

【PTP SE 認定取得コース】

Red Hat SE Specialist - Automation

～ AAP 2.1 座学～

レッドハット株式会社 テクニカルセールス本部
パートナーソリューションアーキテクト部
アソシエイトプリンシパルソリューションアーキテクト
岡野 浩史

最終更新 : 2022年 3月

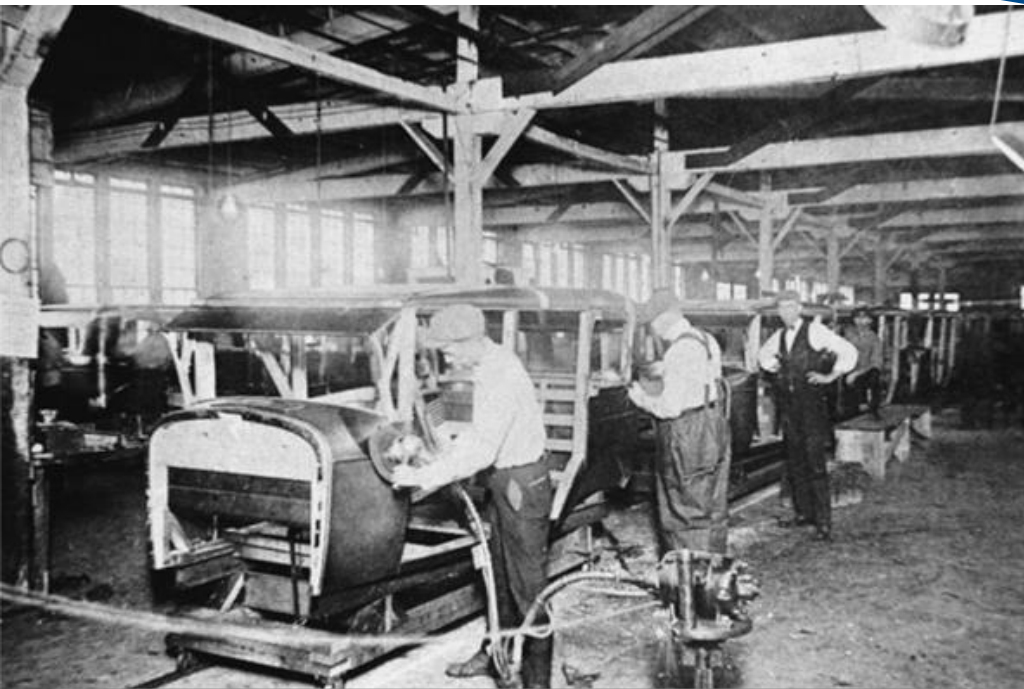


理想的なインフラ管理

人はもう『そこ』にはいない...

人が作業
(機械で補助する)

機械が作業
(人が管理する)



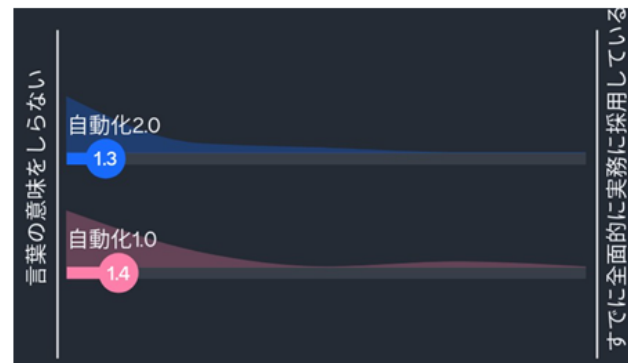
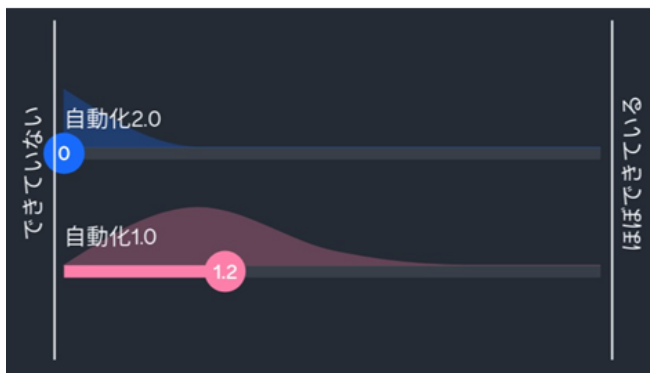
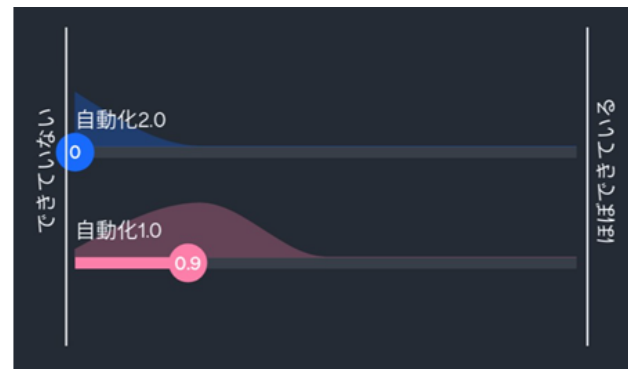
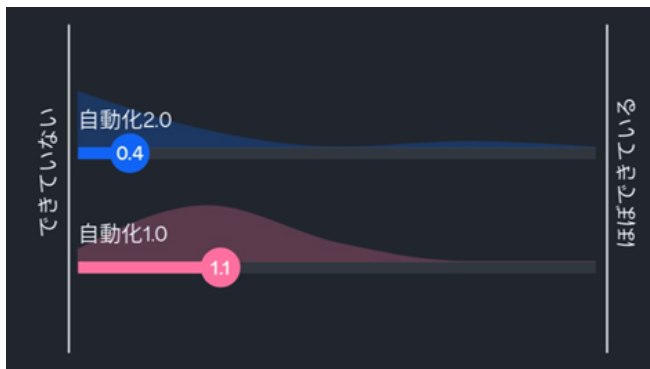
自動化以前～自動化1.0



