



# JSON 및 YAML 소개

# 학습 내용

## 강의의 핵심

배울 내용은 다음과 같습니다.

- JavaScript Object Notation(JSON) 또는 YAML Ain't Markup Language(YAML)의 기본 구문 설명하기
- JSON 및 YAML 데이터 형식의 장단점 설명하기

목표:

- JSON 및 YAML 형식 식별하기
- 두 형식의 기본 구문 이해하기



YAML 및 JSON: 클라우드 인프라 정의를 위한 구문

# 인프라의 언어

## JSON 및 YAML: 클라우드 인프라 생성을 위한 선언형 언어

- 코드형 인프라(IaC)는 텍스트 파일을 사용하여 클라우드에서 필요한 리소스를 선언하는 방식입니다.
- JavaScript Object Notation(JSON) 및 YAML Ain't Markup(YAML) 구문은 AWS에서 클라우드에 리소스를 선언하는 데 사용됩니다.
- 텍스트로 단순한 인프라에서 복잡한 인프라까지 정의할 수 있습니다.
- 코드형 인프라를 구축하려면 JSON 및 YAML의 구문을 이해해야 합니다.



# JSON 소개

# JSON의 정의

## JavaScript Object Notation(JSON)

- 다음을 포함하는 데이터를 저장하고 전송하기 위한 구문입니다.
  - 키-값 페어
    - » `{key1: "value1", key2: "value2"}`
  - 배열(목록) 및 기타 객체
    - » `{Array: [1, 2, 3]}`
- 단일 문서에서 복잡한 데이터 구조를 교환할 수 있습니다.

# 데이터 표현

StudentName	Location	FavoriteSport
María	USA	Tennis
John	UK	Cricket
Diego	USA	Basketball
Kwesi	India	Hockey

행과 열 형식의 데이터



```
{
  "sport_choices": [
    {
      "StudentName": "María",
      "Location": "USA",
      "FavoriteSport": "Tennis"
    },
    {
      "StudentName": "John",
      "Location": "UK",
      "FavoriteSport": "Cricket"
    },
    {
      "StudentName": "Diego",
      "Location": "USA",
      "FavoriteSport": "Basketball"
    },
    {
      "StudentName": "Kwesi",
      "Location": "India",
      "FavoriteSport": "Hockey"
    }
  ]
}
```

JSON 형식의 데이터

# JSON의 장단점

## 장점:

- 최소 구성(최소 구문 및 모형)—애플리케이션 프로그램 인터페이스(API)에 적합합니다.
- 사람이 읽고 쓰기 쉽습니다.
- 시스템이 구문 분석하고 생성하기 쉽습니다.

## 단점:

- 이진 데이터(예: 이미지 파일)에 대한 기본 지원이 없습니다.



# JSON 빌딩 블록: 객체

이 디저트를 어떻게 설명하시겠습니까?

케이크

초콜릿



8인분

\$20

# JSON 빌딩 블록: 객체

해당 정보를 JSON 객체로 구조화할 수 있습니다.



키      값

↓      ↓

```
{  
  "type": "cake",  
  "flavor": "chocolate",  
  "price": "20",  
  "feeds": "8"  
}
```

# JSON 빌딩 블록: 객체

- 중괄호({ })는 컨테이너 역할을 합니다.
- JSON은 범용 데이터 구조를 사용합니다.
  - 객체는 키-값 페어의 순서가 지정되지 않은 집합입니다.
  - 객체는 중괄호({ })로 시작하고 끝납니다.
  - 각 키 뒤에는 콜론(:)이 옵니다.
  - 각 키-값 페어는 쉼표(,)로 구분합니다.

```
{  
  "type": "cake",  
  "flavor": "chocolate",  
  "price": "20",  
  "feeds": "8"  
}
```

# JSON 빌딩 블록: 배열(1/3)

이 디저트를 설명하기 위해 어떤 추가 속성을 사용할 수 있습니까?

