ANSIBLE

Wprowadzenie

PANDA ACADEMY

ZAGADNIENIA - ROADMAPA

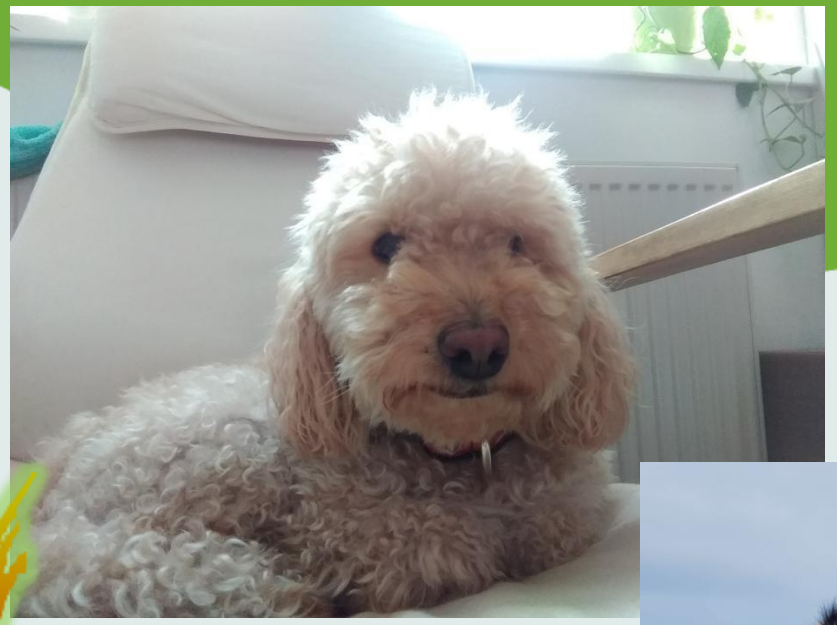
- WPROWADZENIE DO CI/CD/CD
- GIT 1
- ANSIBLE 1
- ANSIBLE 2
- DOCKER 1

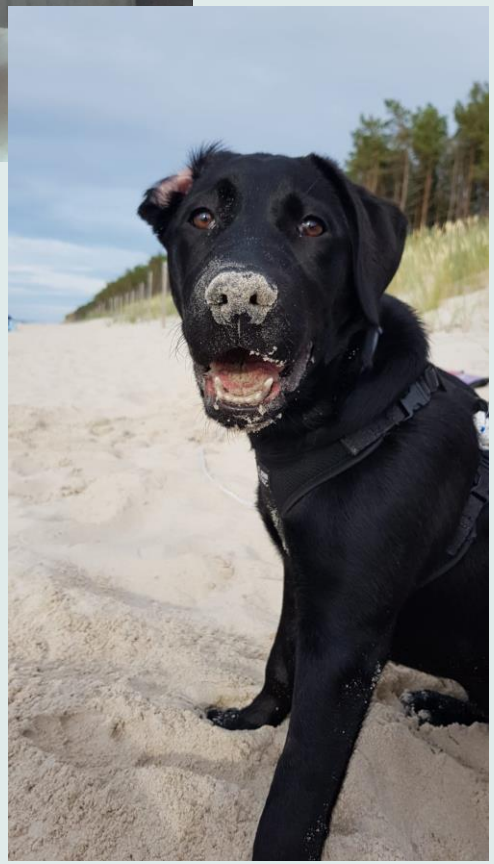
Część 1:

WPROWADZENIE

PANDA ACADEMY

BYDŁO VS PUPILEK





INFRASTRUKTURA JAKO KOD (IAC)

Infrastruktura jako kod (ang. Infrastructure as Code) to proces zarządzania serwerami oraz dostarczania na nie oprogramowania, poprzez użycie kodu w postaci plików (skryptów lub dedykowanych narzędzi), zamiast manualnego działania administratora.

Najczęściej termin IaC oznacza również traktowanie plików definicji równorzędnie z kodem aplikacji.

