

Ansible Callback Plugins

Jiri Tyr

About me

- Using Ansible since 2014
- Ansible contributor
 - Modules: `yum_repository`, `jenkins_plugin`, `ldap_attr`, `ldap_entry`
 - Jinja2 filter: `comment`
 - Bug fixing (`mount` module)
 - Code reviews
- Author of more than 100 publicly available Ansible roles
 - <https://github.com/jtyr>
 - <https://galaxy.ansible.com/jtyr>

What are callback plugins?

What are callback plugins?

- Control most of the Ansible's output by responding to events
- Over 30 built-in plugins
 - Format output (json, yaml, debug, dense, minimal, unixy, stderr, null, ...)
 - Send output to other system (slack, jabber, mail, grafana_annotations, logstash, splunk, ...)

How to use it?

How to use it?

Command line

```
export ANSIBLE_STDOUT_CALLBACK=json  
ansible-playbook -i hosts site.yaml
```

```
export ANSIBLE_LOAD_CALLBACK_PLUGINS=yes  
export ANSIBLE_STDOUT_CALLBACK=json  
ansible all -i hosts -m ping
```

ansible.cfg

```
[defaults]  
callback_plugins = path/to/my/plugins  
Stdout_callback = json  
bin_ansible_callbacks = True
```

How to use it?

```
# Query variable from Ansible
export ANSIBLE_LOAD_CALLBACK_PLUGINS=yes
export ANSIBLE_STDOUT_CALLBACK=json
export MYHOST=host01
ansible $MYHOST -i hosts -m debug -a 'var=ansible_host' | \
jq ".plays[0].tasks[0].hosts[\"$MYHOST\"].ansible_host" | \
sed -e 's/^"//' -e 's/"$//' -e 's/^null$//'
```

How does it work?

How does it work?

- Two versions of callback functions
 - Functions for Ansible 2.x prefixed with v2_
- Three categories of callback functions
 - Playbook (playbook, play, task)
 - Runner (module)
 - Other (any, diff)

How does it work?

Playbook

```
v2_playbook_on_cleanup_task_start
v2_playbook_on_handler_task_start
v2_playbook_on_import_for_host
v2_playbook_on_include
v2_playbook_on_no_hosts_matched
v2_playbook_on_no_hosts_remaining
v2_playbook_on_notify
v2_playbook_on_not_import_for_host
v2_playbook_on_play_start
v2_playbook_on_start
v2_playbook_on_stats
v2_playbook_on_task_start
v2_playbook_on_vars_prompt
```

Runner

```
v2_runner_item_on_failed
v2_runner_item_on_ok
v2_runner_item_on_skipped
v2_runner_on_async_failed
v2_runner_on_async_ok
v2_runner_on_async_poll
v2_runner_on_failed
v2_runner_on_ok
v2_runner_on_skipped
v2_runner_on_start
v2_runner_on_unreachable
v2_runner_retry
```

Other

```
v2_on_any
v2_on_file_diff
```

How does it work?

```
# callback_plugins/mycallback.py
from ansible.plugins.callback import CallbackBase
```

```
class CallbackModule(CallbackBase):
    CALLBACK_VERSION = 2.0
    CALLBACK_TYPE = 'stdout'
    CALLBACK_NAME = 'mycallback'

    def v2_playbook_on_play_start(self, play):
        self._display.display("Starting play")

    def v2_playbook_on_stats(self, stats):
        self._display.display("Showing stats")
```

```
$ ANSIBLE_STDOUT_CALLBACK=mycallback ansible-playbook -i localhost, site.yaml
Starting play
Showing stats
```

CSV callback plugin

CSV callback plugin

- Allows to use Ansible as a reporting tool
- Turns Ansible's output into CSV (Comma-Separated Values)
- Meant to be used by plays producing output to STDOUT (`raw` or `shell` module)
- The output must be either valid CSV format or it has to fail (no empty string)
- Unsuccessful [PR #65801](#)

CSV callback plugin

```
# Default usage
$ export ANSIBLE_CALLBACK_PLUGINS=callback_plugins
$ export ANSIBLE_LOAD_CALLBACK_PLUGINS=1
$ export ANSIBLE_STDOUT_CALLBACK=csv
$ ansible all -i localhost, -c local -m raw -a "echo ','"
host,status
localhost,ok

TOTAL: 1
OK: 1
FAILED: 0
UNREACHABLE: 0
```

CSV callback plugin

```
# Customized usage
[callback_csv]
fields = [
    {'placeholder': '%n', 'name': 'name'},
    {'name': 'host', 'in_message': True},
    {'placeholder': '%h', 'name': 'ip', 'in_message': True},
    {'name': 'distro', 'in_message': True},
    {'name': 'version', 'in_message': True},
    {'name': 'cpu', 'in_message': True},
    {'name': 'mem', 'in_message': True},
    {'placeholder': '%s', 'name': 'status'},
    {'placeholder': '%g', 'name': 'group'},
    {'placeholder': '%v', 'variable': 'inventory_hostname_short', 'name': 'short'}]
format_ok = %n,%m,%s,%g,%v
```

CSV callback plugin

```
# Customized usage
$ export ANSIBLE_CALLBACK_PLUGINS=callback_plugins
$ export ANSIBLE_LOAD_CALLBACK_PLUGINS=1
$ export ANSIBLE_STDOUT_CALLBACK=csv
$ ansible all -i localhost, -c local -m raw \
    -a "echo $(cat ./facts.sh | base64 -w0) |
        base64 -di > ./facts.sh &&
        chmod +x ./facts.sh &&
        ./facts.sh"
name,hostname,ip,distro,version,cpu,mem,status,group,short
localhost,my.hostname.com,192.168.1.123,centos,7,4,3881048,ok,ungrouped,localhost

TOTAL: 1
OK: 1
FAILED: 0
UNREACHABLE: 0
```


CSV callback plugin

```
# Customized usage - Windows
$ export ANSIBLE_CALLBACK_PLUGINS=callback_plugins
$ export ANSIBLE_LOAD_CALLBACK_PLUGINS=1
$ export ANSIBLE_STDOUT_CALLBACK=csv
$ ansible host01 -i hosts -m win_shell -a '$myhost =
[System.Net.Dns]::GetHostByName((hostname)).HostName; $ip = Test-Connection -ComputerName
(hostname) -Count 1 | Select IPV4Address | %{$_.IPV4Address} | %{$_.IPAddressToString};
$version = Get-CimInstance Win32_OperatingSystem | select Version | %{$_.Version}; $cpu =
Get-WmiObject -class Win32_ComputerSystem | select NumberOfProcessors |
%{$_.NumberOfProcessors}; $mem = Get-WmiObject -class Win32_ComputerSystem | select
TotalPhysicalMemory | %{$_.TotalPhysicalMemory}; "{0},{1},windows,{2},{3},{4}" -f
$myhost.ToLower(), $ip, $version, $cpu, [int]($mem/1000)'
name,hostname,ip,distro,version,cpu,mem,status,group,short
host01,my.hostname.com,192.168.1.456,windows,10.0.18363,4,3881048,ok,ungrouped,host01

TOTAL: 1
OK: 1
FAILED: 0
UNREACHABLE: 0
```

Demo

Thank you for your attention!

Questions?