JSON vs YAML

A dive into two popular data serialization languages

Dilan Sachintha Nayanajith Follow

Jun 16, 2020 · 5 min read ★

Photo by vipul uthaiah on Unsplash

Any person with an interest in programming and technology knows what **JSON** is. **YAML** is not as common as JSON, but it is also a popular and awesome data serialization language. For example, any person who has used docker surely knows what YAML is.

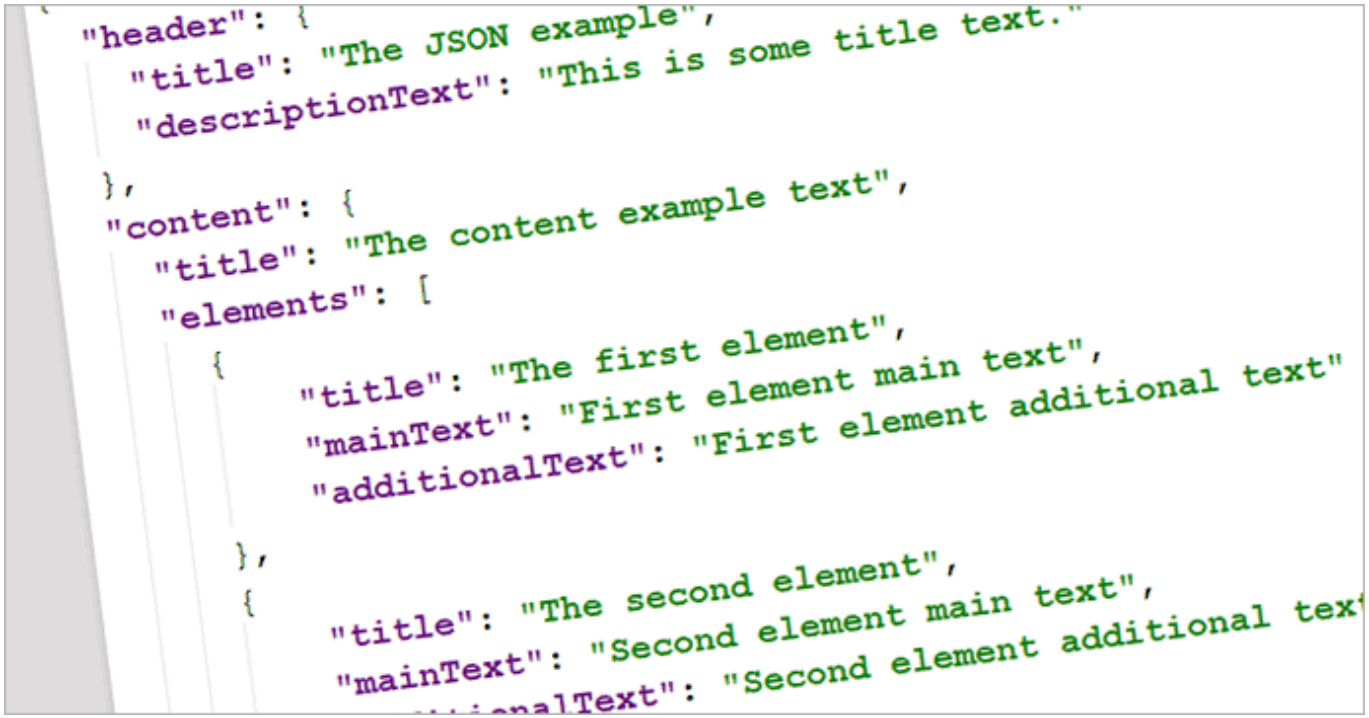
First of all, let's see what is meant by data serialization. According to Devopedia data serialization is the process of converting data objects present in complex data structures into a byte stream for storage, transfer, and distribution purposes on physical devices. So JSON and YAML both are a way of storing the data objects and structures in files.

. . .

What is JSON?

Although most of us know what JSON is, let's have a quick introduction. **JSON** is the short name for **JavaScript Object Notation.** JSON is based on a subset of JavaScript programming language standard ECMA-262 3rd Edition-December 1999. JSON is widely used with JavaScript but since it is language-independent, it can be used with any programming language.

JSON has a standard format for data storing. It stores data in key/value pairs. The records are separated by commas and both field names and strings are enclosed with double-quotes.