PANDA ACADEMY

IAC JAKO KOD APLIKACJI

Trzymanie definicji w repozytorium umożliwia:

- Wersjonowanie
- Wielokrotne używanie kodu
- Automatyzacje
- Posiadanie kopii zapasowej
- Wdrażanie w cyklu życia (jak w przypadku aplikacji)

Definicja IaC często pisana jest deklaratywnie (np. Terraform, jenkinsfile, docker-compose).

IaC jest bardzo przydatne dla Continuous Delivery, niezbędne dla Continuous Deployment.

PANDA ACADEMY

IAC - PRZYKŁADY

- Ansible
- Terraform
- AWS Cloud Formation
- Openshift
- Vagrant
- Docker-compose
- Kubernetes
- Jenkins - Jenkinsfile, Job DSL, CasC*

** Nie jest to infrastruktura tylko konfiguracja, ale wynika z podobnych założeń.*

PANDA ACADEMY

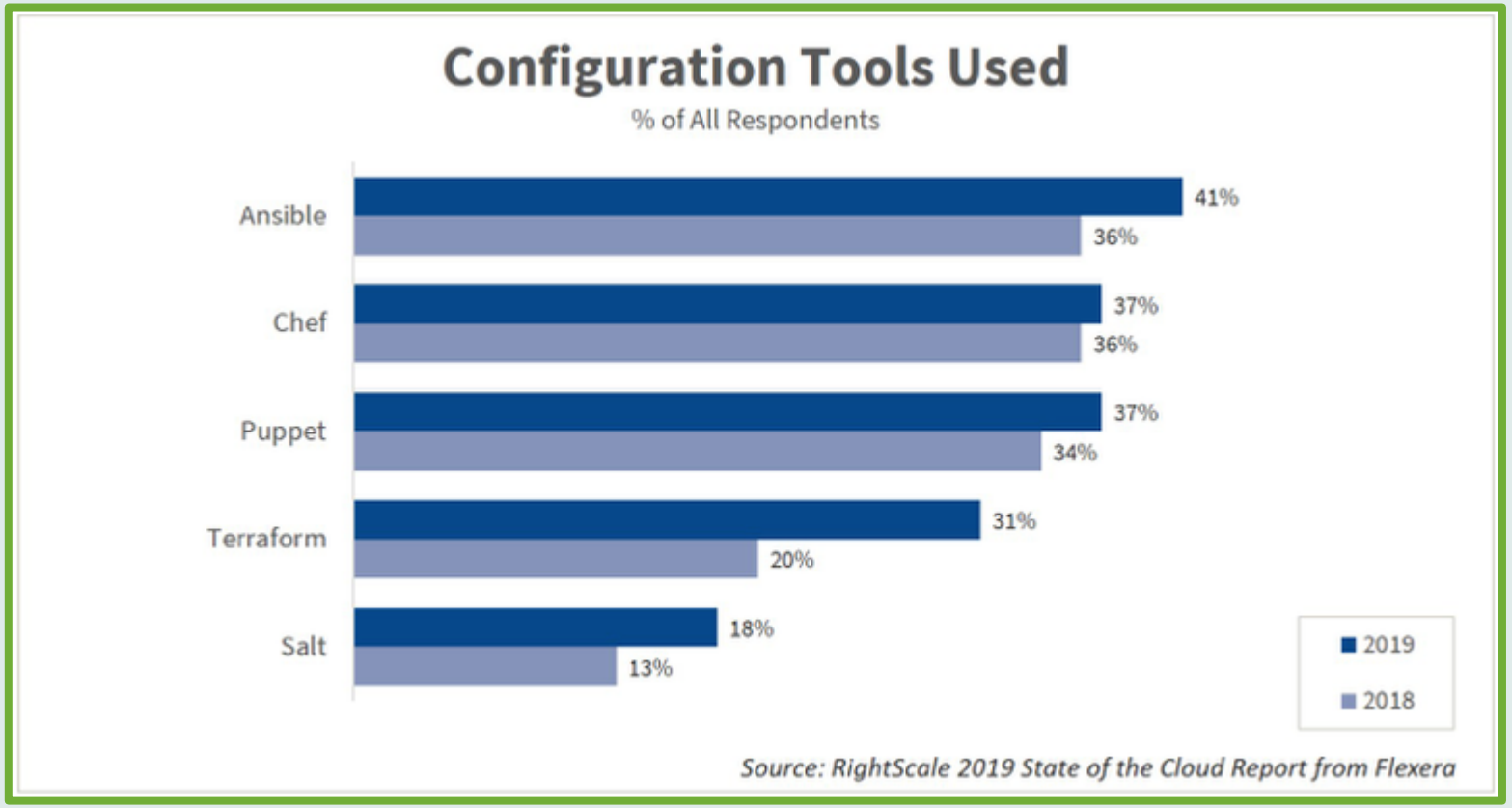
CZYM JEST ANSIBLE?

