Ansible

Diego Pacheco

About Me

- ❑ Cat's Father
- ❑ Principal Software Architect
- ❑ Agile Coach
- ❑ SOA/Microservices Expert
- ❑ DevOps Practitioner
- ❑ Speaker
- ❑ Author

diegopacheco

@diego_pacheco

http://diego-pacheco.blogspot.com.br/

Diego Pacheco

Building Applications with Scala

Write modern, scalable, and reactive applications with the power of Scala

Packt>

Diego Pacheco

Building Effective Microservices

Explore microservices and their implementation hands-on

Packt>

Ansible

Is a Orchestration
And Automation Engine

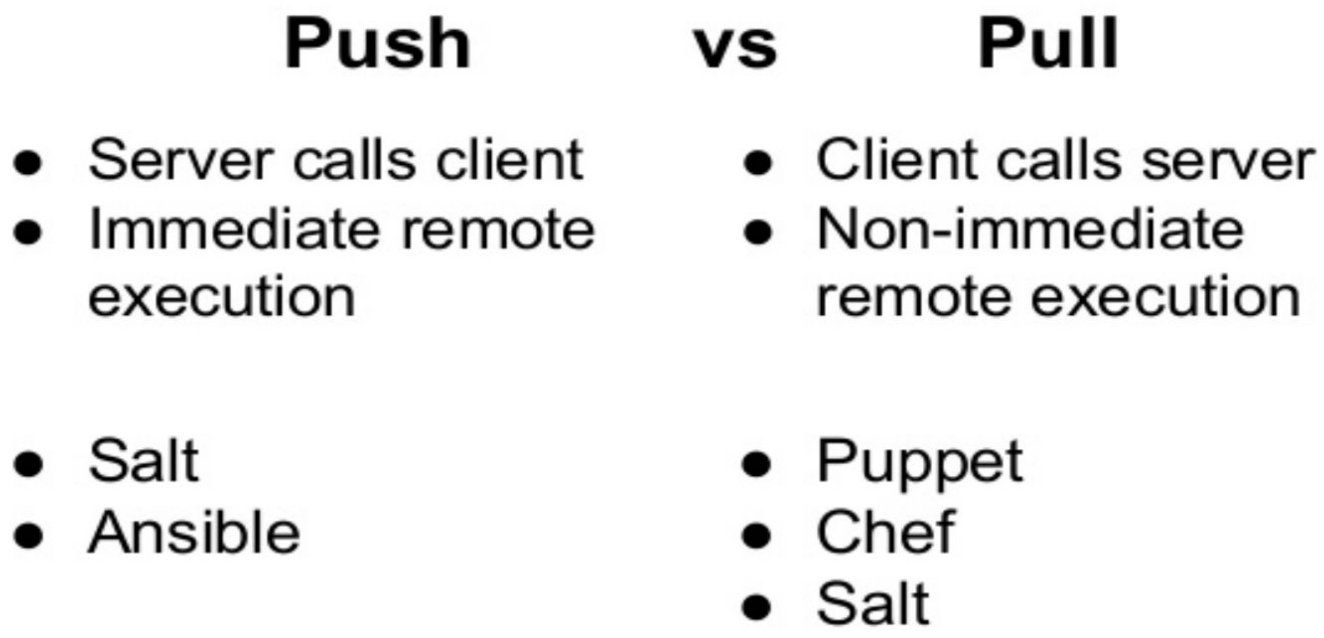
Ansible

Is the language used to write ansible.