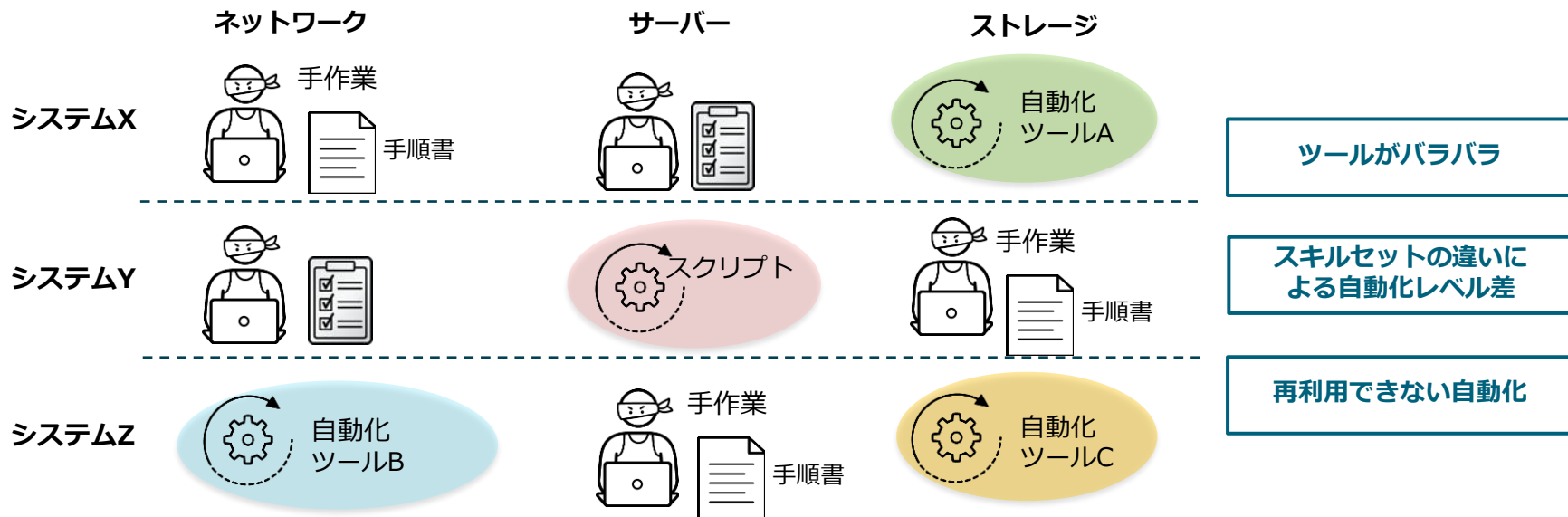
自動化 2.0 の世界

実態 - ほぼ全ての顧客は自動化1.0の状態

Automation Day ワークショップ結果より（4社の実例）



従来のインフラ管理が抱える課題（サイロ化）



自動化を進めても・・・

どの作業もあの人しか
わからない・できない

ブラックボックス化
ボトルネック化

横展開が不可能
標準化を阻害

全体の効率低下

自動化の効果が出ない。広がらない。本気で取り組まれない。

自動化で効果を出すための3つのポイント

ポイント1 自動化の標準化

- 自動化の対象ごとにバラバラの開発物、利用方法となっていた自動化を、標準化された自動化の開発と活用が可能な状態にする。
- 自動化2.0への移行難易度を下げる。

まずは『共通言語』

自動化1.0の課題を解決し
2.0への移行を加速する

ポイント2 自動化のサービス化

機能面特化のため一部のみ紹介

- 自動化を手順の置き換えではなく、サービスとして利用者に提供する。
- 作業に登場する「登場人物」を減らすことで、調整そのものが発生しない状態にする。

**作業をボタン化し
サービスとして提供**

自動化2.0を実現するためのアプローチ

ポイント3 インフラCI化

- 「人與人」の対話によって品質を上げてきた従来の方法から、「人とシステム」の対話へと切り替える。
- 事前の調整や前提の説明等の「対話のための状態同期コスト」を最適化する。

**インフラ運用担当者は
『サービス開発』へ**

Ansible Automation Platform の価値

～ 『対象外』を作らない強力かつ理解しやすいツール～

『標準化』に要求されるツールの性質

～ Ansible Automation Platform の特徴 ① ～



低い学習コスト

Simple

誰もが読める標準化
された自動化言語



例外なし

Powerful

多様な制御対象を
統一的手法で自動化

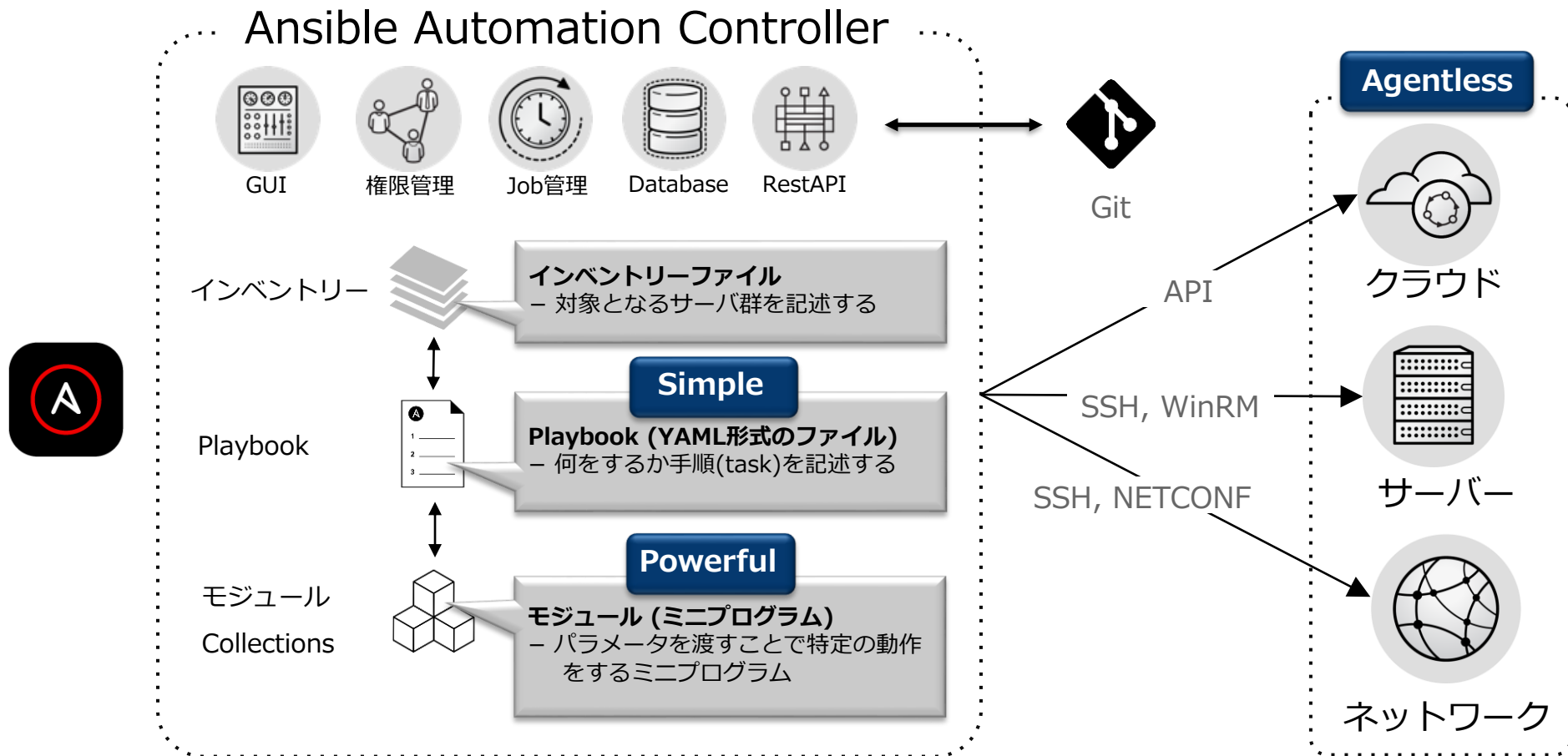


導入が容易

Agentless

追加も簡単
セキュリティ懸念無し

Ansible Automation Platform (AAP) 構成要素



AAP の特徴 - Simple ①

～ 標準化を促進する簡単な言語 ～



Ansibleプレイブック

実行順序



- name: **Apacheのインストールと起動**
hosts: app
become: yes

#Playbook の説明
#app グループが対象（インベントリ）
#権限昇格の有無（プラグインで提供）

tasks:

#実行する手順の内容
#実行時に処理毎に表示される名前

- name: **httpd のインストール**
yum:

name: httpd
state: latest

- name: **httpd を起動**

service:

name: httpd
state: running

TARGET
セクション

タスク
セクション

モジュール

AAP の特徴 - Simple ②

～ プラグイン (使い易さを高める拡張機能)～

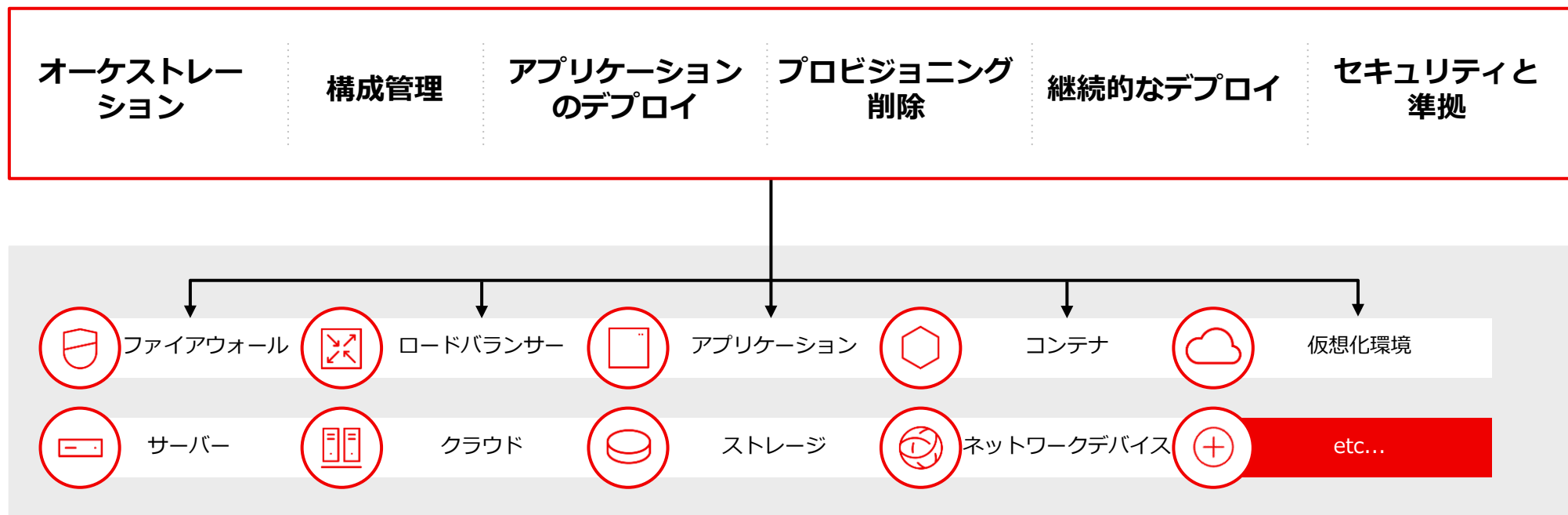
Ansible プラグイン

- Ansible のコア機能を拡張するコードの一部
- 外部ファイルとのアクセスや画面出力など便利な機能を提供



AAP の特徴 - Powerful ①

～ 自動化対象の IT プラットフォームの例 ～



AAP の特徴 - Powerful ②

～ 多くのIT機器を対象とした自動化が可能 ～

130+

認定コンテンツコレクション

55+

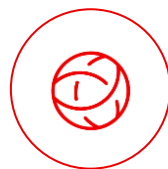
認定テクノロジーパートナー



Infrastructure



Cloud



Network



Security



Edge



AAP の特徴 - Powerful ③

～ モジュール詳細 ～

Ansible モジュール

- Ansible の中核的な機能
- Playbook のタスクの中で実行時に呼び出され対象ノードに作用する（動作イメージとしては Playbook と対象ノードを橋渡しするラッパーとして作用する）
- 通常 Python* (Windows では Powershell) で記述されるが基本的に言語の制限はない

*Playbook を書く際に Python を理解している必要はありません。
モジュールの使い方 (Playbook の書き方) を理解していれば大丈夫です。



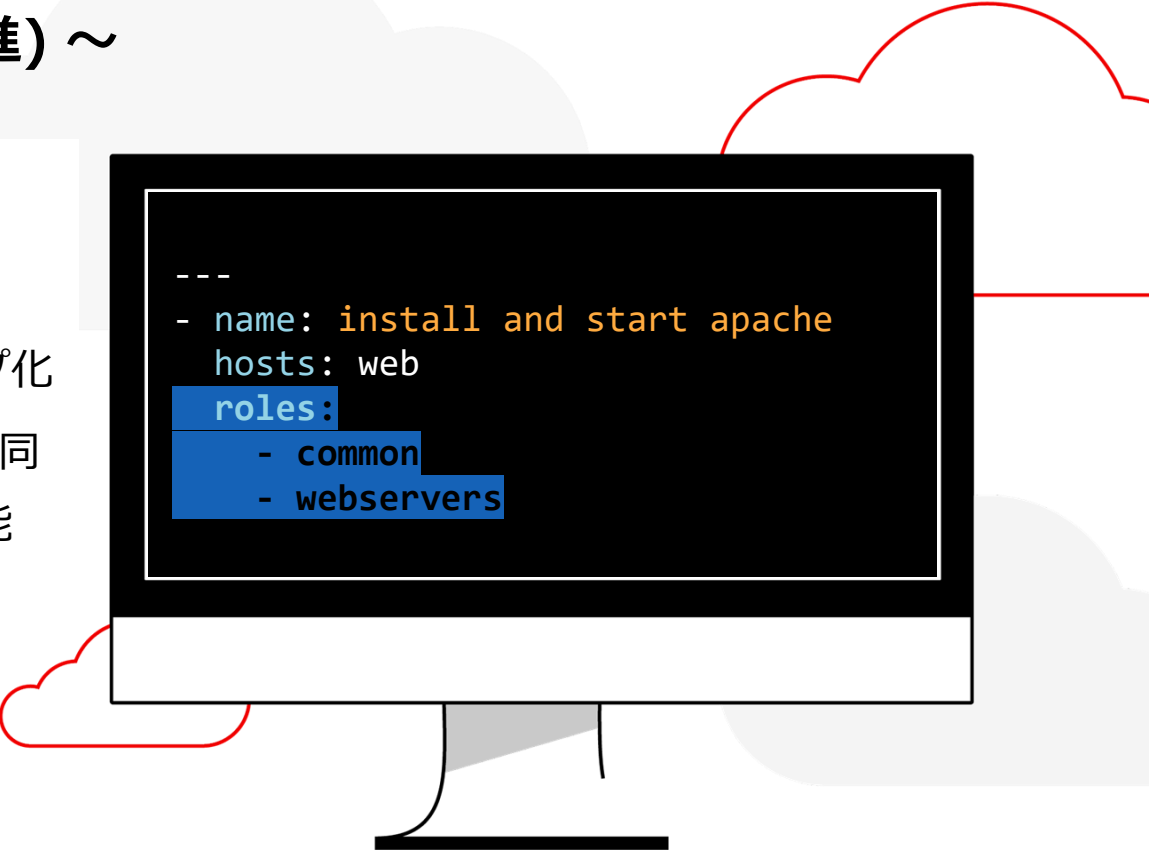
```
- name: latest index.html file ...  
  template:  
    src: files/index.html  
    dest: /var/www/html/
```

AAP の特徴 - Powerful ④

～ Ansible roles (自動化利用の促進) ～

Ansible roles

- ・ Playbook を再利用しやすい形に分解・定義
- ・ タスクと変数を再利用可能な構造にグループ化
- ・ roles 化しておくことで、自分だけではなく同じような自動化を行いたい他の人と共用可能



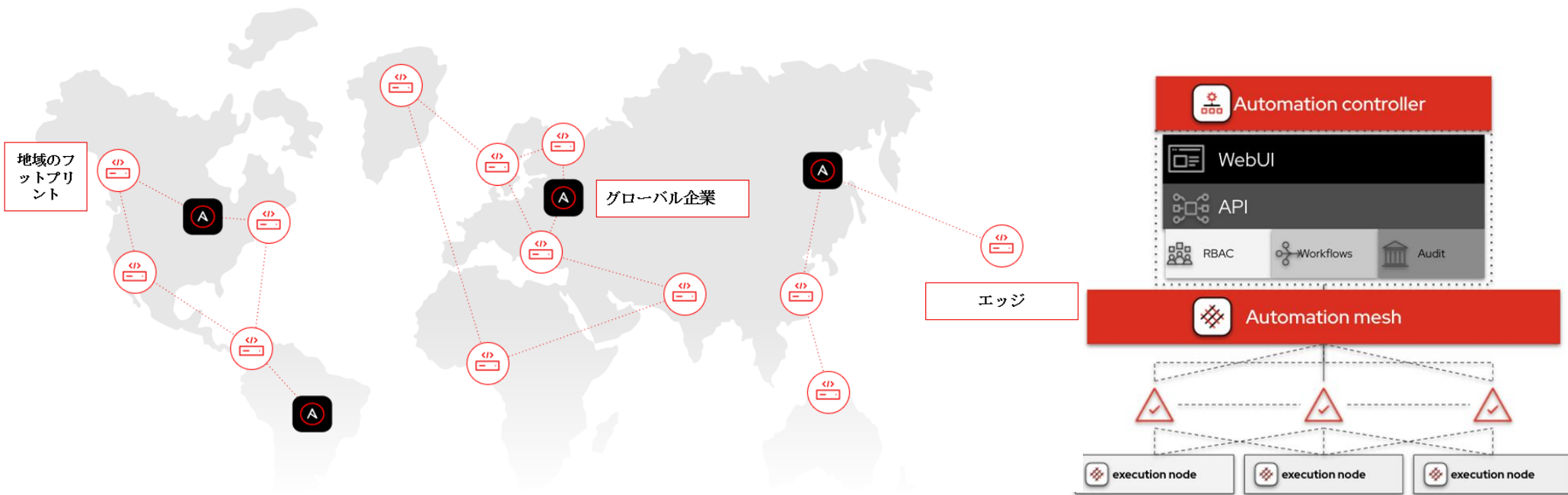
```
---  
- name: install and start apache  
  hosts: web  
  roles:  
    - common  
    - webservers
```