Ansible jest narzędziem służącym do automatyzacji zarządzania konfiguracją systemów (ang. configuration management), tworzenia nowych maszyn (ang. provisioning) oraz instalacji aplikacji.

Ansible, łącząc się ze zdalnymi maszynami używa protokołu SSH. Jest to znacząca różnica względem innych narzędzi CM, które, w trybie domyślnym, działają poprzez instalacje własnego klienta.

PANDA ACADEMY

CONFIGURATION MANAGEMENT NAJWIĘKSI GRACZE



PULL VS PUSH

Pull

- Klient odpytuje serwer o zmiany. Jeżeli na serwerze pojawiają się zmiany, zostają one wysłane.
- Większe obciążenie klienta, mniejsze serwera.
- Dłuższe czasy oczekiwania po wprowadzeniu zmian.
- Łatwiejsze zarządzanie w przypadku dużej ilości klientów, lepsze podejście, gdy serwery żyją „krótko”.
- Przykłady:
  - Jenkins git polling
  - Chef server

Push

- Serwer wdraża zmiany, gdy zostaną dodane, do wszystkich skonfigurowanych klientów.
- Większe obciążenie serwera, mniejsze (lub żadne, poza tym spowodowanym wprowadzeniem zmiany) klienta.
- Łatwiejsze zarządzanie w przypadku stosunkowo prostej architektury oraz małej ilości klientów.
- Przykłady:
  - Jenkins git hook
  - Ansible

PANDA ACADEMY

DLACZEGO NIE SKRYPTY?

Czy Configuration Management jest bardziej użyteczny od skryptów?

PANDA ACADEMY

DLACZEGO NIE SKRYPTY?

Konfigurowanie środowiska może być dokonane za pomocą skryptów. Takie rozwiązanie wymaga jednak rozległej wiedzy o wykorzystywanym języku i wstępnej konfiguracji każdego serwera (dotyczy Groovy, Pythona, czy Perla). Dodatkowo poziom skomplikowania skryptu rośnie wykładniczo w stosunku do ilości serwerów, platform, konfiguracji i ilości narzędzi. Jest trudniejszy w czytaniu przez osoby trzecie oraz w poprawianiu pojedynczych elementów, zwłaszcza w obliczu setek linii kodu.

PANDA ACADEMY

DLACZEGO NIE SKRYPTY?