Is the agent-less it just needs

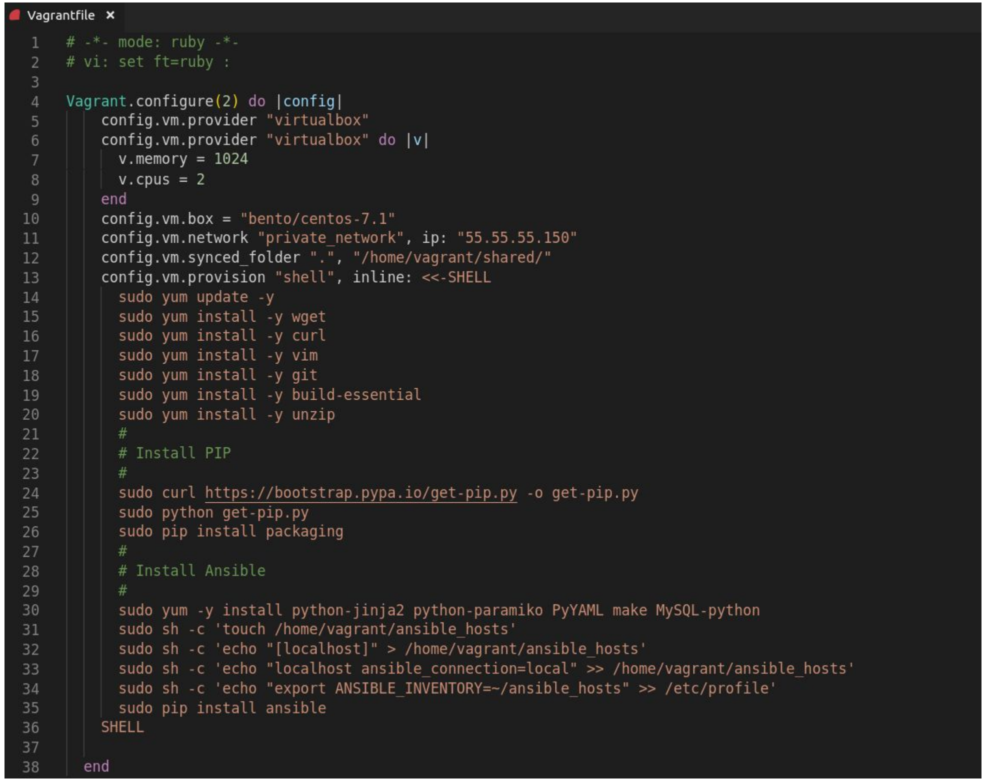
It`s based in recopies, for ansible
This recopies are called: playbooks.

Ansible is Push but has Pull



Push	vs	Pull
• Server calls client • Immediate remote execution		• Client calls server • Non-immediate remote execution
• Salt • Ansible		• Puppet • Chef • Salt

