 문자를 허용하지 않으므로 문자 그대로의 공백 문자인 공백이 대신 사용됩니다.

코멘트는 파운드 또는 해시 기호(#)로 식별할 수 있습니다. 코멘트는 코드의 의도를 설명해 주므로 코멘트를 사용하는 것은 항상 권장됩니다. YAML은 여러 줄의 코멘트를 지원하지 않으므로 각 줄의 맨 앞에는 파운드 문자가 와야 합니다.

YAML 초심자가 흔히 하는 질문은 "3개의 대시는 무엇을 뜻하나요?"입니다. 3개의 대시(---)는 문서의 시작을 알리는 데 사용되며, 각 문서의 끝에는 3개의 점(...)이 사용됩니다.

YAML 파일의 매우 기본적인 예시는 다음과 같습니다.

```
#Comment: This is a supermarket list using YAML #Note that - character represents the list --- food: - vegetables: to
```

YAML 파일의 구조는 맵 또는 목록이며 들여쓰기와 키 값 정의 방식에 따라 계층 구조를 준수합니다. 맵 구조에서는 키-값 쌍을 연결할 수 있습니다. 각 키는 고유해야 하며 순서는 상관없습니다. Python 사전, 또는 Bash 스크립트의 변수 할당을 생각해 보세요.

YAML의 맵은 먼저 해결해야 닫을 수 있고, 그런 다음 새 맵이 생성됩니다. 들여쓰기 수준을 높이거나 이전 맵을 해결하고 인근 맵을 시작함으로써 새 맵을 생성할 수 있습니다.

목록의 값은 특정 순서로 나열되며, 목록 하나에 필요한 수의 항목을 포함할 수 있습니다. 목록 시퀀스는 대시(-) 및 공백으로 시작하며 들여쓰기로 상위 항목과 구분합니다. 여기서 시퀀스는 Python 목록 또는 Bash나 Perl의 어레이에 해당합니다. 목록을 맵에 포함할 수 있습니다.

위 예시에서 'vegetables' 및 'fruits'은 'food'이라는 이름을 가진 목록을 구성하는 항목들입니다.

또한 YAML은 스칼라를 포함할 수 있는데, 스칼라란 문자열, 정수, 날짜, 숫자 또는 부울 등의 값으로 사용할 수 있는 임의의 데이터(유니코드로 인코딩됨)를 말합니다.

YAML 파일을 생성할 때는 이러한 구문 규칙을 준수하고 파일이 유효한지 확인해야 합니다. 이를 위해 파일의 구문을 검증하는 애플리케이션인 린터(linter)를 사용할 수 있습니다. YAML 파일을 생성한 후 애플리케이션에 전달하기 전에 yamllint라는 커맨드로 유효성을 확인할 수 있습니다.

YAML 구문 예시

구문 규칙을 보여주는 학생 기록에 대한 단순한 YAML의 예시는 다음과 같습니다.

```
#Comment: Student record #Describes some characteristics and preferences --- name: Martin D'vloper #key-value
```

PyYAML 라이브러리를 사용해 이 파일을 Python으로 변환하면 다음과 같은 데이터 구조를 얻게 됩니다.

```
[ { "name": "Martin D'vloper", "age": 26, "hobbies": ["painting", "playing_music", "cooking"], "programming_langu
```

YAML은 어디에 사용될까요?

YAML의 가장 흔한 용도 중 하나는 구성 파일을 생성하는 것입니다. YAML과 JSON은 대부분의 경우 서로 바꿔서 사용할 수 있지만 YAML이 JSON보다 가독성이 높고 더 사용자 친화적이기 때문에 구성 파일은 JSON이 아닌 YAML로 작성하는 것이 좋습니다.

YAML은 Ansible 외에 쿠버네티스 리소스 및 배포에도 사용됩니다.

YAML의 장점은 변경 사항을 추적하고 감사하기 위해 GitHub와 같은 소스 제어에 YAML 파일을 추가할 수 있다는 것입니다.

Ansible과 Red Hat Ansible Automation Platform의 차이점은 무엇일까요?

Ansible에서 YAML 사용

Ansible Playbook은 IT 프로세스를 오케스트레이션하는 데 사용됩니다. 플레이북은 하나 이상의 플레이가 포함된 YAML 파일로, 시스템을 원하는 상태로 정의하는 데 사용됩니다.

각 플레이는 한 개 이상의 태스크를 실행할 수 있고, 각 태스크는 Ansible 모듈을 호출합니다. Ansible에서는 이 모듈을 사용하여 자동화 태스크를 완료합니다. Ruby, Python 또는 Bash 등 JSON을 반환할 수 있는 언어라면 무엇이든 Ansible 모듈을 작성할 수 있습니다.

Ansible Playbook은 맵과 목록으로 이루어져 있습니다. 플레이북을 만들려면 플레이의 이름을 지정하는 YAML 목록을 시작하고 태스크를 순서에 따라 나열하세요. 들여쓰기가 논리적 상속을 의미하는 것은 아닙니다. 각 줄이 YAML 데이터 유형(목록 또는 맵)이라고 생각하면 됩니다.

Ansible 사용자는 고급 프로그래밍 언어를 배우지 않고도 YAML 템플릿을 사용해 반복 태스크가 자동으로 수행되도록 프로그래밍할 수 있습니다.

쿠버네티스용 YAML

쿠버네티스는 정의된 상태와 실제 상태를 기반으로 작동합니다. 쿠버네티스 오브젝트는 클러스터의 상태를 나타내며, 사용자가 바라는 워크로드 상태를 쿠버네티스에 알리는 역할을 합니다. YAML 파일을 사용해 포드, 오브젝트, 배포와 같은 쿠버네티스 리소스를 생성할 수 있습니다.

쿠버네티스 오브젝트를 생성할 때는 원하는 오브젝트 상태를 정의하기 위한 사양을 포함해야 합니다. 쿠버네티스 API를 사용해 오브젝트를 생성할 수 있습니다. API에 대한 요청에는 오브젝트 사양이 JSON으로 포함됩니다. 하지만 대부분의 경우 필요한 정보를 kubectl에 YAML 파일로 제공하게 됩니다. Kubectl은 API 요청 시 이 파일을 YAML로 변환합니다.

일단 오브젝트가 생성 및 정의되고 나면 쿠버네티스가 작동하여 해당 오브젝트가 항상 존재하게 합니다.

개발자 또는 시스템 관리자는 YAML 또는 JSON 파일을 사용하여 정의된 상태를 지정하고 쿠버네티스 API에 제출합니다. 쿠버네티스는 컨트롤러를 사용하여 새롭게 정의된 상태와 클러스터의 실제 상태 간 차이점을 분석합니다.

버전 #2
★생성 10 7월 2023 02:34:31, DH Choi
✎수정 10 7월 2023 02:40:28, DH Choi