PANDA ACADEMY

SKRYPTY VS ANSIBLE

Skrypty

- Wymagają istotnej znajomości określonego języka (np. Bash, Python, Groovy, Perl)
- Dla skomplikowanego zestawu operacji długi i nieczytelny kod
- Wymagają wstępnej konfiguracji
- Poziom skomplikowania skryptów rośnie wykładniczo dla większych środowisk
- Nie rozumieją kontekstu wywołania

Ansible

- Jest pisany deklaratywnie, przez co ma niższy próg wejścia
- Kod w formie deklaratywnej – łatwy do czytania przez osoby trzecie
- Nie wymaga wstępnej konfiguracji
- Poziom skomplikowania jest zazwyczaj taki sam, niezależnie od ilości maszyn czy konfiguracji
- Rozumie kontekst wywołania (fakty)
- Posiada gotowe moduły i role w ramach Ansible Galaxy

PANDA ACADEMY

PLIK INVENTORY

```
[local]
localhost

[aws]
[azure]
[serwery-pocztowe]
[linux]
```

Definicja serwerów do obsłużenia przez Ansible nazywa się plikiem inventory.

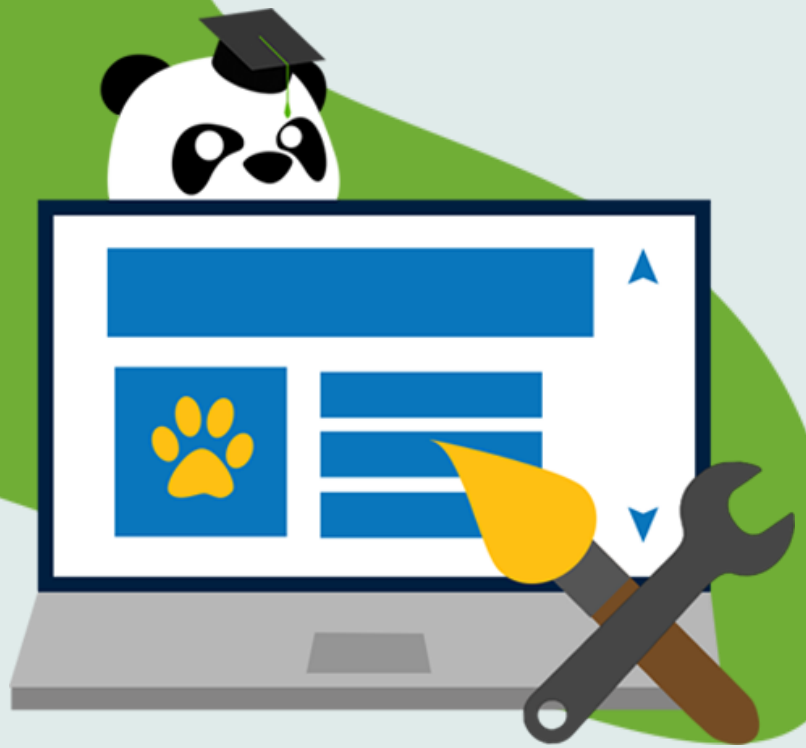
Domyślnie ten plik znajduje się w ścieżce:

`/etc/ansible/hosts`

ale może być utworzony ręcznie w dowolnej innej ścieżce (np. w katalogu domowym).

W ramach pliku inventory możemy zdefiniować dowolną liczbę sekcji zawierających adresy ip, nazwy lub grupy nazw serwerów, podzielonych według pasujących nam reguł.

https://docs.ansible.com/ansible/2.3/intro_inventory.html



ZADANIA SAMODZIELNE

Wykonaj poniższe ćwiczenia w konsoli Bash

- Wejdź do folderu `/etc/ansible/` i edytuj plik `hosts`
- Dodaj sekcję `[local]` (koniecznie pod tą nazwą) i dodaj wpis `localhost`

PANDA ACADEMY

URUCHOMIENIE ANSIBLE: PLAYBOOKI

Playbook to plik w formacie yaml, który przechowuje definicję zadań do wykonania na wybranej grupie hostów zdefiniowanych uprzednio w pliku inventory.

Playbooki mogą interpolować zmienne.

PANDA ACADEMY

KOMPONENTY ANSIBLE

Playbooki zawierają **playlisty**.

Playlisty zawierają zadania **(taski)**.

Taski uruchamiają **moduły**.

Taski uruchamiane są sekwencyjnie.