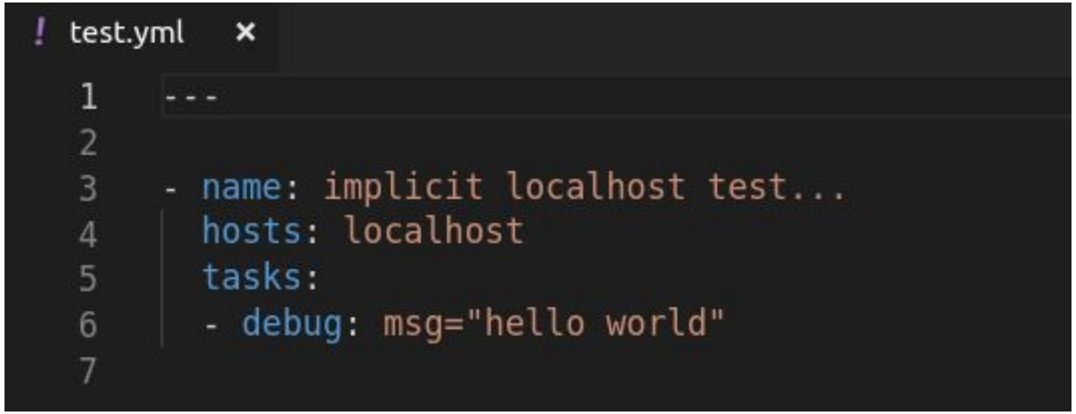
Ansible | installation



Vagrantfile

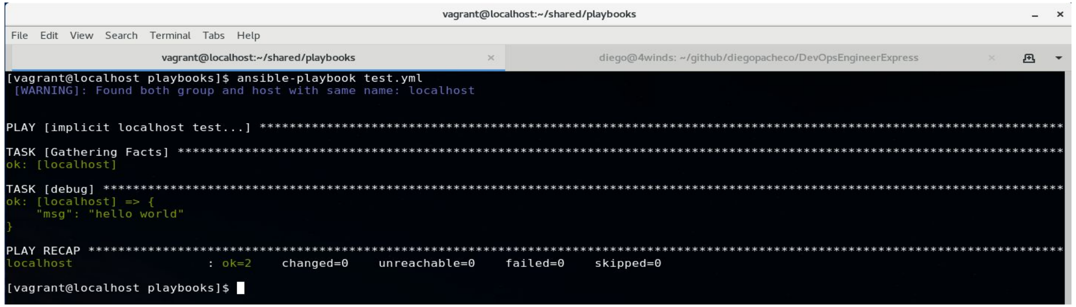
```
# -*- mode: ruby -*-
# vi: set ft=ruby :

Vagrant.configure(2) do |config|
    config.vm.provider "virtualbox"
    config.vm.provider "virtualbox" do |v|
      v.memory = 1024
      v.cpus = 2
    end
    config.vm.box = "bento/centos-7.1"
    config.vm.network "private_network", ip: "55.55.55.150"
    config.vm.synced_folder ".", "/home/vagrant/shared/"
    config.vm.provision "shell", inline: <<-SHELL
      sudo yum update -y
      sudo yum install -y wget
      sudo yum install -y curl
      sudo yum install -y vim
      sudo yum install -y git
      sudo yum install -y build-essential
      sudo yum install -y unzip
      #
      # Install PIP
      #
      sudo curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py
      sudo python get-pip.py
      sudo pip install packaging
      #
      # Install Ansible
      #
      sudo yum -y install python-jinja2 python-paramiko PyYAML make MySQL-python
      sudo sh -c 'touch /home/vagrant/ansible_hosts'
      sudo sh -c 'echo "[localhost]" > /home/vagrant/ansible_hosts'
      sudo sh -c 'echo "localhost ansible_connection=local" >> /home/vagrant/ansible_hosts'
      sudo sh -c 'echo "export ANSIBLE_INVENTORY=~/ansible_hosts" >> /etc/profile'
      sudo pip install ansible
    SHELL

  end
```

Ansible | ansible-playbook test.yml



test.yml

```yaml
---

- name: implicit localhost test...
  hosts: localhost
  tasks:
  - debug: msg="hello world"

```



vagrant@localhost:~/shared/playbooks

File Edit View Search Terminal Tabs Help
vagrant@localhost:~/shared/playbooks
diego@4winds: ~/github/diegopacheco/DevOpsEngineerExpress

```
[vagrant@localhost playbooks]$ ansible-playbook test.yml
 [WARNING]: Found both group and host with same name: localhost


PLAY [implicit localhost test...] *******************************************************************************************

TASK [Gathering Facts] ******************************************************************************************************
ok: [localhost]

TASK [debug] ****************************************************************************************************************
ok: [localhost] => {
    "msg": "hello world"
}

PLAY RECAP ******************************************************************************************************************
localhost                  : ok=2    changed=0    unreachable=0    failed=0    skipped=0

[vagrant@localhost playbooks]$ 
```

Ansible Usage | 3 Ways

LOCAL

You install ansible and install(provision) software on local machine only. This pattern is often used with Packer.