AAP の特徴 - Powerful ⑤

 AAP 2.1 ~

~ automation mesh

(分散型のモジュラー構造。分離されたネットワークにも柔軟に対応) ~



Execution environments
(コンテナ化された実行環境)

automation mesh
モジュール化により要な機能を必要な場所へ配置が可能に！

AAP の特徴 - Powerful ⑥

 AAP 2.1 ~

～ コンテナ化された実行環境の管理・運用・実装 ～

コンテンツ作成ツール



Ansible content tools



Execution environment builder



Automation content navigator



Ansible content collections



Execution
environments

運用ツール



Private automation hub



Automation controller



Automation mesh



Platform operators

ビジネス ツールと分析



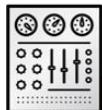
Red Hat Insights for Ansible



Automation services catalog

『サービス化』に要求されるツールの性質

～ Ansible Automation Platform の特徴② ～



サービスの作成

Interface

使い易い共通インター
フェイスで複雑な作業
を簡単にサービス化



サービスの連結

Control

ワークフローで
自動化サービスを連結



必要な人が実行

Delegation

必要な人に権限を委譲
必要な人が自動化サー
ビスを実行

AAP の特徴 - Automation Controller

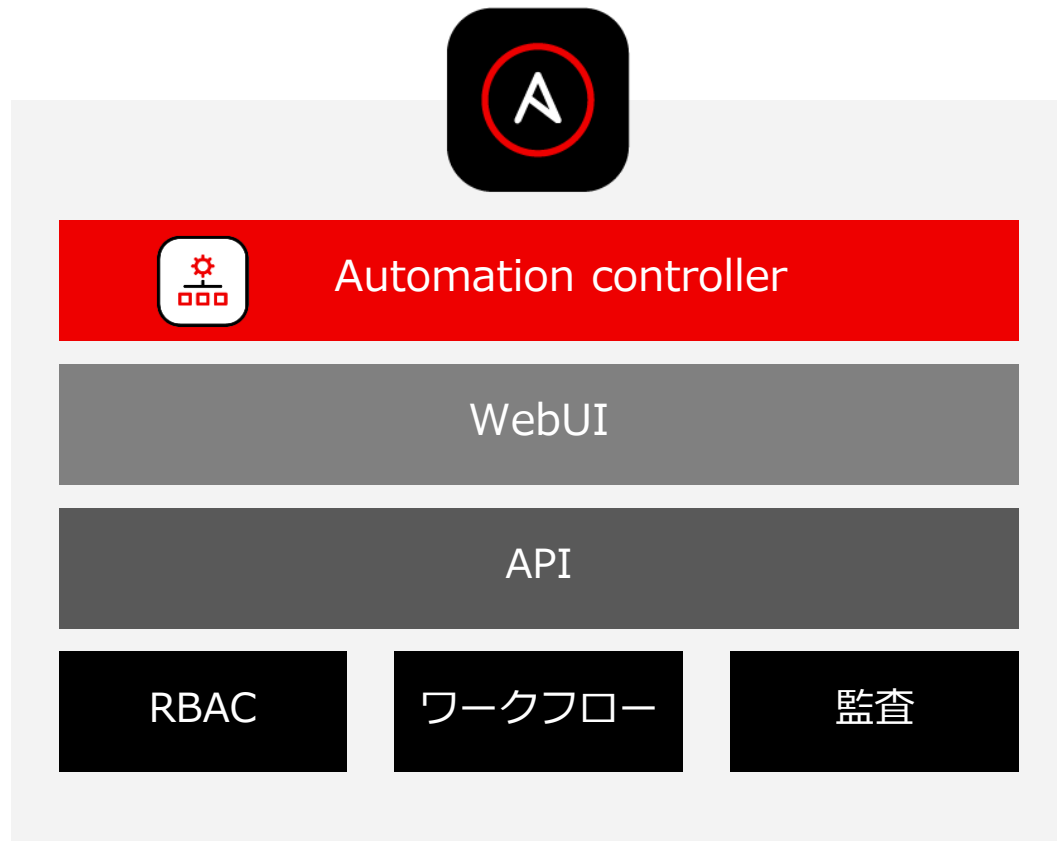
～ Ansible Automation Platform の特徴② ～

Automation controllerとは？

Automation Controllerは、ユーザーが企業全体で自動化を定義、操作、および委任できるようにする Ansible Automation Platform コントロールプレーンです

Automation Controllerは以下を提供します：

- ▶ WebUI と API
- ▶ ロールベースのアクセス制御
- ▶ 強力なワークフロー
- ▶ 集中ロギング
- ▶ クレデンシャル管理
- ▶ プッシュボタンによる自動化



AAP の特徴 - 使い易いインターフェース

洗練された GUI で誰でも簡単に操作できます



外部ソフトウェアとの連携
には RESTAPI 完備

```
REST API / バージョン 2 / Job Template List

Job Template List2

GET /api/v2/job_templates/

HTTP 200 OK
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept
X-API-Node: aap21-4.lab.local
X-API-Product-Name: Red Hat Ansible Automation Platform
X-API-Product-Version: 4.1.1
X-API-Time: 0.048s

{
  "count": 1,
  "next": null,
  "previous": null,
  "results": [
    {
      "id": 7,
      "type": "job_template",
      "url": "/api/v2/job_templates/7/",
      "related": {
        "created_by": "/api/v2/users/1/",
        "modified_by": "/api/v2/users/1/",
        "labels": "/api/v2/job_templates/7/labels/",
        "inventory": "/api/v2/inventories/1/",
        "project": "/api/v2/projects/6/",
        "organization": "/api/v2/organizations/1/",
        "credentials": "/api/v2/job_templates/7/credentials/",
        "jobs": "/api/v2/job_templates/7/jobs/",
        "schedules": "/api/v2/job_templates/7/schedules/",
        "activity_stream": "/api/v2/job_templates/7/activity_stream/",
        "launch": "/api/v2/job_templates/7/launch/",
        "webhook_key": "/api/v2/job_templates/7/webhook_key/",
        "webhook_receiver": "",
        "notification_templates_started": "/api/v2/job_templates/7/notifi
```

CLI も充実！！ Controller CLI (AWX) コマンド！

awx job_templates launch <id>

<https://docs.ansible.com/automation-controller/latest/html/controllercli/usage.html>

AAP の特徴 - 使用権限を委譲（プロジェクト）

“プロジェクト” による Playbook ディレクトリの管理と権限の委譲

Playbook ディレクトリ： /var/lib/awx/projects/<各プロジェクト>/

利用オーナーを決めて利用権限を委譲する！！

```
[root@ansible projects]# pwd
/var/lib/awx/projects
[root@ansible projects]# tree
-- apps
-- aws
-- network
-- vmware
```

