Ansible 2
Infrastructure as Code

Cześć!

Prowadzący: Jakub Andrzejewski

DevOps Engineer @ El Passion

Autor prezentacji: Daniel Kossakowski

1.
Includes

Includes? A co to?

- Podział tasków na mniejsze pliki
- Plik główny zawsze nazywa się main.yml
- Dwie metody:
  - import_tasks - kod analizowany przed uruchomieniem playbooka
  - include_tasks - kod analizowany w trakcie uruchomienia plabooka

Includes? Przykład

```
$ cat dev-host/tasks/main.yml
---
  - import_tasks: packages.yml
  - import_tasks: mounts.yml
  - import_tasks: dirs.yml
```

Includes? Przykład

```
$ cat dev-host/tasks/dirs.yml
---
- name: Prepare sites dirs
  action: file path={{ item }} state=directory owner=admin group=admin mode=0750
  with_items:
        - /opt/sites/{{ inventory_hostname }}
        - /opt/backups/{{ inventory_hostname }}

- name: Set admin user as owner of backup dir
  file:
        path: /opt/backups/{{ inventory_hostname }}
        state: directory
        owner: admin
        group: admin
        recurse: yes
```

2.
Warunki

Warunki? A co to?

- Dodanie dynamiki do kodu ansible
- Warunki uruchamiają różne kody w zależności od zmiennych
- Źródła zmiennych:
  - Zmienne zdefiniowane przez nas
  - Facty pobrane podczas uruchomienia

- Dokumentacja: https://docs.ansible.com/ansible/latest/user_guide/playbooks_conditionals.html

Warunki? Lista factów

```
$ cat roles/role-name/tasks/main.yml
---
- name: Show facts available on the system
  ansible.builtin.debug:
    var: ansible_facts
```

Warunki? Lista factów

```
$ ansible -m ansible.builtin.setup -i hosts all
88.198.33.133 | SUCCESS => {
        "ansible_facts": {
        "ansible_all_ipv4_addresses": [
        "172.17.0.1",
        ],
        "ansible_all_ipv6_addresses": [
        "fe80::42:c9ff:fe70:3a0f",
        "2a01:4f8:a0:300c::2",
        ],
        "ansible_distribution": "Ubuntu",
        "ansible_distribution_major_version": "20",
        "ansible_distribution_release": "focal",
        "ansible_distribution_version": "20.04",
        "ansible_machine": "x86_64",
        "ansible_os_family": "Debian",
```

Warunki? Przykład

- Do dowolnego taska można dodać warunek
- Wywołanie: when
- Sprawdzenie czy zmienna jest obecna
- Sprawdzenie i porównanie wartości zmiennej
- Pobranie wartości z innego taska i przyrównanie

Warunki? Przykład

```
$ cat ./os-base/tasks/main.yml
---
- import_tasks: os-packages-apt.yml
  when: ansible_os_family == "Debian"

- import_tasks: os-packages-yum.yml
  when: ansible_os_family == "RedHat"
```

Warunki? Przykład

```
tasks:
  - name: Shut down CentOS 6 systems
    ansible.builtin.command: /sbin/shutdown -t now
    when:
      - ansible_facts['distribution'] == "CentOS"
      - ansible_facts['distribution_major_version'] == "6"
```

Warunki? Operatory logiczne

- and
- or
- not
- tablica warunów = and
- nawiasy do grupowania

Warunki? Przykład

```
tasks:
  - name: Shut down CentOS 6 and Debian 7 systems
    ansible.builtin.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")
```

Warunki? Przykład

```
vars:
  epic: true
  monumental: "yes"

tasks:
    - name: Run the command if "foo" is defined
      ansible.builtin.shell: echo "I've got '{{ foo }}' and am not afraid to use it!"
      when: foo is defined

    - name: Fail if "bar" is undefined
      ansible.builtin.fail: msg="Bailing out. This play requires 'bar'"
      when: bar is undefined
```

Warunki? Rejestrowanie zmiennych

- Nie wszystkie informacje są w factach
- Ansible pozwala zarejestrować wynik taska do zmiennej
- Zmienna zawiera obiekt
- Budowa zależna od rodzaju taska (modułu)
- Można śledzić też stan tasków:
  - succeeded
  - failed
  - skipped