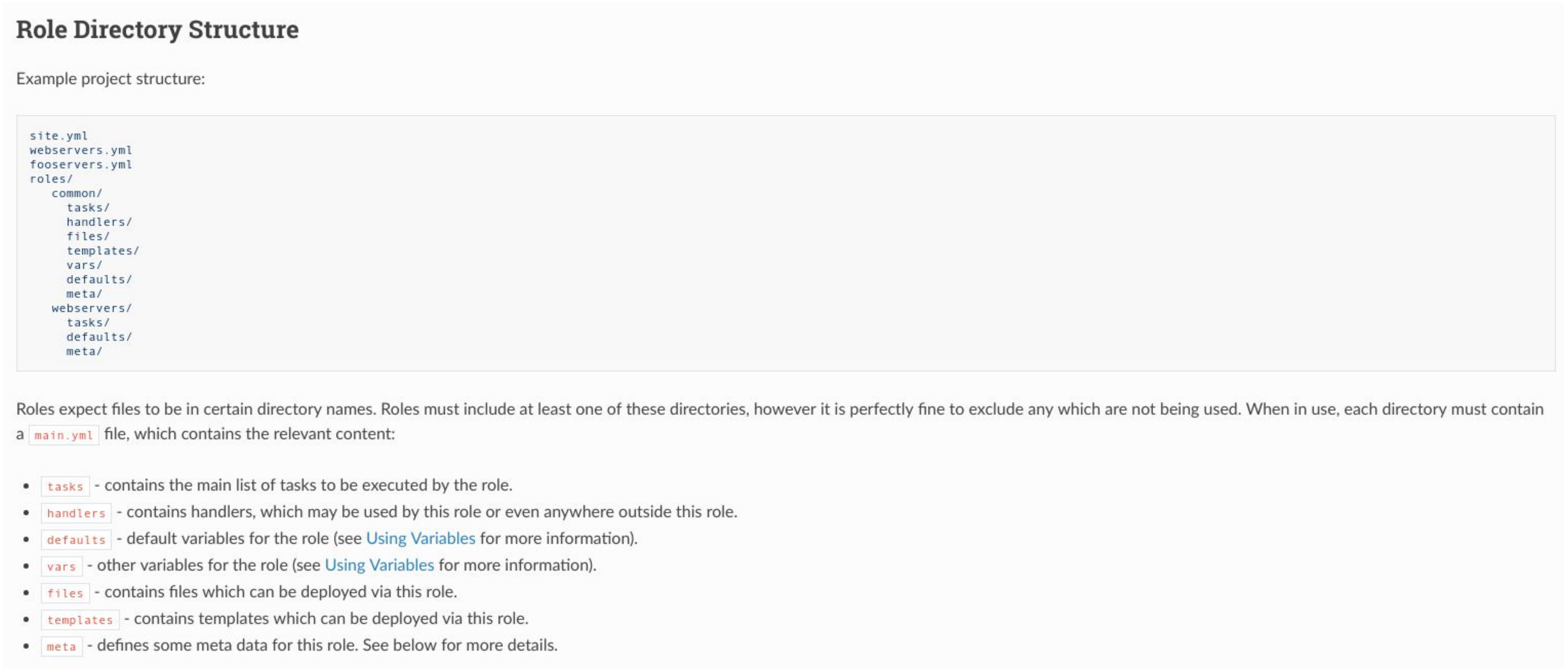
INVENTORY

Ansible can work with stati list of servers or dynamic list(dynamic inventory) in that case ansible does SSH to the machines and apply your playbooks that. That's the old pattern and you need to watch out to not hurt immutable infrastructure principle.

PULL

Basically that's the reverse flow. Ansible will call git for instance get the new files/configs and apply on the machine. IMHO that's more for configs rather than packages otherwise you could hurt immutable infrastructure as well. However there are better dinamic config solutions.

Ansible | Roles and Structure



Role Directory Structure

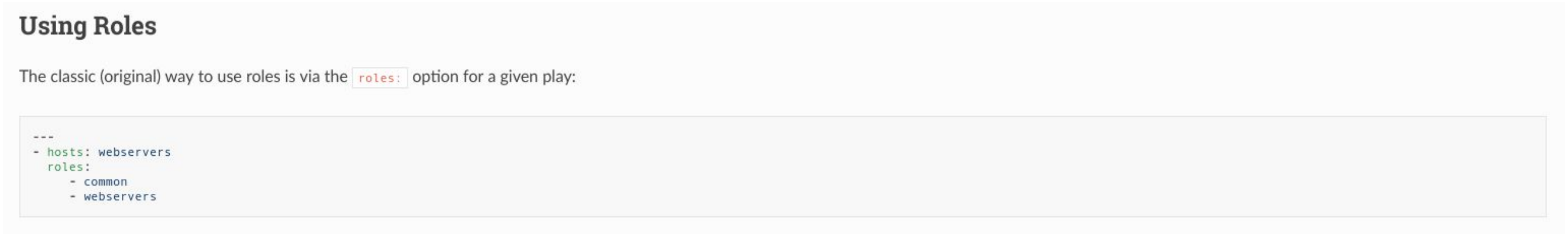
Example project structure:

```
site.yml
webservers.yml
fooservers.yml
roles/
   common/
     tasks/
     handlers/
     files/
     templates/
     vars/
     defaults/
     meta/
   webservers/
     tasks/
     defaults/
     meta/
```

Roles expect files to be in certain directory names. Roles must include at least one of these directories, however it is perfectly fine to exclude any which are not being used. When in use, each directory must contain a `main.yml` file, which contains the relevant content:

- `tasks` - contains the main list of tasks to be executed by the role.
- `handlers` - contains handlers, which may be used by this role or even anywhere outside this role.
- `defaults` - default variables for the role (see Using Variables for more information).
- `vars` - other variables for the role (see Using Variables for more information).
- `files` - contains files which can be deployed via this role.
- `templates` - contains templates which can be deployed via this role.
- `meta` - defines some meta data for this role. See below for more details.