App 用 Playbook

AWS 用 Playbook

Network 用 Playbook

VMware 用 Playbook

Red Hat Ansible Automation Platform

プロジェクト > App-Manager
詳細の編集

名前 * App-Manager 説明 組織 * App-Org

実行環境 ⓘ Default execution environment ソースコントロール認証情報タイプ * 手動

タイプの詳細

プロジェクトのベースパス ⓘ Playbook ディレクトリー * ⓘ

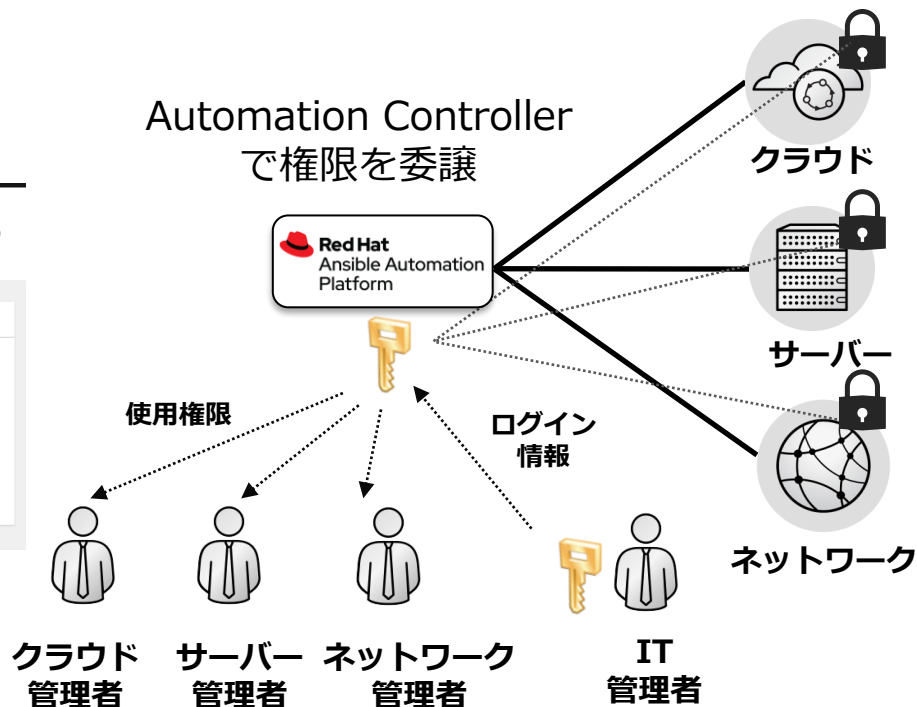
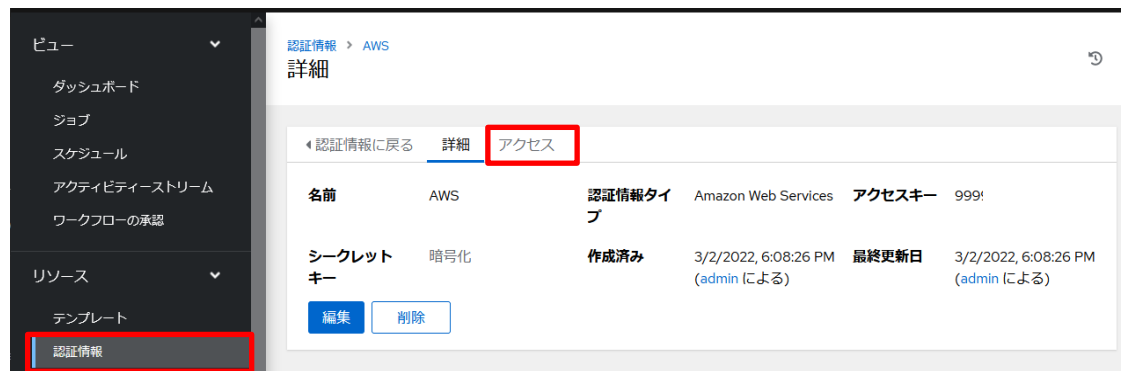
/var/lib/awx/projects apps

保存 取り消し

AAP の特徴 - 使用権限を委譲（認証情報）

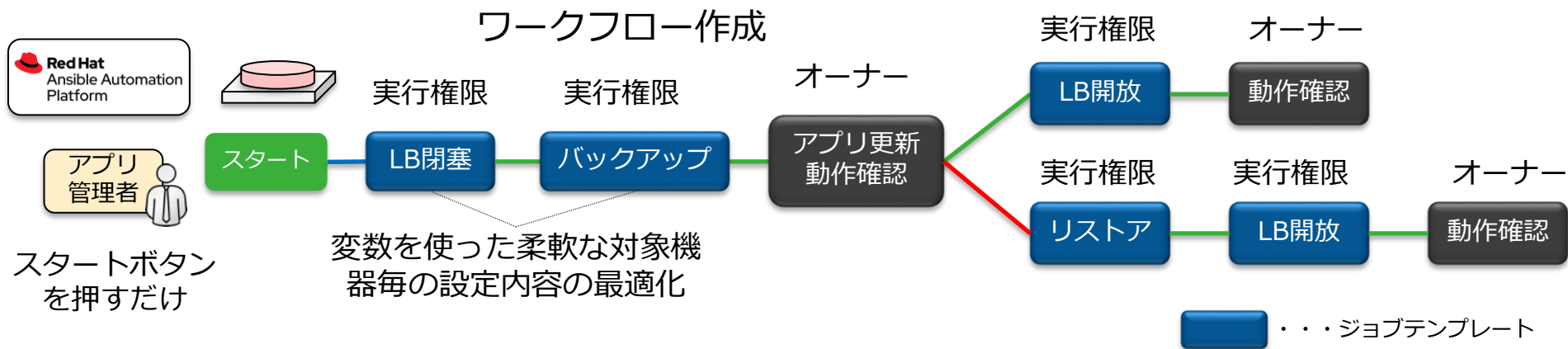
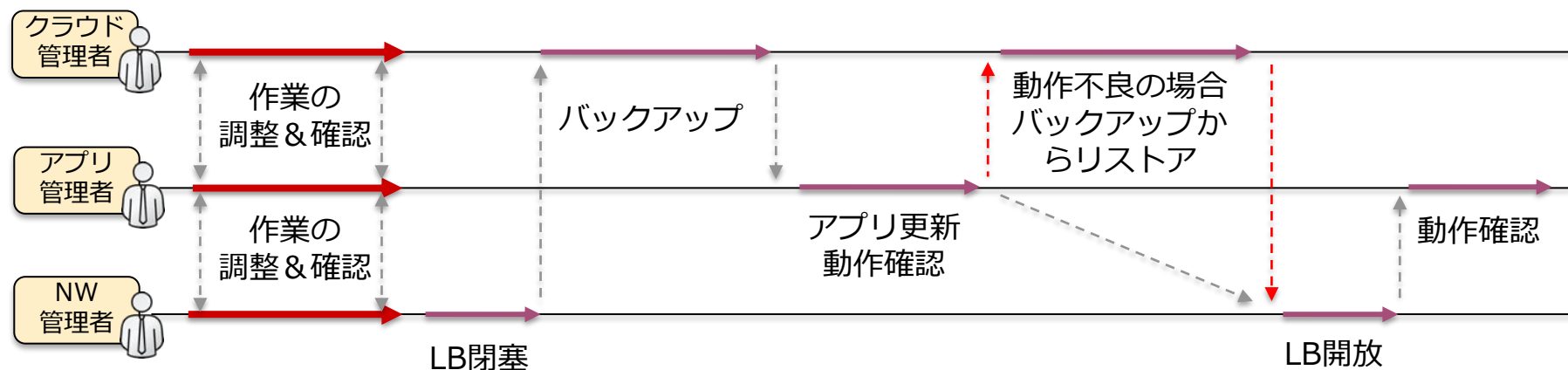
管理対象ノードに対するアクセス権を一括管理

