YAML Introduction

What is YAML?

During it's initial development, YAML was reported to stand for "**Y**et **A**nother **Markup** **L**anguage". However, once it was completed as a standard, that acronym was morphed into "**Y**AML **A**in't **M**arkup **L**anguage".

In short, YAML is meant to be a "human readable data serialization format". In other words, it's meant for non-computers (us) to be able to easily read and recognize the signifance of its content without complex translation applications.

It was designed to be easy to map to high level languages and you will often see it compared to key/value lists, associative arrays and data outlines.

Structure

As a data structure, YAML most closely resembles an outline or list of things with basic descriptions.

For example, if we wanted to list our favorite movies in a way that YAML processing engines would be able to recognize, our YAML file would contain something like:

--- # Our Favorite Movies of All Time
- The Terminator
- Star Trek
- Star Wars

YAML EXAMPLES

YAML Characteristics

In general, YAML has the following characteristics (with samples of each):

Simple Lists:
--- # Grocery List
- Bananas
- Apples
- Oranges
- Cereal
- Eggs

Inline formatting like:
[bananas, apples, oranges, cereal, eggs]

Associative Arrays/Key Value Pairs

--- # Employee Information (List)
Name: John Smith
Age: 44
HireDate: 09/01/2011

--- # Employee Information (Inline)
{Name: John Smith, Age: 44, HireDate: 09/01/2011}

NOTE: Strings do not require quotation and new lines are preserved in order to function as a delimiter for the field

Ansible doesn't really use these too much, but you can also specify a boolean value (true/false) in several forms:

```
create_key: yes
needs_agent: no
knows_oop: True
likes_emacs: TRUE
uses_cvs: false
```

Values can span multiple lines using | or > . Spanning multiple lines using a | will include the newlines. Using a > will ignore newlines; it's used to make what would otherwise be a very long line easier to read and edit. In either case the indentation will be ignored. Examples are:

```
include_newlines: |
        exactly as you see
        will appear these three
        lines of poetry

ignore_newlines: >
        this is really a
        single line of text
        despite appearances
```