Ansible | Roles and Structure

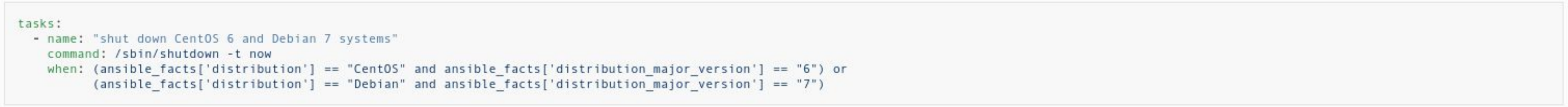


Using Roles

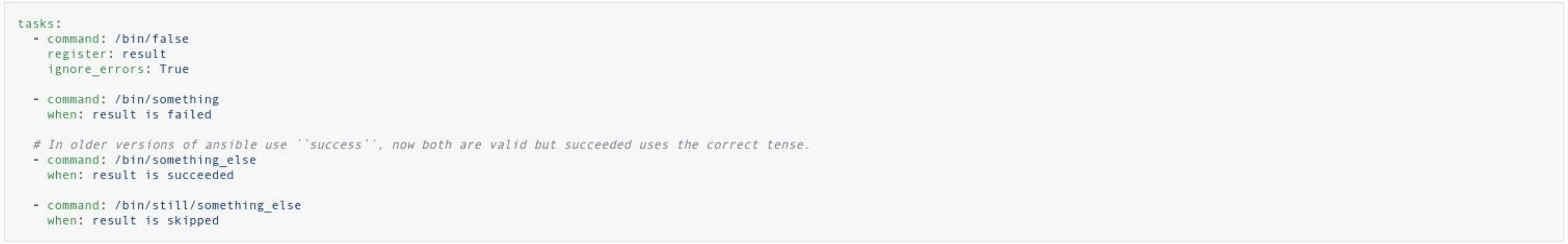
The classic (original) way to use roles is via the `roles:` option for a given play:

```
---
- hosts: webservers
  roles:
     - common
     - webservers
```

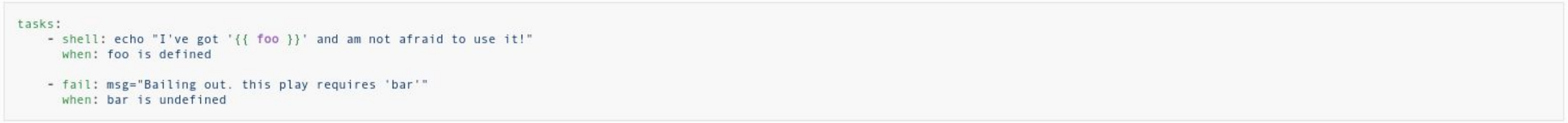
Ansible | Conditionals



```yaml
tasks:
  - name: "shut down CentOS 6 and Debian 7 systems"
    command: /sbin/shutdown -t now
    when: (ansible_facts['distribution'] == "CentOS" and ansible_facts['distribution_major_version'] == "6") or
          (ansible_facts['distribution'] == "Debian" and ansible_facts['distribution_major_version'] == "7")
```

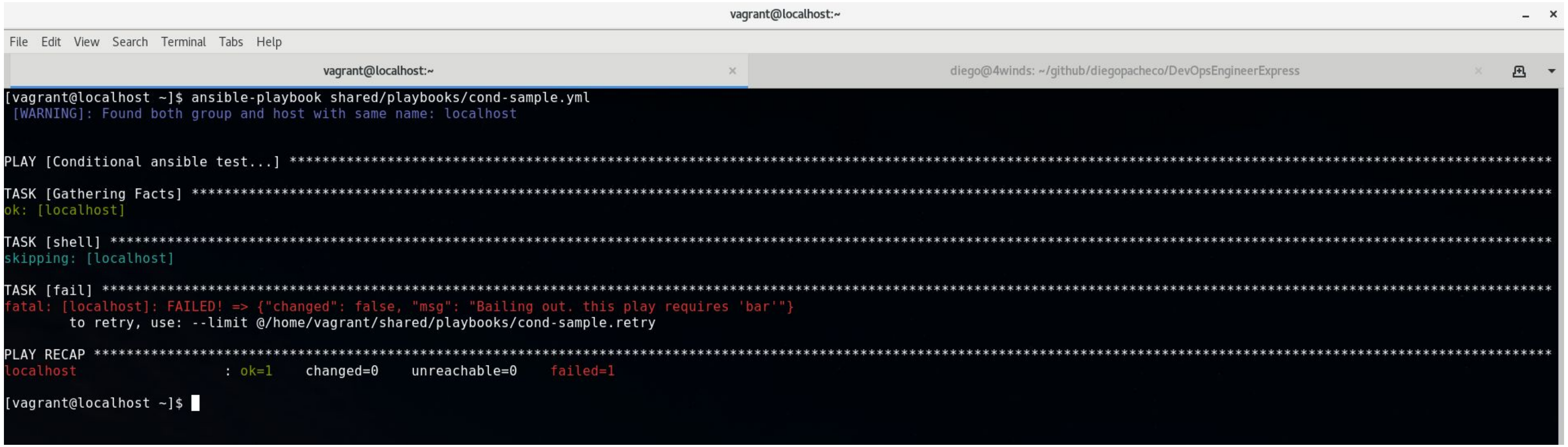


```yaml
tasks:
  - command: /bin/false
    register: result
    ignore_errors: True

  - command: /bin/something
    when: result is failed

  # In older versions of ansible use ``success``, now both are valid but succeeded uses the correct tense.
  - command: /bin/something_else
    when: result is succeeded

  - command: /bin/still/something_else
    when: result is skipped
```



