Ansible Templating

- Intro
- Jinja2
  - Filters
  - Control Structure
  - Lookups (in the context of templating)
  - Documentation

Goal: Refresher with some examples.
Simple but very **powerful** technique found in almost **every** Ansible role you use or build.

My Ansible journey:

- Started to experiment in 2014 and loved it "Simplicity as key design philosophy"
- 2018 Rabobank. Automated the installation and configuration of the BAE System NetReveal Product. YAML manifest, Jenkins, Ansible and Tower
- 2020 ABN-AMRO CISO CTI Platform. Built, configured and integrated 'theHive', MISP and Cortex into containers to run within EKS

We are all learning together here :-)

Ansible uses the Jinja2 template engine

Here's "Hello World" using the jinja2 python lib

```
from jinja2 import Template
template = Template("Hello {{ something }}!")
print(template.render(something="World"))
```

Ansible Jinja2 template - Hello World example

- We need:
  - a variable
  - a single template file - <filename>.j2
  - a single debug task using template lookup plugin

This tiny playbook utilizes a Jinja2 template file, a filter and uses of a lookup plugin.

The Hello World playbook

```
---
- name: Hello World
  hosts: all
  vars:
    something: World

  tasks:
    - name: Show templating results
      ansible.builtin.debug:
        msg: "{{ lookup('template', './some_template.j2') }}"
```

template file

```
Hello {{ something }}
```

Jinja2 syntax

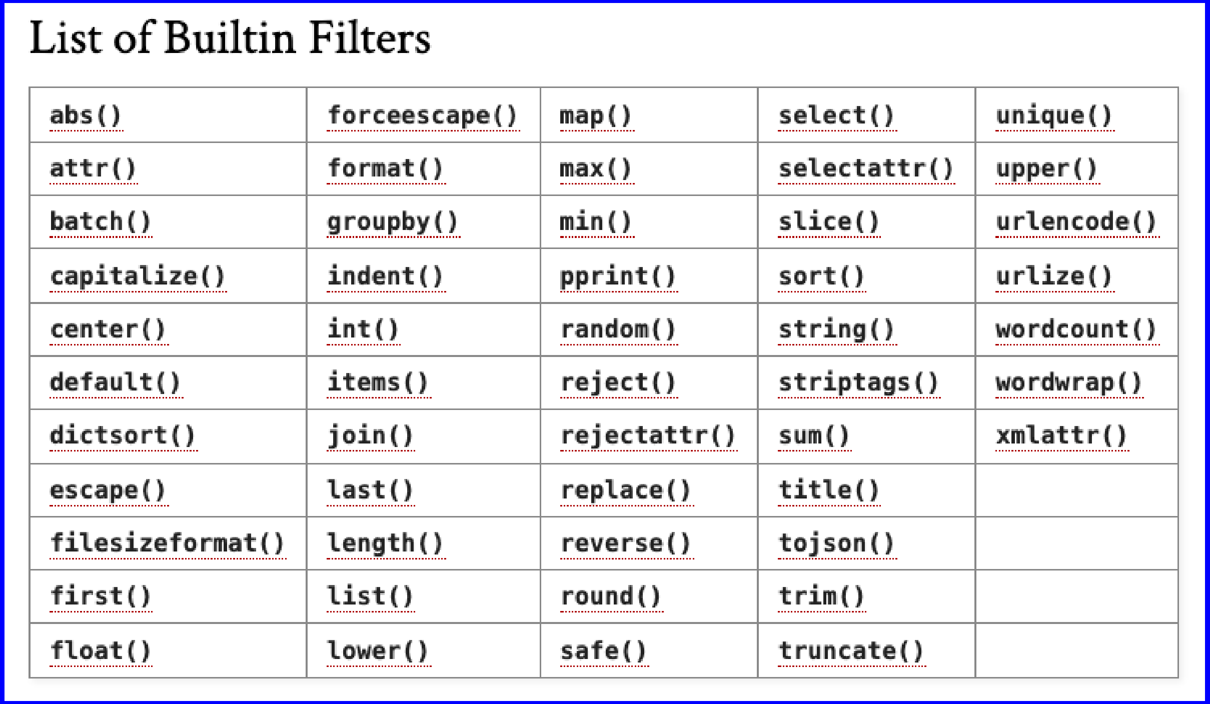
{% ... %} for **Statements** i.e. control structures