PANDA ACADEMY

PRZYKŁADOWY PLAYBOOK

Każdy playbook musi posiadać odwołanie do zdefiniowanego zestawu hostów z pliku inventory (sekcja hosts) oraz definicję zadania (task) lub roli (role) do wykonania.

Pojedyncze zadanie (task) musi zawierać nazwę oraz odwołanie do modułu, wraz z jego argumentami.

```
- name: Introduction to Playbooks
  hosts: local

  tasks:
  - name: create testansible.txt
    file:
      path: testansible.txt
      state: touch

  - name: Copy file with proper rights
    copy:
      src: ./testansible.txt
      dest: /home/testansible.txt
      mode: '0644’
```

PLAYBOOK: URUCHOMIENIE

```
ansible-playbook playbook.yml -i inventory
```

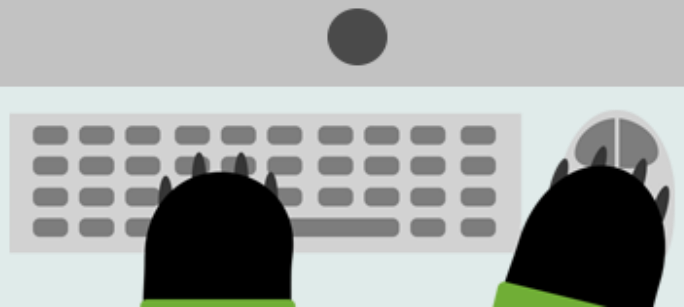
Wywołując komendę ansible-playbook należy podać ścieżkę do pliku playbook.yml (może mieć inną nazwę) oraz opcjonalnie plik inventory, jeżeli jest inny niż /etc/ansible/hosts

PANDA ACADEMY

PLAYBOOK: URUCHOMIENIE

```
ansible-playbook play.yml --syntax-check
```

Przełącznik --syntax-check pozwala
na zweryfikowanie poprawności pliku yaml
przed jego faktycznym uruchomieniem



PANDA ACADEMY

PLAYBOOK: URUCHOMIENIE

```
ansible-playbook play.yml --check
```

Przełącznik --check pozwala uruchomić Ansible
“na sucho” (ang. dry run), czyli zasymulować działanie
bez faktycznego nanoszenia zmian.
Nie myl z --syntax-check!

PANDA ACADEMY

CO, JAK I SKĄD?

- Skąd mam wiedzieć jakie narzędzia w moim systemie obsługuje Ansible?
- Skąd mam wiedzieć czy Ansible obsłuży dany program?
- Skąd mam wiedzieć jakie parametry i argumenty przyjmuje dany task?

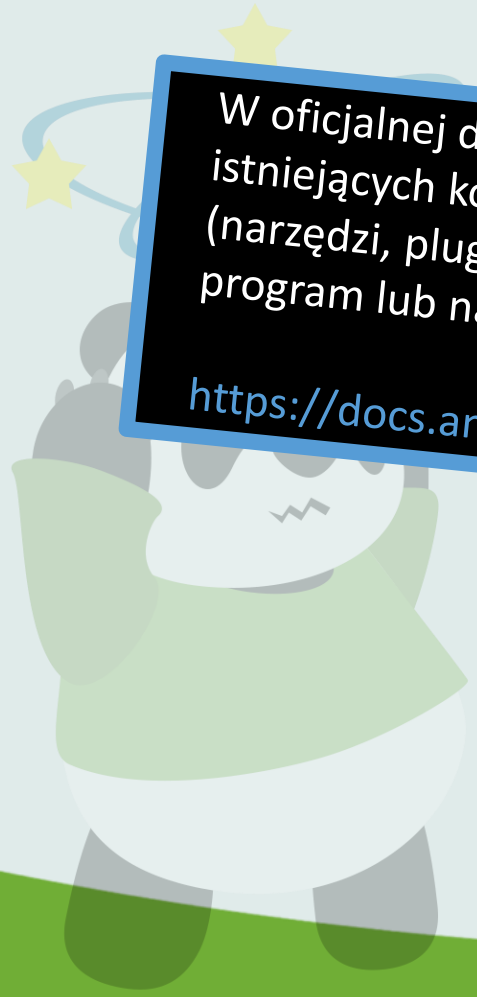
PANDA ACADEMY

MODUŁY I KOLEKCJE

- Skąd mam wiedzieć jakie narzędzia w moim systemie obsługuje
- Skąd mam w... ...etry i argumenty przyjmuje d... ...k?