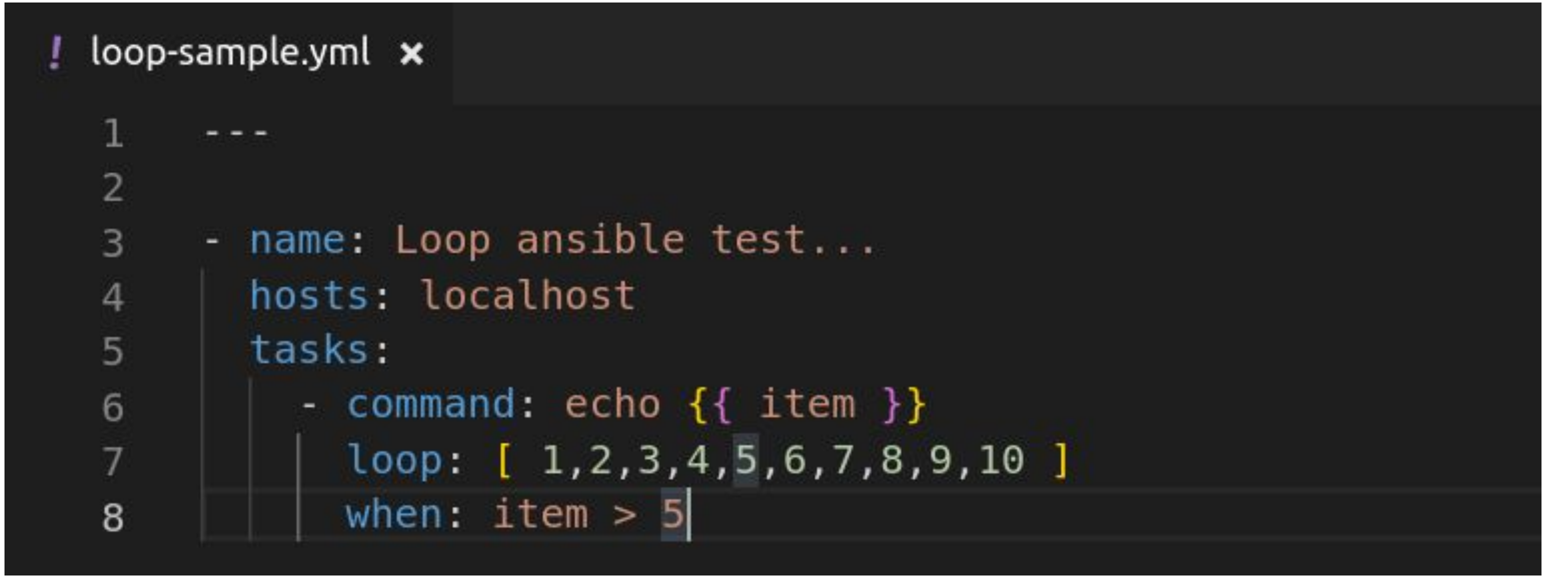
```yaml
tasks:
    - shell: echo "I've got '{{ foo }}' and am not afraid to use it!"
      when: foo is defined

    - fail: msg="Bailing out. this play requires 'bar'"
      when: bar is undefined
```

Ansible | Conditionals


vagrant@localhost:~
File Edit View Search Terminal Tabs Help
vagrant@localhost:~
diego@4winds: ~/github/diegopacheco/DevOpsEngineerExpress