민트



블루베리

## JSON 빌딩 블록: 배열(2/3)

추가 재료 정보를 배열에 넣을 수 있습니다.



```
{  
  "type": "cake",  
  "flavor": "chocolate",  
  "price": "20",  
  "feeds": "8",  
  "additional_ingredients": [  
    "blueberries",  
    "mint"  
  ]  
}
```

# JSON 빌딩 블록: 배열(3/3)

- 대괄호([ ])는 배열을 포함합니다.
- 배열은 정렬된 값 목록입니다.
- 배열은 왼쪽 대괄호([)로 시작하여 오른쪽 대괄호(])로 끝납니다.
- 값은 쉼표(,)로 구분합니다.
- 값은 여러 유형으로 구성할 수 있습니다.
  - 문자열, 숫자, 객체, 배열 또는 부울

```
{  
  "type": "cake",  
  "flavor": "chocolate",  
  "price": "20",  
  "feeds": "8",  
  "additional_ingredients": [  
    "blueberries",  
    "mint"  
  ]  
}
```



# YAML 소개

# YAML이란 무엇입니까?

## YAML Ain't Markup Language(YAML)

- 사람이 읽을 수 있는 데이터 직렬화 언어입니다.
  - 구성 파일에 자주 사용됨
  - 데이터를 설명하는 데 사용됨
- 휴대용으로 설계되었습니다.
  - C, Java, Perl, Python, Ruby 및 기타 언어와 함께 사용할 수 있음
- JSON과 유사한 기능을 제공합니다.
  - 복잡한 데이터 구조를 단일 문서에서 교환 가능



# YAML VS JSON

## YAML의 장점

최적화된 가독성

JSON보다 덜  
장황함 -  
중괄호({}) 없음,  
따옴표 수가 더  
적음

포함된 주석 지원

많은 사람들이  
쉽게 디버깅할 수  
있음

## JSON의 장점

다른 컴퓨터  
시스템에서 더  
널리 사용됨

생성 및 구문  
분석이 쉬움

# YAML 구문 기본 사항

몇 가지 YAML 구문 기본 사항은 다음과 같습니다.

- 공백 들여쓰기를 사용하여 데이터 구조를 표시하고 생성합니다.
  - 탭을 사용하지 않습니다.
  - 2개의 공백이 표준 들여쓰기입니다.
- 줄 시작 부분에 숫자 기호(#)를 사용하여 주석을 추가합니다.
- 목록:
  - 하이픈으로 줄을 시작하여 목록을 만듭니다(줄당 1개의 목록 항목).
  - 또는 항목 사이에 대괄호([])와 쉼표(,)를 사용하여 한 줄에 목록을 정의합니다.
- 콜론 다음에 공백을 사용하여 연관 배열을 만듭니다.
  - <key>: <value>(줄당 1개의 항목)
- 문자열은 일반적으로 따옴표로 묶이지 않습니다.
  - 그러나 큰따옴표나 작은따옴표로 묶을 수 있습니다.

# 예제: JSON 및 YAML

## JSON\*

```
"webServer": {
  "Type": "AWS::EC2::Instance",
  "Properties": {
    "ImageId": "ami-09ead922c1dad67e4",
    "InstanceType": "t2.micro",
    "KeyName": "myKey",
    "SecurityGroupIds": [
      {
        "Ref": "SecurityGroupIDs"
      }
    ],
    "SubnetId": "subnet-42b01138"
  }
}
```

\*실제 코드 아님

## YAML\*

```
webServer:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: ami-09ead922c1dad67e4
    InstanceType: t2.micro
    KeyName: myKey
    SecurityGroupIds: !Ref SecurityGroupID
    SubnetId: subnet-42b01138
```

\*실제 코드 아님

# 핵심 사항



- JSON
  - 데이터를 저장하고 전송하기 위한 구문입니다.
  - 텍스트 기반 형식이므로 사람이 읽을 수 있습니다.
  - 문서를 쉽게 작성할 수 있습니다.
  - 키-값 페어 및 데이터의 배열을 저장합니다.
- YAML
  - 데이터를 저장하기 위한 구문입니다.
  - 텍스트 기반 형식이므로 사람이 읽을 수 있습니다.
  - 문서를 쉽게 작성할 수 있습니다.
  - 키-값 페어, 목록 및 데이터의 연관 배열을 저장합니다.
  - 단일 YAML 문서에 복잡한 데이터 구조를 저장합니다.