**1.)** [How to set linux environment variables with ansible](https://stackoverflow.com/questions/27733511/how-to-set-linux-environment-variables-with-ansible)

There are multiple ways to do this and from your question it's nor clear what you need.

**1. If you need environment variable to be defined PER TASK ONLY, you do this:**

- hosts: dev

tasks:

- name: Echo my\_env\_var

shell: "echo $MY\_ENV\_VARIABLE"

environment:

MY\_ENV\_VARIABLE: whatever\_value

- name: Echo my\_env\_var again

shell: "echo $MY\_ENV\_VARIABLE"

Note that MY\_ENV\_VARIABLE is available ONLY for the first task, environment does not set it permanently on your system.

TASK: [Echo my\_env\_var] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.111.222] => {"changed": true, "cmd": "echo $MY\_ENV\_VARIABLE", ... "stdout": "whatever\_value"}

TASK: [Echo my\_env\_var again] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

changed: [192.168.111.222] => {"changed": true, "cmd": "echo $MY\_ENV\_VARIABLE", ... "stdout": ""}

Hopefully soon using environment will also be possible on **play level**, not only task level as above. There's currently a pull request open for this feature on Ansible's GitHub: <https://github.com/ansible/ansible/pull/8651>

UPDATE: It's now merged as of Jan 2, 2015.

**2. If you want permanent environment variable + system wide / only for certain user**

You should look into how you do it in your Linux distribution / shell, there are multiple places for that. For example in Ubuntu you define that in files like for example:

* ~/.profile
* /etc/environment
* /etc/profile.d directory

You will find Ubuntu docs about it here: <https://help.ubuntu.com/community/EnvironmentVariables>

After all for setting environment variable in ex. Ubuntu you can just use lineinfile module from Ansible and add desired line to certain file. Consult your OS docs to know where to add it to make it permanent.

# 2.) [Ansible Playbooks vs Roles](https://stackoverflow.com/questions/32101001/ansible-playbooks-vs-roles)

Playbook vs Role vs [databases] and similar entries in /etc/ansible/hosts

[databases] is a single name for a group of hosts. It allows you to reference multiple hosts by a single name.

Role is a set of tasks and additional files to configure host to serve for a certain *role*.

Playbook is a mapping between hosts and roles.

Example from [documentation](http://docs.ansible.com/ansible/playbooks_roles.html#roles) describes example project. It contains two things:

* Playbooks. site.yml, webservers.yml, fooservers.yml are playbooks.
* Roles: roles/common/ and roles/webservers/ contain definitions of common and webserversroles accordingly.

Inside playbook (webservers.yml) you have something like:

---

- hosts: webservers <- this group of hosts defined in /etc/ansible/hosts, databases and mail\_servers in example from your question

roles: <- this is list of roles to assign to these hosts

- common

- webservers

If Playbooks are defined inside of YAML files, then where are Roles defined?

They are defined inside roles/\* directories. Roles are defined mostly using YAML files, but can also contain resources of any types (files/, templates/). According to [documentation](http://docs.ansible.com/ansible/latest/playbooks_reuse_roles.html#using-roles) role definition is structured this way:

* If roles/x/tasks/main.yml exists, tasks listed therein will be added to the play
* If roles/x/handlers/main.yml exists, handlers listed therein will be added to the play
* If roles/x/vars/main.yml exists, variables listed therein will be added to the play
* If roles/x/meta/main.yml exists, any role dependencies listed therein will be added to the list of roles (1.3 and later)
* Any copy tasks can reference files in roles/x/files/ without having to path them relatively or absolutely
* Any script tasks can reference scripts in roles/x/files/ without having to path them relatively or absolutely
* Any template tasks can reference files in roles/x/templates/ without having to path them relatively or absolutely
* Any include tasks can reference files in roles/x/tasks/ without having to path them relatively or absolutely

The most important file is roles/x/tasks/main.yml, here you define tasks, which will be executed, when role is executed.

Aside from the ansible.cfg living on the Ansible server, how do I add/configure Ansible with available Playbooks/Roles? For instance, when I run ansible-playbook someplaybook.yaml, how does Ansible know where to find that playbook?

$ ansible-playbook someplaybook.yaml

Will look for a playbook inside current directory.

$ ansible-playbook somedir/somedir/someplaybook.yaml

Will look for a playbook inside somedir/somedir/ directory.

It's your responsibility to put your project with all playbooks and roles on server. Ansible has nothing to do with that.

# 3.) [How to set host\_key\_checking=false in ansible inventory file?](https://stackoverflow.com/questions/23074412/how-to-set-host-key-checking-false-in-ansible-inventory-file)

I would like to use ansible-playbook command instead of 'vagrant provision'. However setting host\_key\_checking=false in the hosts file does not seem to work.

# hosts file

vagrant ansible\_ssh\_private\_key\_file=~/.vagrant.d/insecure\_private\_key

ansible\_ssh\_user=vagrant ansible\_ssh\_port=2222 ansible\_ssh\_host=127.0.0.1

host\_key\_checking=false

Is there a configuration variable outside of Vagrantfile that can override this value?

-🡪Yes, but not at the hosts/inventory level. You can do it at the global level.