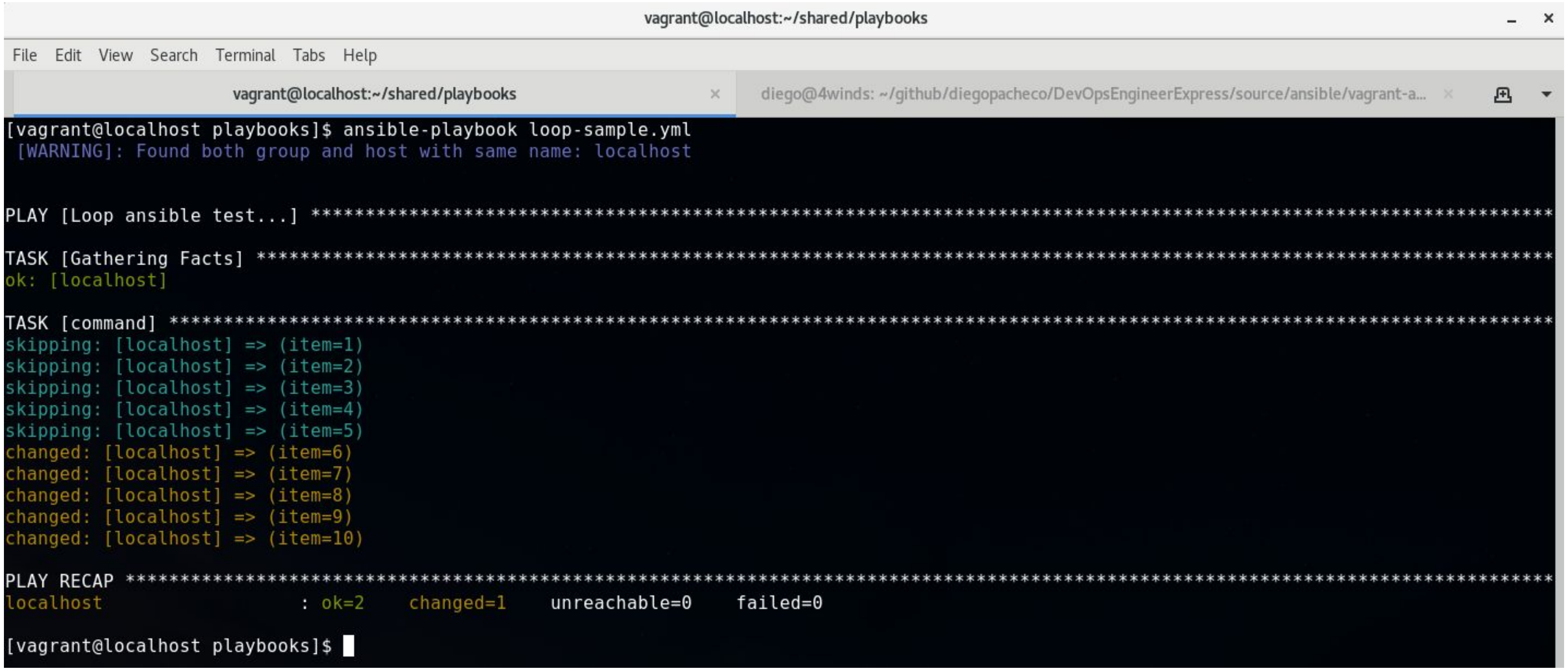
```
[vagrant@localhost ~]$ ansible-playbook shared/playbooks/cond-sample.yml
 [WARNING]: Found both group and host with same name: localhost


PLAY [Conditional ansible test...] *************************************************************************************************************************************************

TASK [Gathering Facts] *************************************************************************************************************************************************************
ok: [localhost]

TASK [shell] ***********************************************************************************************************************************************************************
skipping: [localhost]

TASK [fail] ************************************************************************************************************************************************************************
fatal: [localhost]: FAILED! => {"changed": false, "msg": "Bailing out. this play requires 'bar'"}
        to retry, use: --limit @/home/vagrant/shared/playbooks/cond-sample.retry

PLAY RECAP *************************************************************************************************************************************************************************
localhost                  : ok=1    changed=0    unreachable=0    failed=1

[vagrant@localhost ~]$
```

