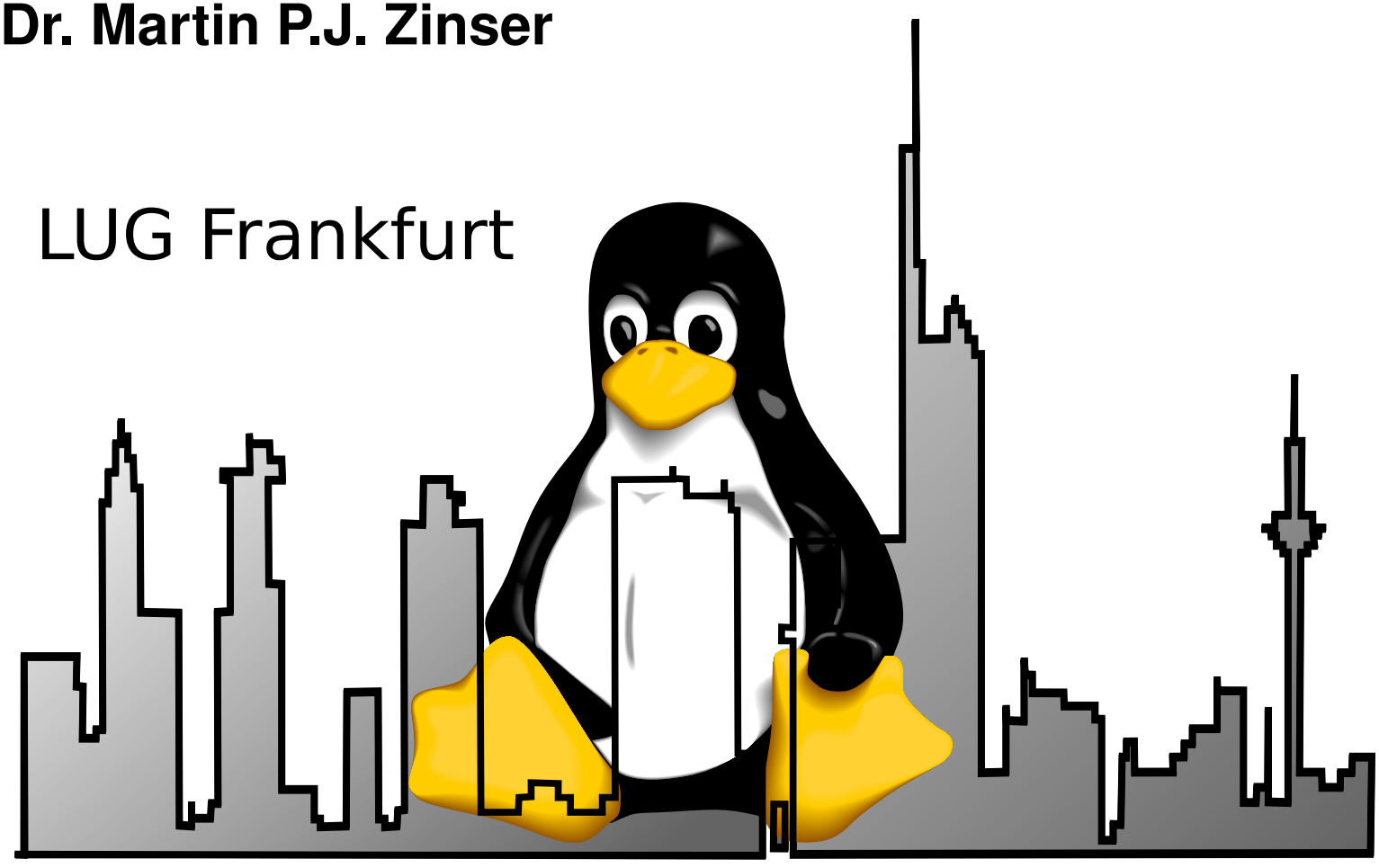
Ansible

Introduction to Ansible

Dr. Martin P.J. Zinser

LUG Frankfurt



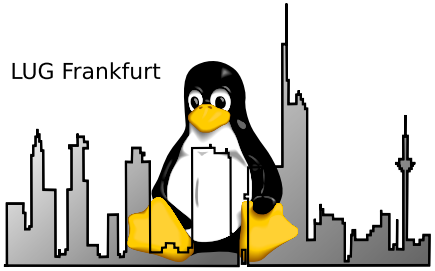


Table of contents I

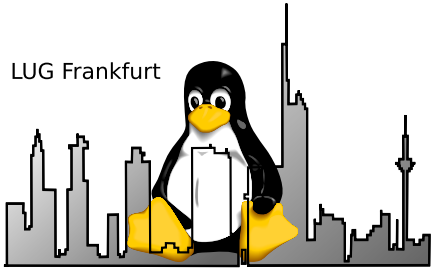
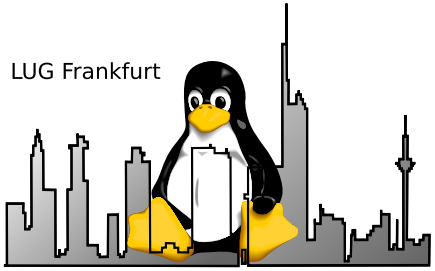
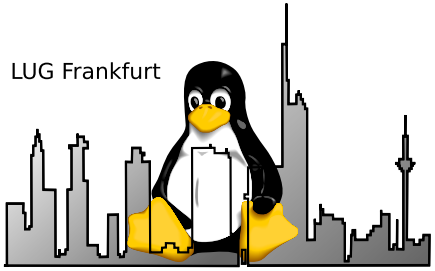


Table of Contents



Motivation

- Discussion on the LUG Mailing List how to reproduce SW inventory of one system on a new one
- Suggestion: This is easy, if you use ansible to manage your SW...
- ... but no how-to



Goal

```
---
- name: Add additional Repos and SW to linux hosts
  hosts:
    - pluto
  gather_facts: true

  tasks:
    - name: Add/update local SW for maker tools
      ansible.builtin.apt:
        name:
          - arduino
          - bobdude
          - thonny
        state: latest
        install_recommends: true
```

At the end of the presentation you should understand what is going on here and be able to setup and use ansible for simple automation tasks.
Note: Many (even semi-)advanced topics and features not covered due to time constraints.

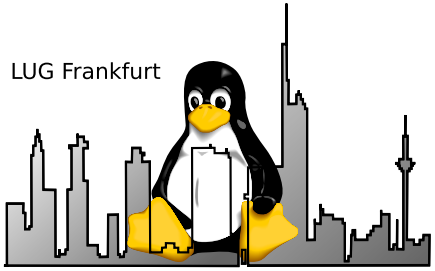
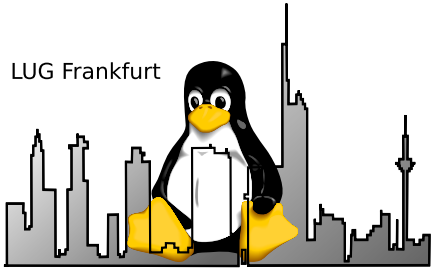


Table of Contents

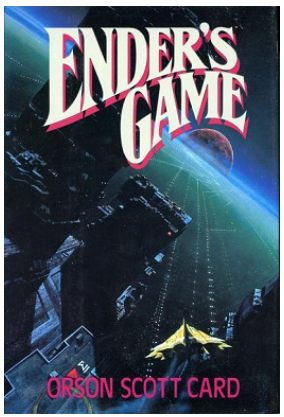


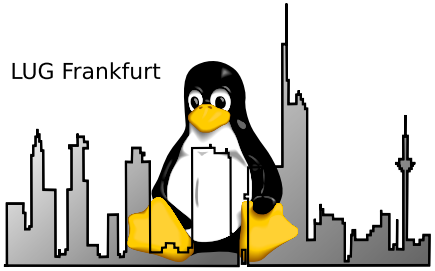
What is Ansible?

An Ansible is a category of fictional devices or technology capable of near-instantaneous or faster-than-light communication.