* You can do it either in the /etc/ansible/ansible.cfg or ~/.ansible.cfg file:
* [defaults]
* host\_key\_checking = False
* Or you can pass it from the command line:
* ansible-playbook -e 'host\_key\_checking=False' yourplaybook.yml
* Or you can setup and env variable (this might not work on newer ansible versions):

export ANSIBLE\_HOST\_KEY\_CHECKING=False

# 4.) [Not possible to source .bashrc with Ansible](https://stackoverflow.com/questions/22256884/not-possible-to-source-bashrc-with-ansible)

-🡪The right way should be:

- hosts: all

tasks:

- name: source bashrc file

shell: "{{ item }}"

with\_items:

- source ~/.bashrc

- your other command

-🡪You have two options to use source with ansible. One is with the "shell:" command and /bin/sh (the ansible default). "source" is called "." in /bin/sh. So your command would be:

- name: source bashrc

sudo: no

shell: . /home/username/.bashrc && [the actual command you want run]

Note you have to run a command after sourcing .bashrc b/c each ssh session is distinct - every ansible command runs in a separate ssh transaction.

Your second option is to force Ansible shell to use bash and then you can use the "source" command:

- name: source bashrc

sudo: no

shell: source /home/username/.bashrc && [the actual command you want run]

args:

executable: /bin/bash

Finally, I'll note that you may want to actually source "/etc/profile" if you're on Ubuntu or similar, which more completely simulates a local login.

# 5.) [How to run Ansible without specifying the inventory but the host directly?](https://stackoverflow.com/questions/17188147/how-to-run-ansible-without-specifying-the-inventory-but-the-host-directly)

I want to run Ansible in Python without specifying the inventory file through (ANSIBLE\_HOST) but just by:

ansible.run.Runner(

module\_name='ping',

host='www.google.com'

)

I can actually do this in fabric easily but just wonder how to do this in Python. On the other hand, documentation of the Ansible API for python is not really complete.

-🡪Surprisingly, the trick is to append a ,

# Host and IP address

ansible all -i example.com,

ansible all -i 93.184.216.119,

or

# Requires 'hosts: all' in your playbook

ansible-playbook -i example.com, playbook.yml

The host parameter preceding the , can be either a hostname or an IPv4/v6 address.

-🡪I know this question is really old but think that this little trick might helpful for future users who need help for this:

ansible-playbook -i '10.254.3.133,' site.yml

if you run for local host:

ansible-playbook -i 'localhost,' --connection=local site.yml

The trick is that after ip address/dns name, put the comma inside the quotes and requires 'hosts: all' in your playbook.

# 6.) [include tasks from another role in ansible playbook](https://stackoverflow.com/questions/30192490/include-tasks-from-another-role-in-ansible-playbook)

I'm designing a kind of playbook lib with individual tasks

so in the usual roles repo, I have something like:

roles

├── common

│   └── tasks

│ ├── A.yml

│   ├── B.yml

│ ├── C.yml

│ ├── D.yml

│ ├── login.yml

│ ├── logout.yml

│   └── save.yml

├── custom\_stuff\_workflow

│ └── tasks

│ └── main.yml

└── other\_stuff\_workflow

└── tasks

└── main.yml

my main.yml in custom\_stuff\_workflow then contain something like:

---

- include: login.yml

- include: A.yml

- include: C.yml

- include: save.yml

- include: logout.yml

and this one in the other workflow:

---

- include: login.yml

- include: B.yml

- include: A.yml

- include: D.yml

- include: save.yml

- include: logout.yml

I can't find a way to do it in a natural way: one way that worked was having all tasks in a single role and tagging the relevant tasks while including a custom\_stuff\_workflow

The problem I have with that is that tags cannot be set in the calling playbook: it's only to be set at command line as I'm distributing this ansible repo with many people in the company, I can't rely on command line invocations (it would be nice to have a #! header in yml to be processed by ansible-playbook command)

I could also copy the relevant tasks (inside common in the above tree) in each workflow, but I don't want to repeat them around

Can someone see a solution to achieve what I'd like without repeating the tasks over different roles?

I guess the corner stone of my problem is that I define tasks as individual and it looks not natural in ansible...

Thanks a lot

PS: note that the tasks in the workflow have to be done in specific order and the only natural steps to abstract would be the login and save/logout

PPS: I've seen this question [How do I call a role from within another role in Ansible?](https://stackoverflow.com/questions/22078333/how-do-i-call-a-role-from-within-another-role-in-ansible) but it does not solve my problem as it's invoking a full role and not a subset of the tasks in a role

-🡪

|  |  |
| --- | --- |
|  | Yes, Ansible doesn't really like tasks as individual components. I think it wants you to use roles, but I can see why you wouldn't want to use roles for simple, reusable tasks.  I currently see two possible solutions: **1. Make those task-files into roles and use dependencies** Then you could do something like this in e.g. custom\_stuff\_workflow  dependencies:  - { role: login }  See: <https://docs.ansible.com/playbooks_roles.html#role-dependencies> **2. Use include with "hardcoded" paths to the task files** - include: ../../common/tasks/login.yml  That worked pretty well in a short test playbook I just did. Keep in mind, you can also use parameters etc. in those includes.  See: <https://docs.ansible.com/playbooks_roles.html#task-include-files-and-encouraging-reuse>  I hope I understood that question correctly and this helps. |

# 7.) [SSH Agent Forwarding with Ansible](https://stackoverflow.com/questions/24124140/ssh-agent-forwarding-with-ansible)

# 8.) [Write variable to a file in Ansible](https://stackoverflow.com/questions/26638180/write-variable-to-a-file-in-ansible)