Warunki? Rejestrowanie zmiennych

```
- name: Check whether solr directory already exists
  stat: path="/opt/sites/{{ target_server }}/{{ site_name }}/build/configs/solr"
  register: solr

- name: Set owner for solr
  file:
        path: /opt/sites/{{ target_server }}/{{ site_name }}/build/configs/solr
        state: directory
        owner: 8983
        group: 8983
        recurse: yes
        follow: no
  when: solr.stat.isdir is defined and solr.stat.isdir
```

Warunki? Rejestrowanie zmiennych

```
tasks:
  - name: Register a variable, ignore errors and continue
    ansible.builtin.command: /bin/false
    register: result
    ignore_errors: true

  - name: Run only if the task that registered the "result" variable fails
    ansible.builtin.command: /bin/something
    when: result is failed

  - name: Run only if the task that registered the "result" variable succeeds
    ansible.builtin.command: /bin/something_else
    when: result is succeeded

  - name: Run only if the task that registered the "result" variable is skipped
    ansible.builtin.command: /bin/still/something_else
    when: result is skipped
```

3.
Szablony

Szablony? A co to?

- Nie wszystkie pliki zawsze wyglądają tak samo
- Pliki generowane na podstawie szablonów Jinja2
- Moduł: ansible.builtin.template
- Szablony przechowujemy w roli w folderze *templates*
- Do każdego pliku można określić:
  - Źródło szablonu
  - Ścieżka do pliku
  - Uprawnienia
  - Właścicieli (owner, group)

Szablony? Przykład

```
- name: Template a file to /etc/file.conf
  ansible.builtin.template:
    src: foo.j2                    # nazwa pliku w folderze templates
    dest: /etc/nginx/nginx.conf    # nazwa pliku i ścieżka, który zostanie utworzony
    owner: root                    # właściciel pliku
    group: wheel                   # nazwa grupy
    mode: '0644’                   # uprawnienia dostępu
```

Szablony? Przykład

```
#
# ! This file is managed by ansible, do not modify it manually  !
# ! Sources are available at: bitbucket.org/rentuu/cicd-ansible !
#

server {
  server_name admin.dev.{{ domain }};
  listen              80;

  return 301 https://$host$request_uri;
}
```

Szablony? Przekazywanie zmiennych

- Do każdego szablonu przekazywane są zmienne
- Źródła zmiennych takie same jak przy warunkach
- Wywołanie poprzez umieszczenie nazwy pomiędzy blokami {{ i }}
- Przykład:
  - {{ domain }}
  - server_name admin.dev.{{ domain }};

Szablony? Przekazywanie zmiennych

```
#
# ! This file is managed by ansible, do not modify it manually  !
# ! Sources are available at: bitbucket.org/rentuu/cicd-ansible !
#

server {
  server_name admin.dev.{{ domain }};
  listen              80;

  return 301 https://$host$request_uri;
}
```

Szablony? Pętle

- Oprócz zmiennych tekstowych można przekazać tablice
- Po tablicach można iterować
- Przykład bloku pętli for:
  - {% for tenant in tenants %}
  - {% endfor %}

Szablony? Pętle

```
{% for tenant in tenants %}

# {{ tenant }} - start
location ^~ /{{ tenant }}/404 {
        error_page 404 /{{ tenant }}/index.html;
}
location ^~ /{{ tenant }} {
        alias /home/cicd/tenants/{{ tenant }}/admin-app;
        try_files $uri $uri/ /index.html =404;

        access_log /home/cicd/logs/nginx/{{ tenant }}/admin-app.access.log;
        error_log     /home/cicd/logs/nginx/{{ tenant }}/admin-app.error.log;
}
# {{ tenant }} - end
{% endfor %}
```

Szablony? Warunki

- Przykład bloku if:
  - {% if %}
  - {% else %}
  - {% endif %}

Szablony? Pętle

```
#
# ! This file is managed by ansible, do not modify it manually  !
# ! Sources are available at: bitbucket.org/rentuu/cicd-ansible !
#

{% if someTest %}
    <p> Something is True </p>
{% else %}
    <p> Something is False </p>
{% endif %}
```

4.
Handlers

Handlers? A co to?