Introduced as a contraction of *answerable* by Ursula K. Le Guin in the Hainish Cycle.

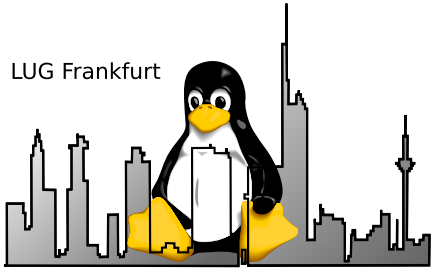
Also used by Orson Scott Card in the Enderverse.





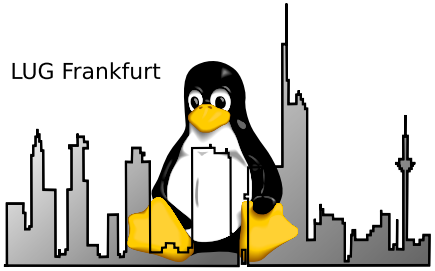
What is Ansible the SW?

- Infrastructure as code
- SW provisioning, configuration management, application deployment
- Agentless
- **Declarative**
- Unixoid OSes (including MacOS) and Windows



Target features

- Small SW footprint (SSH and Python)
- Consistent results across deployments
- Small attack surface (agentless)
- **Idempotency**
- Shallow learning curve
- Simple, text based ($\rightarrow$ Version Control/DevOps)



What else?

- Other IaC tools: Chef, Puppet/OpenVox, Salt, CFEngine
- Cloud IaC: Terraform/OpenTofu

(My) Ansible Goldilocks Zone

- $\leq$ 5 nodes: Manual
- 6 $\leq$ nodes $\leq$ 100: Ansible
- nodes $\geq$ 100: Centralized config management

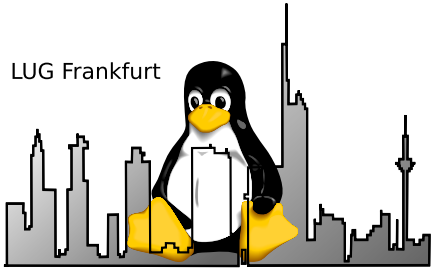
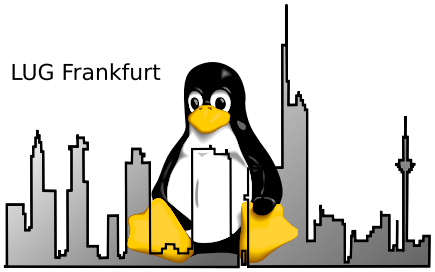


Table of Contents



Ansible lingo

Control node System with ansible installed. Used to manage other nodes[1]

Managed node System under (partial) management of ansible

Inventory List of managed nodes, potentially grouped. Either static or dynamic

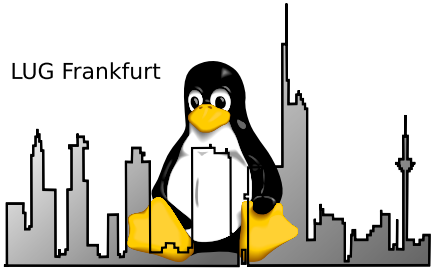
Module Program that implements a particular activity

Playbook A file with one or more plays is a playbook

Play Description of the desired state of the managed nodes

Task Automation activity using a module to ensure a particular state

[1] Node = physical or virtual server



Ansible installation

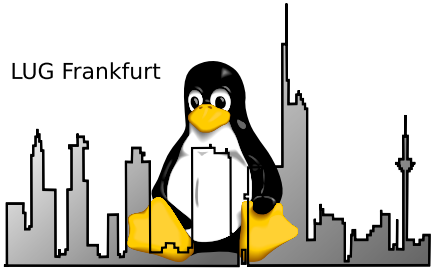
Control node

- ansible (ansible-core will be pulled in by the package manager)
- Optional: ansible-lint to check playbooks for conformance

Managed node

- Standard python installation sufficient for most use cases.
- Special modules might require additional python libraries (e.g. ldap)
- Note: Ansible can not be used for the initial installation of a physical server. It is possible to create VMs (on-prem and cloud) via a playbook[2]