Ansible | Loops


loop-sample.yml

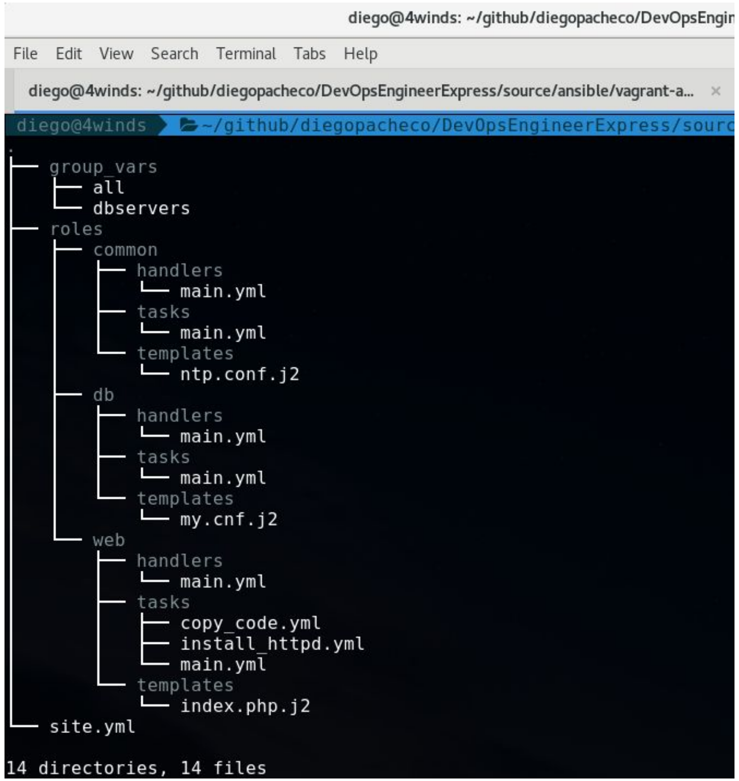
```yaml
---

- name: Loop ansible test...
  hosts: localhost
  tasks:
    - command: echo {{ item }}
      loop: [ 1,2,3,4,5,6,7,8,9,10 ]
      when: item > 5
```

Ansible | Loops


vagrant@localhost:~/shared/playbooks
File Edit View Search Terminal Tabs Help
vagrant@localhost:~/shared/playbooks
diego@4winds: ~/github/diegopacheco/DevOpsEngineerExpress/source/ansible/vagrant-a...

```
[vagrant@localhost playbooks]$ ansible-playbook loop-sample.yml
 [WARNING]: Found both group and host with same name: localhost


PLAY [Loop ansible test...] *******************************************************************************************

TASK [Gathering Facts] ************************************************************************************************
ok: [localhost]

TASK [command] ********************************************************************************************************
skipping: [localhost] => (item=1)
skipping: [localhost] => (item=2)
skipping: [localhost] => (item=3)
skipping: [localhost] => (item=4)
skipping: [localhost] => (item=5)
changed: [localhost] => (item=6)
changed: [localhost] => (item=7)
changed: [localhost] => (item=8)
changed: [localhost] => (item=9)
changed: [localhost] => (item=10)

PLAY RECAP ************************************************************************************************************
localhost                  : ok=2    changed=1    unreachable=0    failed=0

[vagrant@localhost playbooks]$ 
```

More Complicated Sample | LAMP Stack


diego@4winds: ~/github/diegopacheco/DevOpsEngin
File Edit View Search Terminal Tabs Help
diego@4winds: ~/github/diegopacheco/DevOpsEngineerExpress/source/ansible/vagrant-a...

```
 diego@4winds  ~/github/diegopacheco/DevOpsEngineerExpress/sourc
.
├── group_vars
│   ├── all
│   └── dbservers
├── roles
│   ├── common
│   │   ├── handlers
│   │   │   └── main.yml
│   │   ├── tasks
│   │   │   └── main.yml
│   │   └── templates
│   │       └── ntp.conf.j2
│   ├── db
│   │   ├── handlers
│   │   │   └── main.yml
│   │   ├── tasks
│   │   │   └── main.yml
│   │   └── templates
│   │       └── my.cnf.j2
│   └── web
│       ├── handlers
│       │   └── main.yml
│       ├── tasks
│       │   ├── copy_code.yml
│       │   ├── install_httpd.yml
│       │   └── main.yml
│       └── templates
│           └── index.php.j2
└── site.yml

14 directories, 14 files
```

https://github.com/diegopacheco/DevOpsEngineerExpress/tree/master/source/ansible/vagrant-ansible/playbooks/lamp_simple

Ansible is Flexible!

- There are several modules and plugins available.
  - Always use modules instead of doing bash by hand.
  - Makes linter easy and less error prone.
- You can create your own modules and plugins.
- More information on docs

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

  https://docs.ansible.com/ansible/latest/modules/modules_by_category.html

Ansible

Diego Pacheco