- Handler to akcja wykonywana po zmianie
- Plik main.yml w folderze handlers
- Handler nie uruchamia się sam, trzeba go wywołać (notify)
- Uruchamiany za każdym razem gdy wystąpi zmiana

Handlers? Przykład

```
$ cat roles/nginx/handlers/main.yaml
- - -

  - name: "restart supervisor"
          service:
          name:  "supervisor"
          state: "restarted"

  - name: "restart nginx"
          service:
          name:  "nginx"
          state: "restarted"
```

Handlers? Przykład

```
- name: "Create laravel worker for all tenants"
  template:
    src:   "laravel-worker.conf.j2"
    dest:  "/home/cicd/conf.d/supervisor/laravel-worker-{{ tenant }}.conf"
    owner: "cicd"
    group: "cicd"
    mode:  0644
  loop: "{{ tenants }}"              # Wykonanie taska dla każdego elementu w tablicy
  Loop_control:                      # Sekcja dodatkowych ustawień dla pętli
    loop_var: "tenant"               # Określenie nazwy zmiennej z pojedynczym elementem
  notify:
        - "restart supervisor"       # Wywołanie handlera
```

5.
Tagi

Tagi? A co to?

- Granulacja kodu
- Tagi pozwalają określić miejsca wywołań kodu
- Duży kod warto tagować
- Czasami chcemy uruchomić kawałek playbooka, zamiast całego
- Do jednego elementu można przypisać kilka tagów
- Proces:
  - Dodanie tagów do kodu
  - Wywołanie ansible z konkretnym tagiem lub flagą skip_tags

Tagi? Przypisywanie

- Do pojedynczego taska
- Do sekcji include_tasks i import_tasks
- Do całego playbooka
- Do ról

Tagi? Przykład

```
tasks:
- name: Install the servers
  ansible.builtin.yum:
    name:
    - httpd
    - memcached
    state: present
  tags:
  - packages
  - webservers

- name: Configure the service
  ansible.builtin.template:
    src: templates/src.j2
    dest: /etc/foo.conf
  tags:
  - configuration
```

Tagi? Wywołanie

- --tags all - uruchom wszystkie taski (domyślne zachowanie)
- --tags [tag1, tag2] - uruchom tylko taski z podanych tagów
- --skip-tags [tag3, tag4] - uruchom wszystkie taski, bez podanych tagów
- --tags tagged - uruchom taski z tagami (nazwa obojętna)
- --tags untagged - uruchom taski bez tagów

6.
Ansible Galaxy

Galaxy? A co to?

- Repozytorium z modułami community
- Oprócz modułów również role
- Możliwość umieszczenia paczek w pliku requirements.yaml
- URL: https://galaxy.ansible.com/

Galaxy? Przykładowe moduły

- Kubernetes - https://galaxy.ansible.com/community/kubernetes
- AWS - https://galaxy.ansible.com/amazon/aws
- Docker - https://galaxy.ansible.com/community/docker
- MySQL - https://galaxy.ansible.com/community/mysql

Galaxy? Instalacja modułu

- Galaxy jest wbudowane w ansible
- Wywołanie za pomocą komendy ansible-galaxy

```
$ ansible-galaxy collection install community.docker
Starting galaxy collection install process
Process install dependency map
Starting collection install process
Downloading https://galaxy.ansible.com/download/community-docker-2.0.2.tar.gz to
/home/dkossako/.ansible/tmp/ansible-local-7255_hx2kvtu/tmprl8jf2w8/community-docker-2.0.2-trlklalk
Installing 'community.docker:2.0.2' to
'/home/dkossako/.ansible/collections/ansible_collections/community/docker'
community.docker:2.0.2 was installed successfully
```

Galaxy? requirements.txt

```
$ ansible-galaxy list
 - ansible-network.network-engine, v2.7.2
 - ansible-network.config_manager, v2.6.2
 - ansible-network.cisco_nxos, v2.7.1
 - ansible-network.vyos, v2.7.3
 - ansible-network.cisco_ios, v2.7.0


$ ansible-galaxy install -r requirements.yaml
Starting galaxy collection install process
Process install dependency map
Starting collection install process
community.docker:2.0.2 was installed successfully
```

Galaxy? requirements.txt

```
---
collections:
 - community.docker
 - community.mysql

roles:
 - idealista.mysql_role
```

Galaxy? Mysql

- Parametry mysql_db:
  - login_host
  - login_user
  - login_password
  - state (absent, dump, import, present)
- Credentiale (a przynajmniej hasło) jako zmienne z linii komend
- Dokumentacja: https://docs.ansible.com/ansible/latest/collections/community/mysql/mysql_db_module.html

Galaxy? MySQL

```
---

 - name: "Drop db-name"
   mysql_db:
     state: "absent"
     name: "db-name"
     login_host: "host"
     login_user: "root"
     login_password: "pass"
```

Koniec!

Dziękuję za uwagę

Pytania?