[2] I have not tested this yet



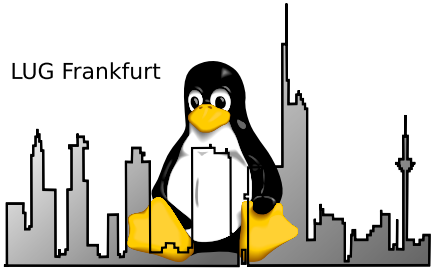
Digression: YAML - YAML Ain‘t a Markup Language

Definition

YAML is a human readable data serialization language

- - -	Starts a document
key: value	Creates a mapping of key to value (also {k1: v1, k2: v2,...})
- value	Element in a sequence (also name: [v1, v2,...])
#	Starts a comment (also inline)
strings	Can be bare[3], ’quoted’, "double quoted"

[3] Try to avoid bare strings, as this can lead to collisions with keywords



Digression: YAML - YAML Ain‘t a Markup Language (cont.)

```
- name: Create plugin diretory for arduino2 IDE
  ansible.builtin.file:
    path: "/home/{{ item }}/.arduinoIDE/plugins"
    state: directory
    owner: "{{ item }}"
    group: users
    mode: ’0755’
  with_items:
    - szinser
    - mzinser
```

Nested Mapping - ansible.builtin.file:

Sequence - Contents of *with_items*

Note: Indentation is significant in YAML and denotes a parent/child dependency.

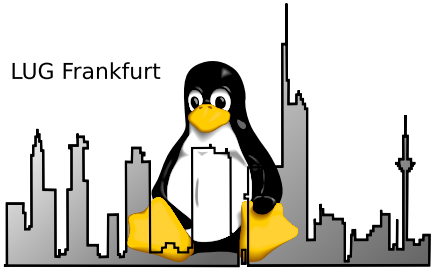
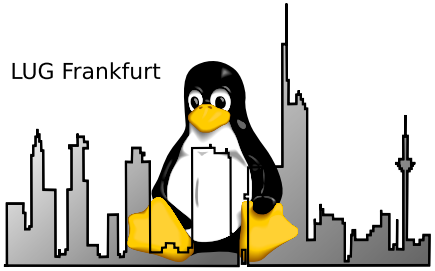


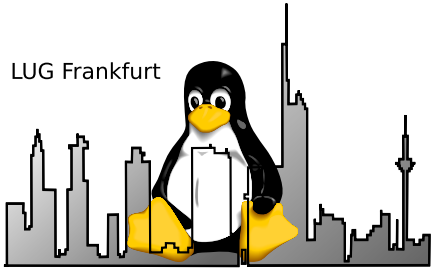
Table of Contents



Ansible inventory

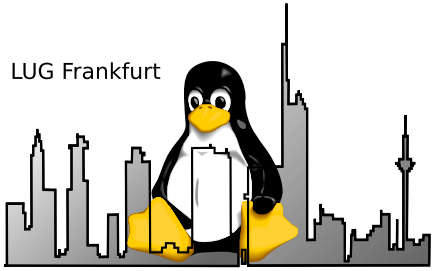
- List of nodes that ansible can manage kept in an INI file[4]
- One line per host name or IP address
- Group into [host-groups]
- Build host group from sub groups [host-group:children]
- Implicit host groups all and ungrouped
- Use ranges to keep lists compact
- If *inventory is a directory, all files in dir are parsed/executed*

[4] Also YAML and dynamic generated possible



Ansible inventory - Example

```
voyager                                 <--- Simple entry
newton ansible_host=10.0.0.27           <--- Defining host variable
10.128.0.[1:9]                          <--- Using Range
[Kronau]                                <--- Host Group
bach
newton
[Dietzenbach]
hal
crest
[Zinser:children]                       <--- Nested Host Group
Kronau
Dietzenbach
```



