Red Hat
Ansible Automation
Platform

Ansible 101

Introduction developing Ansible Automation



Ansible playbooks

```
---
- name: Install and start apache
  hosts: web
  become: true

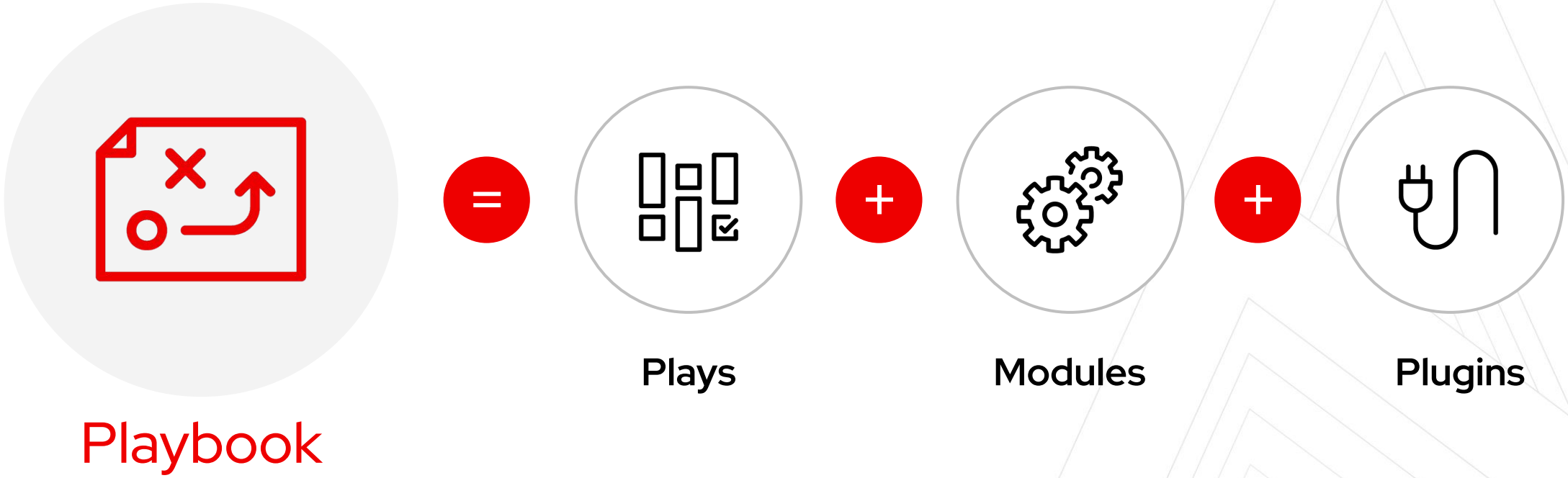
  tasks:
    - name: Ensure the httpd package is installed
       ansible.builtin.package:
         name: httpd
         state: present

    - name: Create the index.html file
      ansible.builtin.template:
        src: files/index.html
        dest: /var/www/html/

    - name: Start the httpd service if needed
      ansible.builtin.service:
        name: httpd
        state: started
```


Red Hat
Ansible Automation
Platform

What makes up an Ansible playbook?



Red Hat
Ansible Automation
Platform

Ansible plays. What am I automating?

What are they?

- Top level specification for a group of tasks
- Will tell that play which hosts it will execute on and control behavior such as fact gathering or privilege level

Building blocks for playbooks

- Multiple plays can exist within an Ansible playbook

```
---
- name: Ensure the httpd package is installed
  hosts: web
  become: true
```

Ansible modules. The “tools in the toolkit”.

What are they?

- Parameterized components with internal logic, representing a single step to be done
- The modules “do” things in Ansible

Language

- Usually created in Python, or Powershell for Windows setups, but can be developed in any language

```
- name: Create the index.html file
  ansible.builtin.template:
    src: files/index.html
    dest: /var/www/html/
```

Ansible plugins. The “extra bits”.

What are they?

- Plugins are pieces of code that augment Ansible’s core functionality
- Ansible uses a plugin architecture to enable a rich, flexible, and expandable feature set

```
---
- name: Example Playbook Using json_query Filter
  hosts: localhost
  gather_facts: no

  vars:
    complex_data:
      users:
        - name: "Alice"
          age: 25
        - name: "Bob"
          age: 30
        - name: "Charlie"
          age: 35

  tasks:
    - name: Extract names of all users
      ansible.builtin.debug:
        msg: "{{ complex_data | community.general.json_query('users[*].name') }}"
```

Ansible Inventory. The systems that a playbook runs against.

What are they?

- List of systems in your infrastructure that automation is executed against

How do they work?