必要な人に必要な権限を委譲する



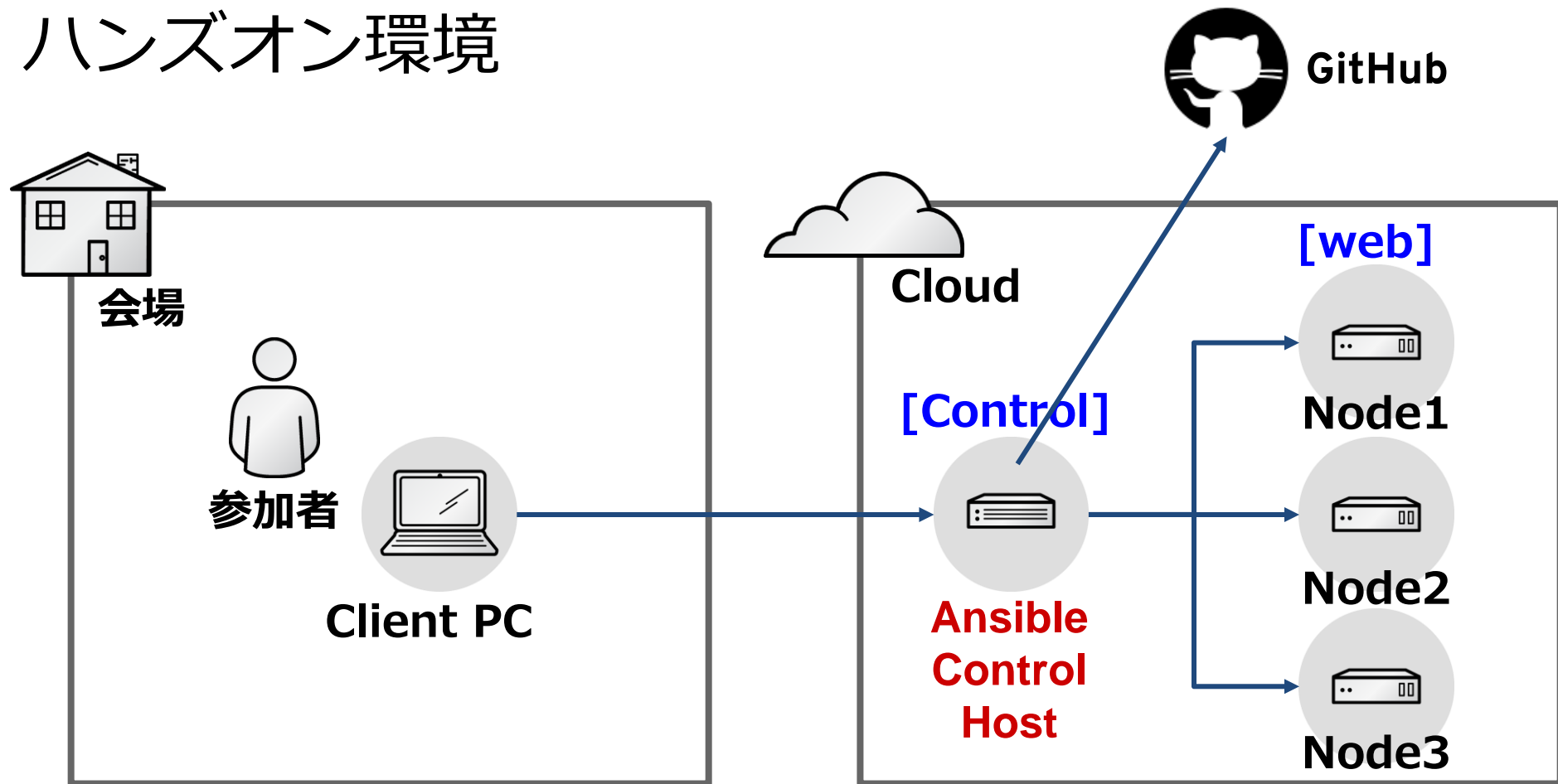
※一例としてプロジェクトと認証情報のみ例を出しましたが『テンプレート』や『インベントリ』など様々なオブジェクトに対し権限の委譲が可能です。

AAP の特徴 - ワークフローでサービスを連結



Command-line Ansible 演習

ハンズオン環境



1人1環境 (Control Host 1台 & Web Node 3台) をクラウド上に用意

ハンズオン - Ansible コマンドラインの演習

1.1 - 前提条件の確認

1.2 - Ansible の基本

1.3 - 初めての Playbook の作成

1.4 - 変数の使用

1.5 - 条件、ハンドラー、ループ

1.6 - テンプレート

1.7 - Roles

Playbook の例

ansible-playbookコマンドの実行

\$ ansible-playbook -i inventory_file playbook.yml

TARGET
セクション

```
---  
- name: Apache server installed  
  hosts: web  
  become: yes
```

Playbookの説明
対象ホストの指定 (インベントリの中より)
権限昇格の有無

TASKS
セクション

```
  tasks:  
    - name: latest Apache version installed  
      yum:  
        name: httpd  
        state: latest  
    - name: Apache enabled and running  
      service:  
        name: httpd  
        enabled: true  
        state: started  
    - name: copy index.html  
      copy:  
        src: ~/ansible-files/index.html  
        dest: /var/www/html/
```

Task の説明
利用モジュールを宣言
httpd のインストール

httpdサービスの開始と有効化

index.html ファイルのコピー
 (Ansibleホスト→対象ノード)

実行順序

モジュール

Inventory ファイルの例

- 管理対象サーバを記述

- ホスト名
- IPアドレス
- ssh のユーザ名

- グループ化できる

- 変数の指定も可能

- ansible-playbook コマンドの `-i` オプションで指定する

```
[web]
web-1.example.com
web-2.example.com
web-3.example.com
```