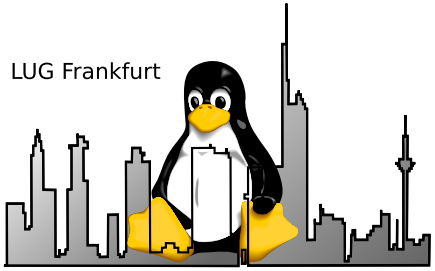
Ansible inventory - Location and test

Location:

- Systemwide in /etc/ansible/hosts
- Override in ansible configuration file
- Explicit when invoking ansible with –inventory/-i Path

Verify:

```
ansible <testvalue> --list-hosts (--inventory Path)
ansible Kronau --list-hosts
hosts (2):
  bach
  newton
```



Ansible configuration

Search List: $ANSIBLE_CONFIG, ./ansible.cfg, ~/.ansible.cfg, /etc/ansible/ansible.cfg

Only the first one found is used, values are not merged.

Check with ansible(-playbook) -v

```
[defaults]
inventory = inventory                    <- Inventory to use
remote_tmp = /tmp/ansible-${USER}/tmp    <- Make it easier to find
gathering = explicit                     <- Facts on demand (time!)
callback_result_format = yaml            <- Human readable (vs. json)
[privilege_escalation]
become = true                            <- Switch user on remote
become_user = root                       <- Which user on remote
become_method = sudo                     <- How to switch
```

Note: For distros without root user (e.g. Ubuntu), become_user should be the local admin user.

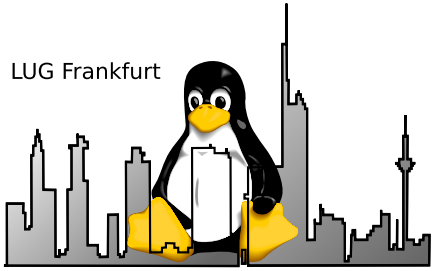
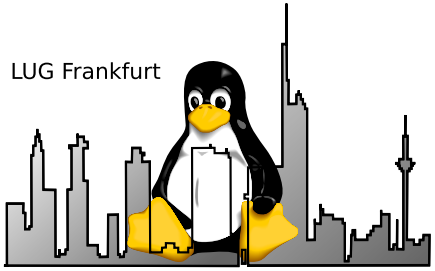


Table of Contents



Ad hoc commands

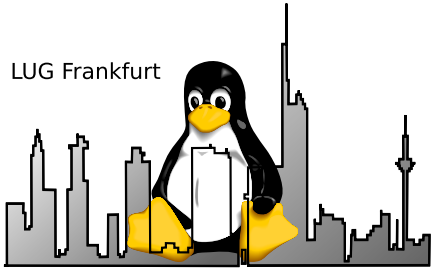
ansible host(s) -m module [-a 'module params'] [-i inventory]

run a (simple) task on managed nodes without crafting a playbook.

Examples:

- ansible.builtin.ping - Check (ansible) connectivity (Not ICMP)
- ansible.builtin.copy - Copy file from control node to managed nodes
- ansible.builtin.systemd - Control systemd services
- ansible.builtin.command/shell - Execute a command on managed nodes

My current (default) installation of ansible on Debian contains 10460 modules.



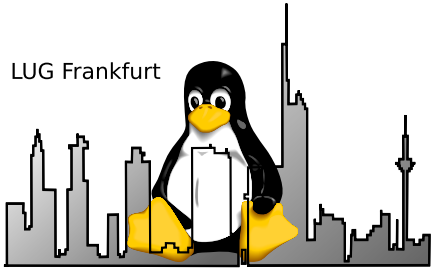
Ad hoc commands (cont.)

ansible localhost -m setup

run during [Gather facts] and may be used as variables in templates and conditions (> 1600 lines on my system)[5]

```
localhost | SUCCESS => {
    "ansible_facts": {
        "ansible_all_ipv4_addresses": [
            "10.0.0.13"
        ],
        "ansible_all_ipv6_addresses": [
            "fe80::6e0b:84ff:fe0b:f9a8"
        ],
        "ansible_apparmor": {
            "status": "enabled"
        },
        "ansible_architecture": "x86_64",
        "ansible_bios_date": "10/22/2014",
        "ansible_bios_vendor": "LENOVO",
        "ansible_bios_version": "FBKTA1AUS",
        "ansible_board_asset_tag": "NA",
        "ansible_board_name": "SHARKBAY",
...
```