{{ ... }} for **Expressions** to **print** to the template output

{# ... #} for **comments** not included in the template output

Filters in print expressions {{ | }}

```
{{ something | default('world') }}
{{ my_list | sort | join(',') }}
{{ '192.0.2.0' | ipaddr }}
{{ myvar | ipv4 }}
```



List of Builtin Filters

abs()	forceescape()	map()	select()	unique()
attr()	format()	max()	selectattr()	upper()
batch()	groupby()	min()	slice()	urlencode()
capitalize()	indent()	pprint()	sort()	urlize()
center()	int()	random()	string()	wordcount()
default()	items()	reject()	striptags()	wordwrap()
dictsort()	join()	rejectattr()	sum()	xmlattr()
escape()	last()	replace()	title()	
filesizeformat()	length()	reverse()	tojson()	
first()	list()	round()	trim()	
float()	lower()	safe()	truncate()	

Statements {% ... %} (control structures)

For, If, Macros, Call, Filter

for ... in

```
{% for user in users %}
  {{ user.username }}
{% endfor %}
```

```
{% for key, value in my_dict.items() %}
    {{ key }}
    {{ value }}
{% endfor %}
```

if … in

```
{% if 'priority' in data %}
    <p>Priority: {{ data['priority'] }}</p>
{% endif %}
```

if … is

```
{% if variable is defined %}
    value of variable: {{ variable }}
{% else %}
    variable is not defined
{% endif %}
```

Tip: watch out to manage whitespace around control structures {%-, {%+, -%}, +%} as required

Lookups

- Lookup plugins retrieve data from outside sources (files, databases, key/value stores, APIs).
- Lookups execute and are evaluated on the Ansible control machine (Plugin vs Module).
- Data returned by a lookup plugin is available using the standard templating system.
- Lookups can be explicitly part of Jinja2 expressions fed into the loop keyword.

Simple lookup

```
- name: show templating results
  debug:
    msg: "{{ lookup('template', './some_template.j2') }}"
```

variable from lookup

```
vars:
  motd_value: "{{ lookup('file', '/etc/motd') }}"
tasks:
  - debug:
      msg: "motd value is {{ motd_value }}"
```

Dig lookup in a loop

```
- name: use in a loop
  ansible.builtin.debug:
    msg: "MX record for gmail.com {{ item }}"
  with_items: "{{ lookup('community.general.dig', 'gmail.com./MX', wantlist=True) }}"
```

URL lookup in a loop

```
- name: url lookup splits lines by default
  ansible.builtin.debug:
    msg="{{item}}"
  loop: "{{ lookup('ansible.builtin.url', 'https://github.com/gremlin.keys', wantlist=True) }}"
```

Documentation

Start here:

https://docs.ansible.com/ansible/latest/user_guide/playbooks_templating.html

https://jinja.palletsprojects.com/en/3.1.x/templates/#id11

https://docs.ansible.com/ansible/latest/collections/ansible/builtin/template_module.html

Task

Create a simple playbook or role to template the sshd_config on one of your sandbox systems based on the Ansible builtin template module below

```
- name: Update sshd configuration safely, avoid locking yourself out
  ansible.builtin.template:
    src: etc/ssh/sshd_config.j2
    dest: /etc/ssh/sshd_config
    owner: root
    group: root
    mode: '0600'
    validate: /usr/sbin/sshd -t -f %s
    backup: yes
```