JSON example — Image from Supertext

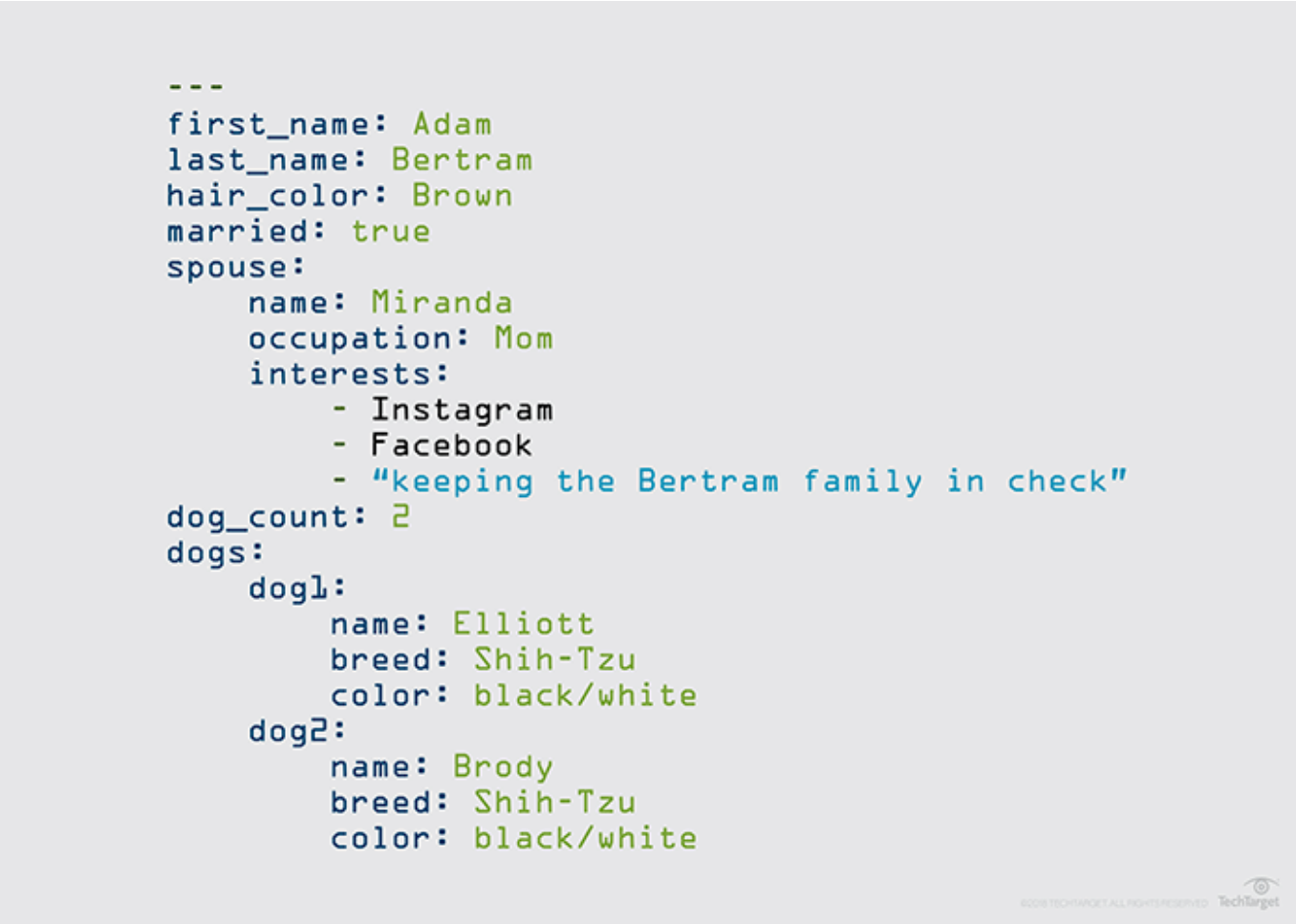
Here, *"title"* is a key, and *"The first element"* is a value. "elements" key contains an array of element objects and the *"content"* key contains a data object.

. . .

What is YAML?

YAML is the short name for **YAML Ain't Markup Language.** And **YAML Ain't Markup Language** is the short name for **YAML ain't markup language ain't markup language.** Pretty cool right. To make things more interesting, the official YAML website is also displayed in YAML format.

YAML uses three dashes ("`---`") to indicate the start of a document and three dots ("`...`") to indicate the end of a document. Unlike JSON, YAML uses the indentations just like in Python to show the levels in the data. The key/value pairs are separated with a colon and the lists begin with a hyphen in YAML. And also YAML files are written with the extension YML in some places and both `.YAML` and `.YML` means the same file type.



```yaml
---
first_name: Adam
last_name: Bertram
hair_color: Brown
married: true
spouse:
    name: Miranda
    occupation: Mom
    interests:
        - Instagram
        - Facebook
        - "keeping the Bertram family in check"
dog_count: 2
dogs:
    dog1:
        name: Elliott
        breed: Shih-Tzu
        color: black/white
    dog2:
        name: Brody
        breed: Shih-Tzu
        color: black/white
```

YAML example — Image from TechTarget

Here, *first_name* is a key and *Adam* is a value. *spouse* points to a data object with *name*, *occupation*, *interests* as children and *interests* is an array of data.