[5] Depends on local SW installed (e.g. facter included)



Module documentation

Show all modules on control node

```
$ ansible-doc -l
amazon.aws.autoscaling_group         Create or delete AWS AutoScaling Groups (ASGs)
amazon.aws.autoscaling_group_info    Gather information about EC2 Auto Scaling Groups (ASGs) in AWS
...
```

Detailed documentation:

```
$ ansible-doc ansible.builtin.copy
> MODULE ansible.builtin.copy (/usr/lib/python3.11/site-packages/ansible/modules/copy.py)

  The ansible.builtin.copy module copies a file or a directory
  structure from the local or remote machine to a location on
  ...
OPTIONS (red indicates it is required):

   attributes  The attributes the resulting filesystem object should have.
   ...
```

ansible-doc -s module terse *playbook* like list of all module options

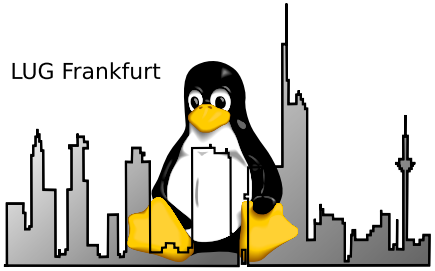
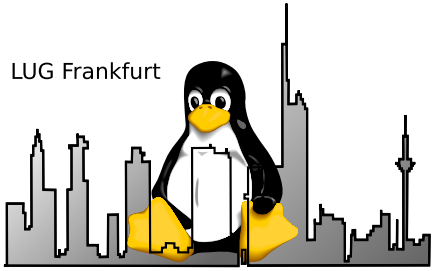


Table of Contents



Ansible playbooks

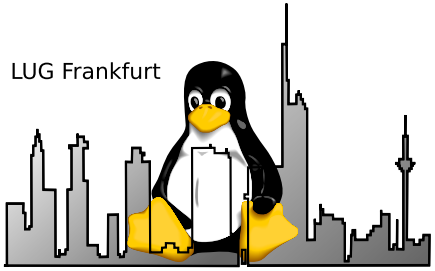
Play - Ordered set of tasks run against managed node

Playbook - Text file with a list of one or more plays to run in order

```
---
- name: Setup environment for Calliope mini
  hosts: saturn

  tasks:
    - name: Ensure mc-user group exists
      ansible.builtin.group:
        name: mc-user
        state: present

    - name: Setup Calliope mini user 1 (calliuser1)
      ansible.builtin.user:
        name: calliuser1
        group: mc-user
        shell: /bin/bash
        password: $6$mysecretsalt$9LuQ2h0...
        update_password: on_create
        state: present
      register: c1_data
```



Ansible playbooks (cont.)

`- - -` - Begin of document marker (YAML)

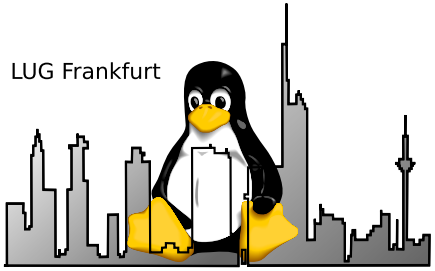
`name` - Name of the following play

`hosts` - Managed nodes to target

`tasks` - List of actions to be performed

`name/module` - Action to be performed

Running the playbook: `ansible-playbook calliope.yml`
Output verbosity can be -v to -vvvv
Dry run (to see what would be changed) -C



Multiple Plays

Orchestrate related deployment against different groups of hosts (may also use different become* keywords.

```
---
- name: Setup db server
  hosts: db

  tasks:
    - name: Install db server
      ansible.builtin.apt:
        name:
          - mariadb-server
        state: latest
...
- name: Setup db clients
  hosts: db-client-group
  tasks:
    - name: Install db client
      ansible.builtin.apt:
        name:
          - mariadb-client
        state: latest
...
```