W oficjalnej dokumentacji Ansible, znajdziemy wykaz wszystkich obecnie istniejących kolekcji, które zawierają w sobie zestaw gotowych implementacji (narzędzi, pluginów, modułów i ich argumentów) pozwalających obsłużyć dany program lub narzędzie.

https://docs.ansible.com/ansible/latest/collections/index.html#list-of-collections



PANDA ACADEMY

PIERWSZE URUCHOMIENIE

Utwórz plik playbook.yml zawierający kod znajdujący się obok (pamiętając o odpowiednich wcięciach).

Uruchom playbook na lokalnej maszynie korzystając z komendy

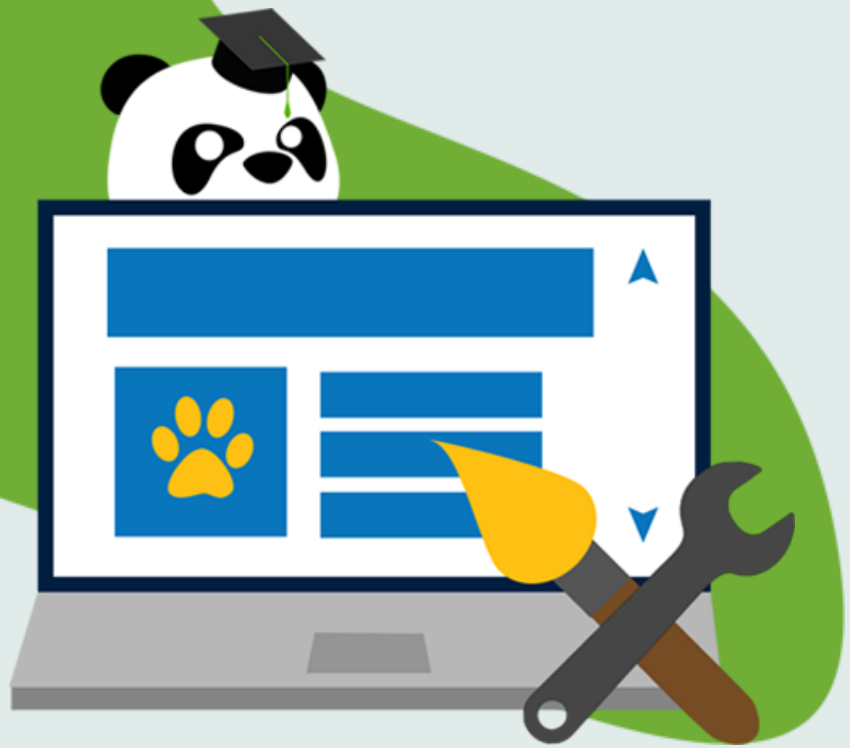
```
ansible-playbook playbook.yml
```

Zweryfikuj czy plik testansible.txt znajduje się w ścieżce /home/panda.

```
- name: Introduction to Playbooks
  hosts: local
  connection: local

  tasks:
  - name: create testansible.txt
    file:
      path: testansible.txt
      state: touch
```

