* YAML is a light-weight, human-readable data-serialization language. It is primarily designed to make

the format easy to read while including advanced features.

* It is very easy and simple for represent complex mapping. Due to which it is heavily used in

configuration settings.

* It is similar to XML and JSON files but uses a more minimalist syntax even while maintaining similar

capabilities.

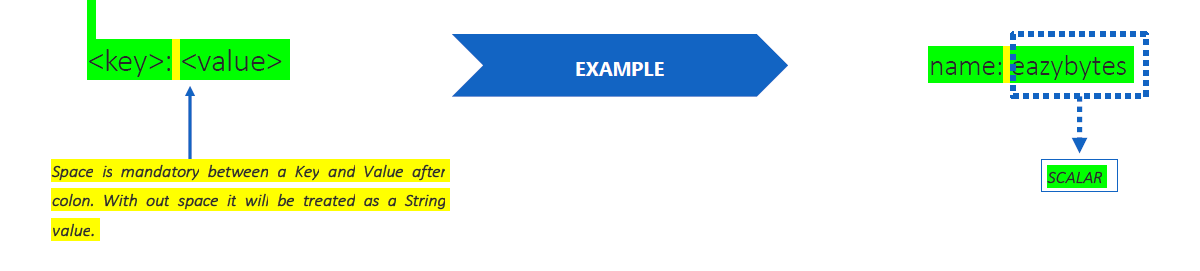
* YAML files are created with extensions “.yaml” or “.yml” . You can use any IDE or text editor to open/create YAML files.
* YAML is similar inline style to JSON (is a superset of JSON).



**KEY VALUE PAIRS**

Most things in a YAML file are a form of key value pair Key value pairs are the basis for all other YAML

constructions

****

**Data Types in YAML**

YAML has three types of data types:

* Scalar
* List
* Dictionary

**Note**:

In yaml data are stored In form of key value pair. (**key: <value>**). Data type is decided from <value> not from key

In order to provide comment use # (hashtag symbol) at the start of the line

**Comment in yaml file**

# This is an example of

# a block comment

# in a YAML file

**Scalars**

Scalars are the simplest data type in YAML and can represent basic types, including boolean, integers, and floating-point numbers. The following basic data types are supported by YAML:

* Boolean --- (true and false) or (yes or no)
* Integer -- any number
* Float --- number with decimal point or exponential ( example -- 180.0  , 12.3015e+05   )
* String --- alphanumeric character, with or without ' or " ( name: rakesh , name: 'rakesh ', age: ‘10’)
* Date
* Timestamp



**Note:**

Octal numbers start with 0 and hexadecimal number with 0x like python

**String**

* YAML strings are Unicode. In the following example, we are going to define a simple string, without using quotes. Strings in YAML doesn’t need explicit double or single quotes.
* Use single or double quotes in YAML if your value includes special characters. For example, these special characters may require quotes. {,},[,],,,&,:,\*,#,?,|.-,<.>,=,!,%,@,\.
* "Yes" and "No" should be enclosed in quotes (single or double) or else they will be interpreted as True and False boolean values.

**Note**

YAML can autodetect types. However, it is often necessary to explicitly specify the type using a tag.

To force a type, we can prefix the type with a !! symbol. Here's an example:

age: !!float 23

married: !!str true

binary: !!int 0b101010

hexadecimal: !!int 0x1C7A

name: !!str "James"

**Common tags supported in YAML**

|  |  |
| --- | --- |
| **Tag** | **Description** |
| !!bool | Denotes a boolean value |
| !!int | Denotes a integer value |
| !!float | Denotes a floating point number |
| !!str | Denotes a string |

**List Datatype**

In yaml we can create list using two day-

1. Using python like syntax. Example-

items: [6, 7, 8, 9, 10]

name: [six, seven, eight, nine, ten]

1. Using block level syntax. Example-

items:

  - 6

  - 7

  - 8

name:

  - "six"

  - "seven"

  - "eight"

  - "nine"

**Dictionary**

If we want to write a complex YAML file which holds the complex data structure, we will use dictionaries.

It is a collection of key: value pairs and each of the key: value pairs can be nested with a lot of options.

Example

    student2:

      fatherName: "William"

      motherName: "Marry"

      subjectDetails:

        subject1: 70

        subject2: 100

Explanation:

student2 is key which contains the other key(fathername, mothername, subjectdetail) and value (William, Marry, subject1: 70 and subject2: 80 )

subject1: 70 and subject2: 80 are another key value pair for key - subjectdetails

**Example of json to yaml**