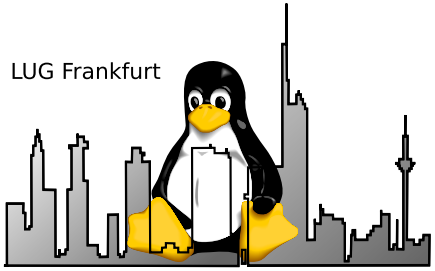
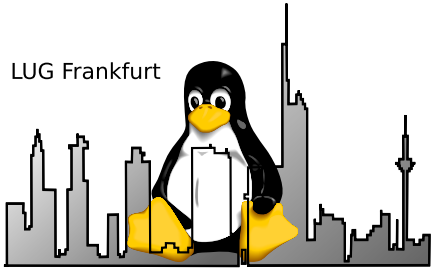


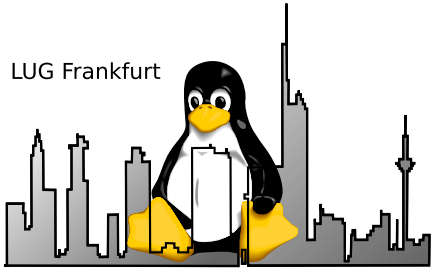
Table of Contents



Variables

- May be either defined in playbook or seperate file(s)
- in the play: *vars:*
- explicit external files: *vars_files:*
- in the inventory on host and group level
- if *group_vars* resp. *host_vars* directories exist, files with the same name as the current host/group will be automatically included

```
vars:
  my_loc: "Kronau"
vars_files:
  - vars/location.yml
```

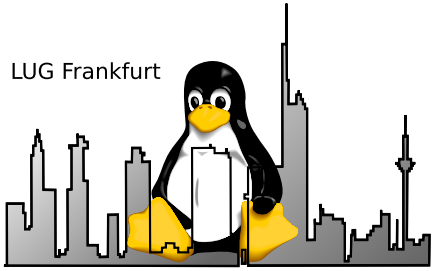


Variables (cont.)

- Variables are referred to using {{ var }}
- If a variable reference is starting a value it *must be quoted*
  e.g. owner: “{{ item }}”
- *... may be overridden using ansible-playbook my.yml -e "var=value"*
- *Can be structured (List or Array). Best practise: use array['key'] notation (vs. array.key)*
- *Store return value of task in registered variable*

```
- ansible.builtin.template:
    src: named-conf.j2
    dest: /etc/named.conf
    ...
  register: namedChange

 - ansible.builtin.service:
     name: named
     state: restarted
   when: namedChange.changed
```



Includes

- You may include variable files and task files
- Layout of the files as usual
- Used to structure and keep playbook size managable

```
- name: Include vars from YAML or JSON file
  ansible.builtin.include: vars/example.yml

- name: Include tasks here
  ansible.builtin.include: tasks/my_task.yml
```

1. Variables defined in include_vars files override all earlier declarations
2. Variables may also be defined with the vars keyword of the include module

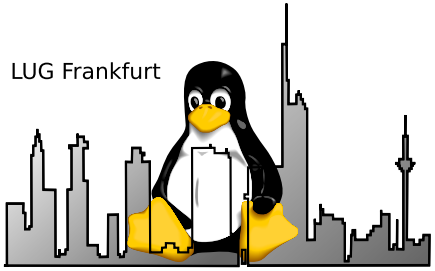
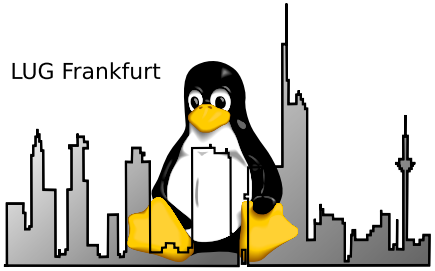


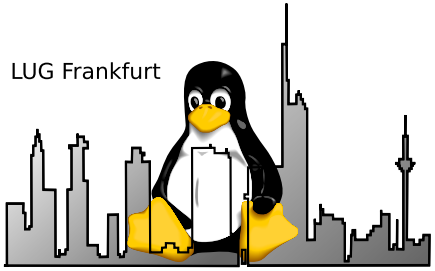
Table of Contents



ansible-lint

- Extra package to install on the control node
- Checks ansible playbook for adherance to best practices
- Layout (empty lines/trailing spaces/indentation)
- Use of fully qualified module name vs short form (e.g. ansible.builtin.copy vs copy)
- Plays/tasks named
- ...