- Defines the systems that Ansible manages and targets for automation.
- Organizes hosts into groups (e.g., web servers, databases) for better management.
- Group variables apply settings across multiple systems efficiently.
- Host-specific variables allow detailed customization for individual systems.

```
[web]
webserver1.example.com
webserver2.example.com

[db]
dbserver1.example.com

[switches]
leaf01.internal.com
leaf02.internal.com
```

Ansible Inventory

```
[app1srv]
appserver01 ansible_host=10.42.0.2
appserver02 ansible_host=10.42.0.3

[web]
node-[1:30]

[web:vars]
apache_listen_port=8080
apache_root_path=/var/www/mywebdocs/

[all:vars]
ansible_user=kev
ansible_ssh_private_key_file=/home/kev/.ssh/id_rsa
```


Red Hat
Ansible Automation
Platform

Ansible Inventory

```
[app1srv]
appserver01 ansible_host=10.42.0.2
appserver02 ansible_host=10.42.0.3

[web]
node-[1:30]

[web:vars]
apache_listen_port=8080
apache_root_path=/var/www/mywebdocs/

[all:vars]
ansible_user=kev
ansible_ssh_private_key_file=/home/kev/.ssh/id_rsa
```


Red Hat
Ansible Automation
Platform

Ansible playbook

A play

```
---
- name: Install and start apache
  hosts: web
  become: true

  tasks:
    - name: Ensure the httpd package is installed
      ansible.builtin.package:
        name: httpd
        state: present

    - name: Create the index.html file
      ansible.builtin.template:
        src: files/index.html
        dest: /var/www/html/

    - name: Start the httpd service if needed
      ansible.builtin.service:
        name: httpd
        state: started
```


Red Hat
Ansible Automation
Platform

Ansible playbook

A task

```
---
- name: Install and start apache
  hosts: web
  become: true

  tasks:
    - name: Ensure the httpd package is installed
      ansible.builtin.package:
        name: httpd
        state: present

    - name: Create the index.html file
      ansible.builtin.template:
        src: files/index.html
        dest: /var/www/html/

    - name: Start the httpd service if needed
      ansible.builtin.service:
        name: httpd
        state: started
```


Red Hat
Ansible Automation
Platform

Ansible playbook

A module 

```yaml
---
- name: Install and start apache
  hosts: web
  become: true

  tasks:
    - name: Ensure the httpd package is installed
      ansible.builtin.package:
        name: httpd
        state: present

    - name: Create the index.html file
      ansible.builtin.template:
        src: files/index.html
        dest: /var/www/html/

    - name: Start the httpd service if needed
      ansible.builtin.service:
        name: httpd
        state: started
```

Running an Ansible playbook. The most important colors of Ansible.

A task executed as expected, no change was made.

A task executed as expected, making a change.

A task failed to execute successfully.

Running an Ansible Playbook with `ansible-navigator`

What is it?

It is a command line utility and text-based user interface (TUI) for running, testing and developing Ansible automation content

- Review EEs
- Develop collections
- Develop playbooks
- Troubleshoot problems

```
# Direct command-line interface method
$ ansible-navigator run playbook.yml \
-i inventory.ini \
-m stdout


# Text-based User Interface method
$ ansible-navigator run playbook.yml -i inventory.ini
```

Ansible Variable Playbook

```
---
- name: variable playbook test
  hosts: localhost

  vars:
    var_one: awesome
    var_two: ansible is
    var_three: "{{ var_two }} {{ var_one }}"

  tasks:
    - name: print out var_three
      ansible.builtin.debug:
        msg: "{{ var_three }}"
```

Ansible Variable Playbook

```
---
- name: variable playbook test
  hosts: localhost

  vars:
    var_one: awesome
    var_two: ansible is
    var_three: "{{ var_two }} {{ var_one }}"

  tasks:
    - name: print out var_three
      ansible.builtin.debug:
        msg: "{{ var_three }}"
```

ansible is awesome

Red Hat
Ansible Automation
Platform

Ansible Facts

Facts

- Just like variables, really...
- ... but: coming from the host itself!
- Check them out with the setup module

```
tasks:
  - name: Collect all facts of host
    ansible.builtin.setup:
      gather_subset:
        - 'all'
```


Red Hat
Ansible Automation
Platform

Ansible Facts Playbook

```
---
- name: Ansible Facts playbook
  hosts: localhost
  gather_facts: true

  tasks:
    - name: Define custom fact
      ansible.builtin.set_fact:
        server_role: “frontend”

    - name: Use custom and built-in fact
      ansible.builtin.debug:
        msg: ‘The {{ ansible_hostname }} is assigned the role: {{ server_role }}’
```

```
$ ansible-navigator run facts_playbook.yml
```


Red Hat
Ansible Automation
Platform

Conditionals

What are they?