. . .

Pros & Cons — JSON vs YAML

Theoretically, both JSON and YAML are expected to perform the same task. That is providing a human-readable data interchange format.

YAML

```yaml
simple-property: a simple value

object-property:
  a-property: a value
  another-property: another value


array-property:
  - item-1-property-1: one
    item-1-property-2: 2
  - item-2-property-1: three
    item-2-property-2: 4

# no comment in JSON
```

JSON

```json
{
  "simple-property": "a simple value",

  "object-property": {
    "a-property": "a value",
    "another-property": "another value"
  },

  "array-of-objects": [
    { "item-1-property-1": "one",
      "item-1-property-2": 2 },
    { "item-2-property-1": "three",
      "item-2-property-2": 4 }
  ]
}
```

Same content in both formats — Image from Manning

Readability & Complexity

The design goal of JSON is to be simpler as possible and be universally usable. So this has reduced the readability of the data up to some extent. In contrast, the design goal of YAML is to provide a good human-readable format and provide support for serializing arbitrary native data structures. So this has increased the readability of the YAML files but it has made the parsing and generation of files somewhat complex. We can see this clearly

in the YAML official website where it shows the content in YAML format and it is easily readable for anyone visiting the site. On the other hand, if it was displayed in JSON format, the webpage would be useless.

It is said that YAML is a superset of JSON format. What is simply meant by this is that we can parse JSON using a YAML parser. However, in practical scenarios, this parsing may rise problems, but it is theoretically possible.

Serialization Performance

In the data serialization contest, JSON is the winner because of the ability to quickly and easily parse JSON serialized data with its simpler design. And this has made JSON more popular among developers which results in more and more native support and this has improved the performance more again. Therefore JSON has become the most widely used data interchange format for web applications and web services.

Community Support

For any programming language, we can easily find a JSON library integrated with the language due to its popularity, it's ease use of implementation and it's simplicity. The JSON official website lists many languages with many supporting libraries for JSON. YAML also has

widespread support and many libraries for integrating it with various languages, but not as much as JSON. You can get the list of libraries and languages that support YAML here.

Comments

So far we discussed the pros of JSON over YAML. But some considerably important features of YAML is not found with JSON. YAML supports comments where JSON does not. We can comment anywhere in the document with a simple # character. This has proven advantageous when writing configuration files where one developer can easily describe the configuration using the comments. Therefore YAML format is used in many technology stacks like ElasticSearch, Docker for storing configuration information.

Ability to use Complex Structures

Another feature that YAML provides is the ability to reference other data objects. With this referencing, it is possible to write recursive data in the YAML file.

```
    - variables:
        - name: CLUSTER_NAME
    - step: &snapshotDatabases
        name: Snapshot Databases
        image: *image
        script:
          - ./scripts/initStep.sh
          - ./scripts/dbSnapshot.sh
  restoreDB:
    - variables:
        - name: CLUSTER_NAME
        - name: NEXT_CLUSTER_NAME
    - step: &restoreDB
        name: restore the DB instances from snapshots
        image: &image nebulagarage/deploy:1.1.3
        script:
          - ./scripts/initStep.sh
          - ./scripts/restoreDB.sh
  deployStagingNew:
    - variables:
        - name: CLUSTER_NAME
        - name: NEXT_CLUSTER_NAME
    - step:
        name: deploy to staging
        image: &image nebulagarage/deploy:latest
        script:
          - echo "hello world"
    - step: *snapshotDatabases
    - step: *restoreDB
```

Referencing other objects — Image from Atlassian Community

For this, we can define **Anchors** in the YAML file using “&” and refer to them later using **Aliases**, “*”. This is a very important feature in YAML that JSON does not offer. In JSON, it is impossible to serialize complex structures with object references. But the above feature in YAML solves that

problem. But one drawback of this is the possibility of infinite looping in some converters.

So from all these points, we can see that both JSON and YAML have their strengths and their weaknesses. A good developer should be able to identify these and use the correct format in the correct place.

. . .

I hope you got something from my article. Hope to bring more articles soon. And thank you for reading. Cheers.

. . .

Resources

- *Official YAML website*
- *Official JSON website*
- *Data serialization from Devopedia*

Json Yaml Programming Web Development Technology