グループ

```
[app]
app-1.example.com
app-2.example.com
```

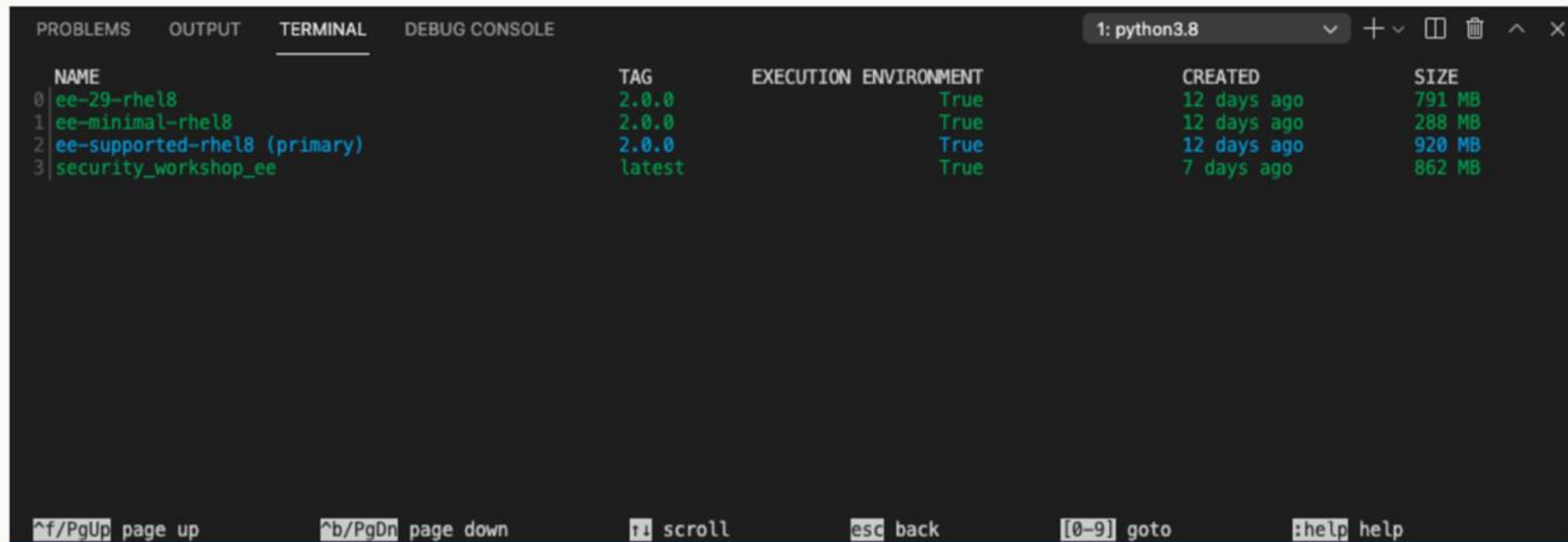
```
[db]
db-1.example.com
```

演習 1.1 - 前提条件の確認

それでは演習をやってみましょう！

ssh 環境にアクセス、ansible-navigatorで操作環境を閲覧します。

```
$ ansible-navigator images
```

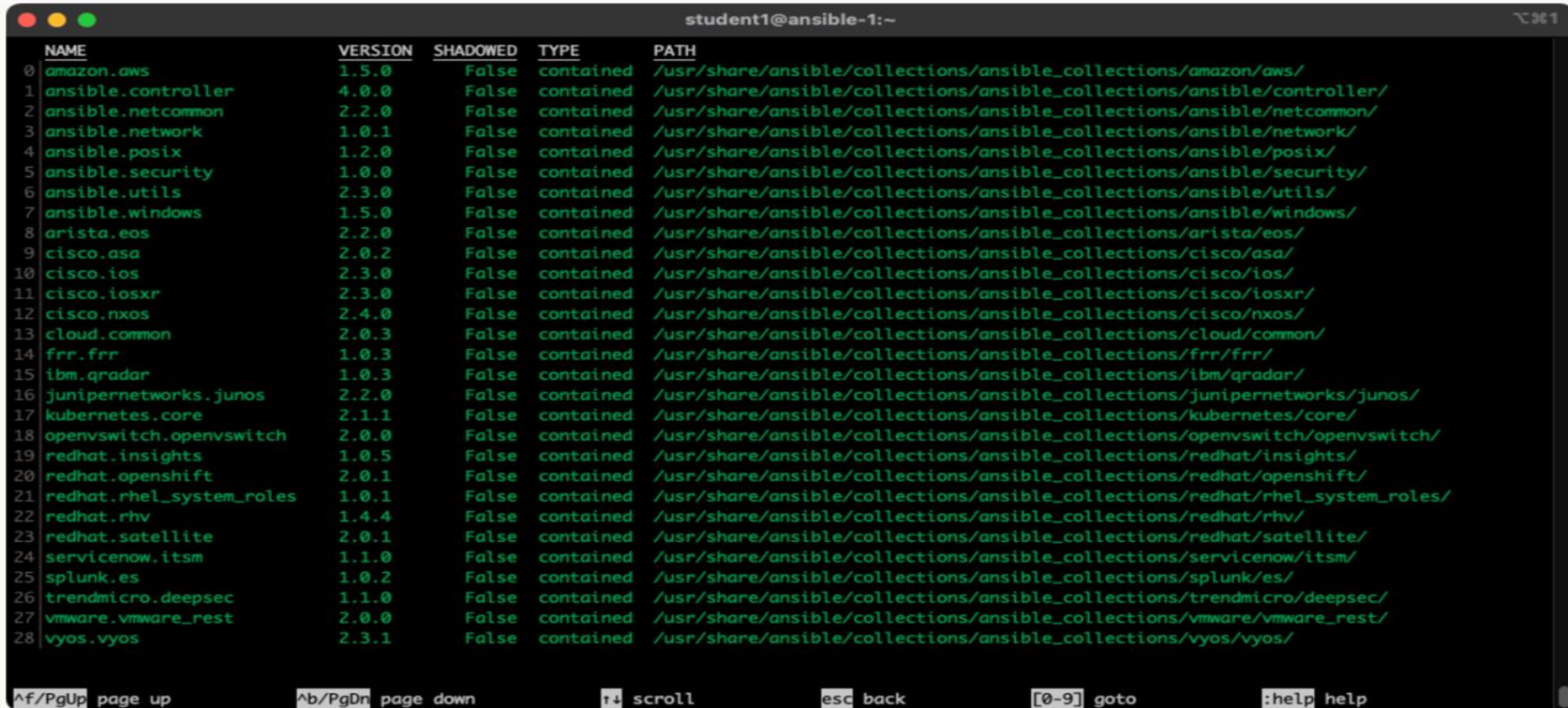


	NAME	TAG	EXECUTION ENVIRONMENT	CREATED	SIZE
0	ee-29-rhel8	2.0.0	True	12 days ago	791 MB
1	ee-minimal-rhel8	2.0.0	True	12 days ago	288 MB
2	ee-supported-rhel8 (primary)	2.0.0	True	12 days ago	920 MB
3	security_workshop_ee	latest	True	7 days ago	862 MB

演習 1.2 - Ansible の基本

ansible-navigatorを利用したInventoryの確認、Execution Environmentで特定のモジュール一覧や、コレクション利用方法などを参照します。

```
$ :collections
```



The screenshot shows the 'collections' view in the ansible-navigator application. The interface has a dark theme. At the top, the title bar says 'student1@ansible-1:~'. Below it, a table lists various Ansible collections. The table has five columns: NAME, VERSION, SHADOWED, TYPE, and PATH. The collections listed include amazon.aws, ansible.controller, ansible.netcommon, ansible.network, ansible.posix, ansible.security, ansible.utils, ansible.windows, arista.eos, cisco.asa, cisco.ios, cisco.iosxr, cisco.nxos, cloud.common, frr.frr, ibm.qradar, junipernetworks.junos, kubernetes.core, openvswitch.openvswitch, redhat.insights, redhat.openshift, redhat.rhel_system_roles, redhat.rhv, redhat.satellite, servicenow.itsm, splunk.es, trendmicro.deepsec, vmware.vmware_rest, and vyos.vyos. All collections are marked as 'contained' and 'False' for shadowed. The bottom of the window shows navigation controls like 'page up', 'page down', 'scroll', 'back', 'goto', and 'help'.

	NAME	VERSION	SHADOWED	TYPE	PATH
0	amazon.aws	1.5.0	False	contained	/usr/share/ansible/collections/ansible_collections/amazon/aws/
1	ansible.controller	4.0.0	False	contained	/usr/share/ansible/collections/ansible_collections/ansible/controller/
2	ansible.netcommon	2.2.0	False	contained	/usr/share/ansible/collections/ansible_collections/ansible/netcommon/
3	ansible.network	1.0.1	False	contained	/usr/share/ansible/collections/ansible_collections/ansible/network/
4	ansible.posix	1.2.0	False	contained	/usr/share/ansible/collections/ansible_collections/ansible/posix/
5	ansible.security	1.0.0	False	contained	/usr/share/ansible/collections/ansible_collections/ansible/security/
6	ansible.utils	2.3.0	False	contained	/usr/share/ansible/collections/ansible_collections/ansible/utils/
7	ansible.windows	1.5.0	False	contained	/usr/share/ansible/collections/ansible_collections/ansible/windows/
8	arista.eos	2.2.0	False	contained	/usr/share/ansible/collections/ansible_collections/arista/eos/
9	cisco.asa	2.0.2	False	contained	/usr/share/ansible/collections/ansible_collections/cisco/asa/
10	cisco.ios	2.3.0	False	contained	/usr/share/ansible/collections/ansible_collections/cisco/ios/
11	cisco.iosxr	2.3.0	False	contained	/usr/share/ansible/collections/ansible_collections/cisco/iosxr/
12	cisco.nxos	2.4.0	False	contained	/usr/share/ansible/collections/ansible_collections/cisco/nxos/
13	cloud.common	2.0.3	False	contained	/usr/share/ansible/collections/ansible_collections/cloud/common/
14	frr.frr	1.0.3	False	contained	/usr/share/ansible/collections/ansible_collections/frr/frr/
15	ibm.qradar	1.0.3	False	contained	/usr/share/ansible/collections/ansible_collections/ibm/qradar/
16	junipernetworks.junos	2.2.0	False	contained	/usr/share/ansible/collections/ansible_collections/junipernetworks/junos/
17	kubernetes.core	2.1.1	False	contained	/usr/share/ansible/collections/ansible_collections/kubernetes/core/
18	openvswitch.openvswitch	2.0.0	False	contained	/usr/share/ansible/collections/ansible_collections/openvswitch/openvswitch/
19	redhat.insights	1.0.5	False	contained	/usr/share/ansible/collections/ansible_collections/redhat/insights/
20	redhat.openshift	2.0.1	False	contained	/usr/share/ansible/collections/ansible_collections/redhat/openshift/
21	redhat.rhel_system_roles	1.0.1	False	contained	/usr/share/ansible/collections/ansible_collections/redhat/rhel_system_roles/
22	redhat.rhv	1.4.4	False	contained	/usr/share/ansible/collections/ansible_collections/redhat/rhv/
23	redhat.satellite	2.0.1	False	contained	/usr/share/ansible/collections/ansible_collections/redhat/satellite/
24	servicenow.itsm	1.1.0	False	contained	/usr/share/ansible/collections/ansible_collections/servicenow/itsm/
25	splunk.es	1.0.2	False	contained	/usr/share/ansible/collections/ansible_collections/splunk/es/
26	trendmicro.deepsec	1.1.0	False	contained	/usr/share/ansible/collections/ansible_collections/trendmicro/deepsec/
27	vmware.vmware_rest	2.0.0	False	contained	/usr/share/ansible/collections/ansible_collections/vmware/vmware_rest/
28	vyos.vyos	2.3.1	False	contained	/usr/share/ansible/collections/ansible_collections/vyos/vyos/