Usage: ansible-lint playbook.yml



Sources

- ATIX Webinar Ansible Best Practices
- Red Hat Automation with Ansible Student workbook
- ... lots of internet searches and posts over the years

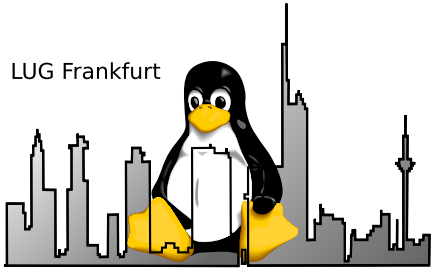
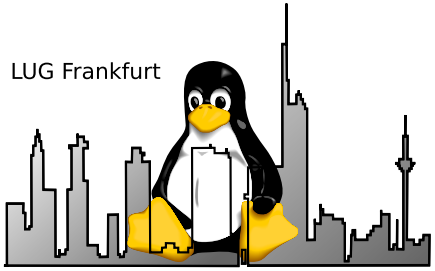


Table of Contents



Appendix: Semi advanced topics

- One may run the setup module with a filter, only returning the values needed in the playbook

```
    ansible.builtin.setup:
      filter:
        - ansible_distribution
- ansible_distribution_version
```

- Custom facts in /etc/ansible/facts.d
  - Static INI or JSON files; Executable script returning JSON
  - Must have extension .fact !
  - Returned by setup in ansible_local
- Information about ansible environment and groups in hostvars[“node”]

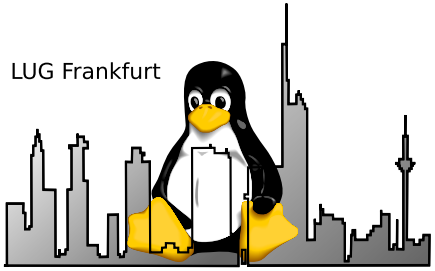
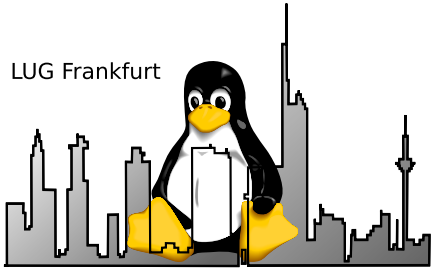


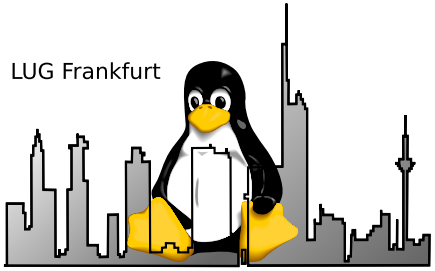
Table of Contents



Parallel Execution

Note: These two topics should have been in the slides from the start. There were some good questions about this during the talk.

Parallel Execution A play may be executed on a list or group of hosts. By default the play is executed on up to five nodes in parallel. Tasks run in order per node, but there is no provision that task x is completed on all nodes before task x+1 starts executing. This needs to be kept in mind when reading and interpreting the playbook output.
This behaviour can be configured, check on playbook execution strategies.



Error handling

Error handling Normal behaviour is, that ansible will stop a play for a node after any task fails on this node. Execution on all other hosts is not affected by this (e.g. if you run a playbook against five nodes and a task fails on two of them, the playbook will continue on the other three to the end).
If you follow best practices and your play is idempotent, you may just re-execute the playbook again after the error condition has been fixed. The result will be the same as if the first attempt would have been succeeded.
Also this behaviour can be configured, check the ansible documentation for Error handling in playbooks.

So Long, And Thanks for All the Fish