- Conditionals allow tasks to run **only if certain conditions are met.**
- Enable **dynamic automation** by checking the values of variables & making decisions at runtime.

```
vars:
   my_mood: happy

 tasks:
 - name: task, based on my_mood var
   ansible.builtin.debug:
     msg: "Yay! I am {{ my_mood }}!"
   when: my_mood == "happy"
```


Red Hat
Ansible Automation
Platform

Ansible Handler Tasks

```
---
- name: variable playbook test
  hosts: localhost

  tasks:
  - name: Ensure httpd package is present
    ansible.builtin.package:
      name: httpd
      state: latest
  notify: restart_httpd

  handlers:
  - name: restart_httpd
    ansible.builtin.service:
      name: httpd
      state: restarted
```

Ansible Handler Tasks

```yaml
tasks:
- name: Ensure httpd package is present
  ansible.builtin.package:
    name: httpd
    state: latest
  notify: restart_httpd

- name: Standardized index.html file
  ansible.builtin.copy:
    content: "This is my index.html file for {{ ansible_host }}"
    dest: /var/www/html/index.html
  notify: restart_httpd
```

```
TASK [Ensure httpd package is present] ***********************************************
ok: [web2]
ok: [web1]

TASK [Standardized index.html file] **************************************************
changed: [web2]
changed: [web1]

NOTIFIED: [restart_httpd] ************************************************************
changed: [web2]
changed: [web1]
```

unchanged

changed

handler runs once

If **either** task notifies a **changed** result, the handler will be notified **ONCE**.

Ansible Handler Tasks

```yaml
tasks:
- name: Ensure httpd package is present
  ansible.builtin.package:
    name: httpd
    state: latest
  notify: restart_httpd

- name: Standardized index.html file
  ansible.builtin.copy:
    content: "This is my index.html file for {{ ansible_host }}"
    dest: /var/www/html/index.html
  notify: restart_httpd
```

If **both** of these tasks notifies of a **changed** result, the handler will be notified **ONCE**.

```
TASK [Ensure httpd package is present] ***********************************************
changed: [web2]
changed: [web1]

TASK [Standardized index.html file] **************************************************
changed: [web2]
changed: [web1]

NOTIFIED: [restart_httpd] ************************************************************
changed: [web2]
changed: [web1]
```

changed

changed

handler runs once

Ansible Handler Tasks

```
tasks:
- name: Ensure httpd package is present
  ansible.builtin.package:
    name: httpd
    state: latest
  notify: restart_httpd

- name: Standardized index.html file
  ansible.builtin.copy:
    content: "This is my index.html file for {{ ansible_host }}"
    dest: /var/www/html/index.html
  notify: restart_httpd
```

If **neither** task notifies a **changed** result, the handler *does not run.*

```
TASK [Ensure httpd package is present] ******************************************
ok: [web2]
ok: [web1]

TASK [Standardized index.html file] *********************************************
ok: [web2]
ok: [web1]

PLAY RECAP **********************************************************************
web2       : ok=2   changed=0   unreachable=0   failed=0   skipped=0   rescued=0 ignored=0
web1       : ok=2   changed=0   unreachable=0   failed=0   skipped=0   rescued=0 ignored=0
```

unchanged

unchanged

Ansible Variables and Loops

```
---
- name: Ensure users
  hosts: node1
  become: true

  tasks:
    - name: Ensure user is present
      ansible.builtin.user:
        name: "{{ item }}"
        state: present
      loop:
        - dev_user
        - qa_user
        - prod_user
```

Templates

What are they?

- Templates are files that dynamically generate content by using variables, conditionals, loops and filters.

How do they work?

- First by reading in the source template file (.j2)
- Filling in the dynamic values based on the variables or facts
- Deploy the processed file with the variable values in place

```
---
- name: Ensure apache is installed and started
  hosts: web
  become: true
  vars:
    http_port: 80
    http_docroot: /var/www/mysite.com

  tasks:
    - name: Verify correct config file is present
      ansible.builtin.template:
        src: templates/httpd.conf.j2
        dest: /etc/httpd/conf/httpd.conf
```


Red Hat
Ansible Automation
Platform

Ansible Templates

```yaml
- name: Ensure apache is installed and started
  hosts: web
  become: true
  vars:
    http_port: 80
    http_docroot: /var/www/mysite.com

  tasks:
    - name: Verify correct config file is present
      ansible.builtin.template:
        src: templates/httpd.conf.j2
        dest: /etc/httpd/conf/httpd.conf
```

```
## Excerpt from httpd.conf.j2

# Change this to Listen on specific IP addresses as shown below to
# prevent Apache from glomming onto all bound IP addresses.
#
# Listen 80   ## original line
Listen {{ http_port }}

# DocumentRoot: The directory out of which you will serve your
# documents.
# DocumentRoot "/var/www/html"
DocumentRoot {{ http_docroot }}
```

Debugging in Ansible. Identify and Resolve Issues.

How does it work?

- Debugging helps identify and resolve issues in playbooks.

- Ansible offers several methods for debugging, including:
  - Debug module
  - Increased verbosity levels

```
- name: Display Variable Value
  ansible.builtin.debug:
    var: apache_service_name

- name: Display Custom Message
  ansible.builtin.debug:
    msg: "Apache service name is {{ apache_service_name }}"
```



Where to go next

Learn more

- Workshops
- Documents
- Youtube
- X

Get started

- Self-paced labs
- Ansible Automation Platform trial
- console.redhat.com
- Ansible Lightspeed trial

Get serious

- Red Hat Automation Adoption Journey
- Red Hat Training
- Red Hat Consulting

Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.



linkedin.com/company/red-hat



youtube.com/c/AnsibleAutomation



facebook.com/redhatinc



twitter.com/ansible