演習 1.3 - 初めての Playbook の作成

```
---  
- name: Apache server installed  
  hosts: web  
  become: yes  
  tasks:  
    - name: latest Apache version installed  
      yum:  
        name: httpd  
        state: latest  
    - name: Apache enabled and running  
      service:  
        name: httpd  
        enabled: true  
        state: started  
    - name: copy index.html  
      copy:  
        src: ~/ansible-files/index.html  
        dest: /var/www/html/
```

対象ノードや権限昇格、変数などTask
を実行するための条件を記述する領域

タスク と呼ばれる部分
具体的な実行内容をモジュールとともに
記述する部分
ここでは、yum モジュール、service モ
ジュール、copy モジュールがそれぞれ
オプションとともに記述されています。

コメント:

演習のplaybook作成はviベースでガイドされてますが、
vscodeで進めていただくのが簡単です。
vscode画面「Terminal→New Terminal」でコマンドライン
を呼び出し利用できます。

演習 1.3 - 初めての Playbook の作成

- name: Apache server installed

hosts: web

become: yes

tasks:

- name: latest Apache version installed

yum:

name: httpd

state: latest

- name: Apache enabled and running

service:

name: httpd

enabled: true

state: started

- name: copy index.html

copy:

src: ~/ansible-files/index.html

dest: /var/www/html/

スペース2個分あける！

スペース4個分あける！

スペース6個分あける！

プレイブックはわかりやすいのですがお作法にうるさいです。

特に先頭のスペースには、よく注意を払って書いてください！！

※ tab を使ってはいけません！

演習 1.4 - 変数の使用

Ansible のディレクトリ構造を利用した便利な変数 "**{{xxx}}**" の定義

- ~/ansible-files/deploy_index_html.yml ---> プレイブック実行ディレクトリ
- ~/ansible-files/**group_vars/web**・・・グループ [web] に対する変数値を定義
- ~/ansible-files/**host_vars/node2**・・・ホスト "node2" に対する変数の値を定義

deploy_index_html.yml

グループwebでは
stage 変数に
dev を入力

/group_vars/web

stage: dev

ノード "node2" で
は、stage 変数に
prod を入力

/host_vars/node2

stage: prod

```
---
- name: Copy index.html
  hosts: web
  become: yes
  tasks:
    - name: copy index.html
      copy:
        src: ~/ansible-files/{{ stage }}_index.html
        dest: /var/www/html/index.html
```

優先順位 host_vars > group_vars

コメント：
groupに対する変数、ホストに対する変数だとhost
が優先される点に注意！変数は複数設定が可能など
ころが便利ではありますが複雑になる場合も。

演習 1.4 変数 - その他定義場所と優先順位

group_vars / host_vars 以外にも変数は様々なところに定義できます。
その際の優先順位があります。

- | | |
|---|----------------------------------|
| 1. extra vars (-e 指定で実行) | 9. registered vars |
| 2. task vars (only for the task) | 10. host facts |
| 3. block vars (only for tasks in block) | 11. host_vars (playbook) |
| 4. role and include vars | 12. group_vars (playbook) |
| 5. play vars_files | 13. host_vars (inventory) |
| 6. play vars_prompt | 14. group_vars (inventory) |
| 7. play vars | 15. inventory vars |
| 8. set_facts | 16. role defaults |

演習 1.4 Ansible ファクト

Ansible を使っていると、ファクトという名前が良く出てきます。これは対象ノードのプロパティ情報を定義、表示、さらにプレイブックから利用可能にするためのものです。Ansible が非常に強力なところは、取得された膨大なファクト情報を自動的に、`ansible_xxx`という変数に入れてくれるところです。例えば、ファクトの値を元に特定のホストにだけプレイブックを実行！なども可能です。

```
$ ansible localhost -m setup
localhost | success >> {
  "ansible_facts": {
    "ansible_default_ipv4": {
      "address": "192.168.1.37",
      "alias": "wlan0",
      "gateway": "192.168.1.1",
      "interface": "wlan0",
      "macaddress": "c4:85:08:3b:a9:16",
      "mtu": 1500,
      "netmask": "255.255.255.0",
      "network": "192.168.1.0",
      "type": "ether"
    }
  },
}
```

例えば右の例では...

`ansible_default_ipv4.address`

192.168.1.37

`ansible_default_ipv4.gateway`

192.168.1.1

が自動的に定義され、取得可能です

※収集される変数全体は以下のコマンドで確認可能です。

`$ ansible <host/group_name> -m setup`

演習 1.5 - 条件、ハンドラー、ループ

- 特定のタスクのみ実行 (tag)
- 繰り返し (loop, with_item, with_nested, until ...)
- 条件分岐 (when, notify/handlers ...)
- ファイル操作 (copy, template)
- 他のplaybookの読み込み (include, role, ...)
- 外部情報の参照
 - 環境変数、ファイル など (environment, lookup, vars_prompt,...)
- カスタムモジュールを書いて拡張も可能

演習 1.5 条件分岐 - when

```
---  
- name: Install vsftpd on ftpservers  
  hosts: all  
  become: yes  
  tasks:  
    - name: Install FTP server when host in ftpserver group  
      yum:  
        name: vsftpd  
        state: latest  
        when: inventory_hostname in groups["ftpserver"]
```

hosts: all ですが、when の条件により、グループ "ftpserver" に属するものだけにこのタスクを実行

演習 1.5 ハンドラー - notify/handles

```
---
- name: manage httpd.conf
  hosts: web
  become: yes
  tasks:
    - name: Copy Apache configuration file
      copy:
        src: httpd.conf
        dest: /etc/httpd/conf/
      notify:
        - restart_apache
  handlers:
    - name: restart_apache
      service:
        name: httpd
        state: restarted
```

この場合は、ソースとターゲットの httpd.conf が異なるときに限り、handlers が呼び出されて httpd サービスのリスタートが実施されます。

notify が記述されたタスクが実行された時だけ、handlers に記述された内容が実行される。

演習 1.5 繰り返し - loop

```
---  
- name: Ensure users  
  hosts: node1  
  become: yes  
  
  tasks:  
    - name: Ensure three users are present  
      user:  
        name: "{{ item }}"  
        state: present  
      loop:  
        - dev_user  
        - qa_user  
        - prod_user
```

この場合は、user モジュールにより、

1回目 : dev_user

2回目 : qa_user

3回目 : prod_user

が作成されます。

loop に指定された項目が順に item 変数に入力され繰り返し実行されます。

演習 1.6 - テンプレート

テンプレートモジュールは、変数を含むテンプレートファイルに変数の値を入力した上で対象ホストにファイルコピーを実施します。

motd-facts.j2

```
Welcome to {{ ansible_hostname }}.  
{{ ansible_distribution }} {{ ansible_distribution_version }}  
deployed on {{ ansible_architecture }} architecture.
```

```
---  
- name: Fill motd file with host data  
  hosts: node1  
  become: yes  
  tasks:  
    - template:  
      src: motd-facts.j2  
      dest: /etc/motd  
      owner: root  
      group: root  
      mode: 0644
```

テンプレート内にある4つの変数に、gather_factsで収集された変数が入力され、対象ホスト node1 の /etc/motd ファイルとしてコピーされます。

演習 1.7 - Roles

Role は、プレイブックを複数連携して実行しやすくするための仕組みを提供します。Ansible では1つのプレイブックから他の複数のプレイブックを実行することができますが、この1つ1つをロールと言います。

Roles ディレクトリ構造

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

メインプレイブック

```
# site.yml
---
- hosts: web
  roles:
    - common
    - apache
```

Role はメインのプレイブックから読みだして使います。

tasks/ 配下にメインタスクが入っており、defaults/、vars/ の変数などを使いながら実行されます。このとき、ディレクトリに関する配慮は不要です。

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.