Kod ćwiczenia znajdziesz tu:
https://pastebin.com/N0Es7eDc



ZADANIA SAMODZIELNE

Wykonaj poniższe ćwiczenia

- Utwórz plik packages.yml w folderze ansible w katalogu domowym, a następnie:
  - Dodaj nazwę dla playlisty
  - Dodaj definicję hosta: local
  - Dodaj rodzaj połączenia connection: local
  - Dodaj become: true aby podnieść uprawnienia
  - Dodaj definicję tasków dla unzip i net-tools dodając dwukrotnie
    - nazwę taska
    - rodzaj wykonywanego modułu (package)
    - nazwę instalowanego narzędzia (unzip / net-tools)
    - pożądany stan (present)
  - Zweryfikuj poprawność instalacji

PANDA ACADEMY

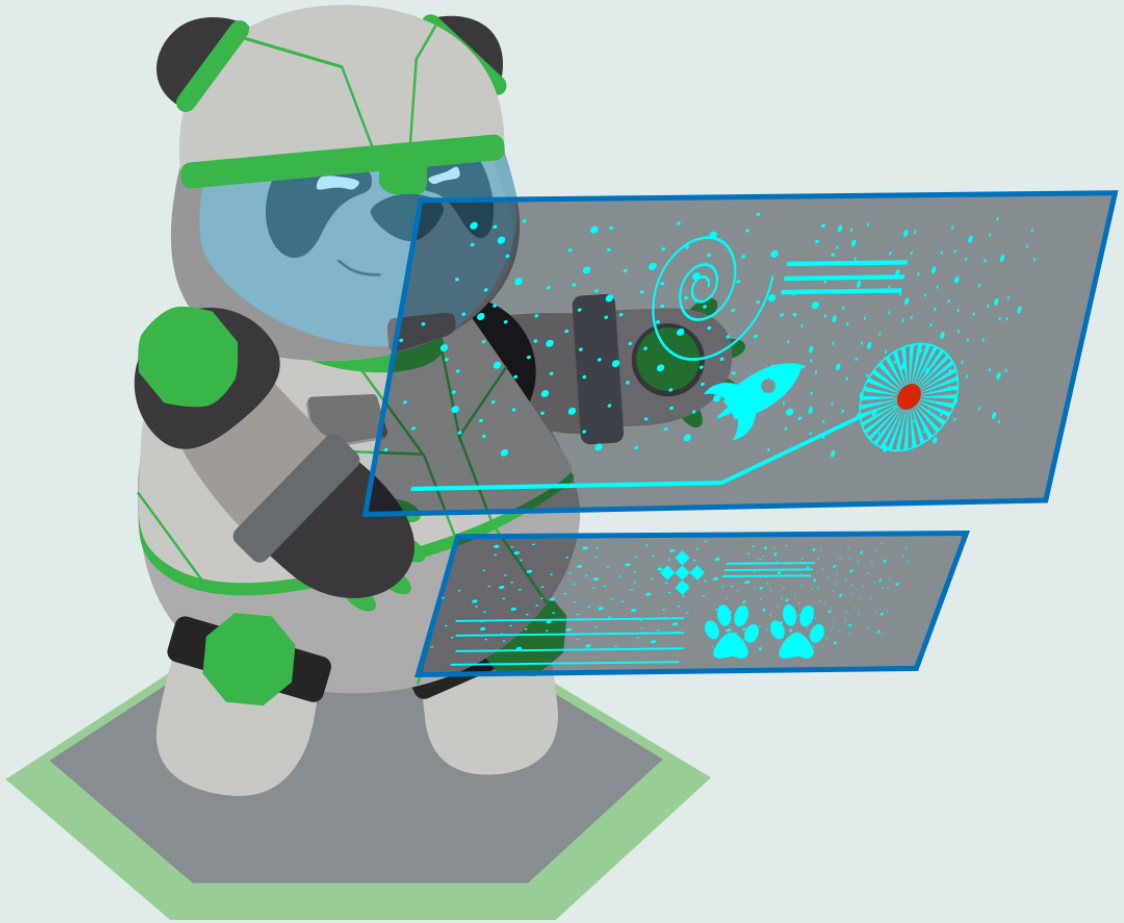
CZYM JEST ROLA?

Rola to komponent Ansible, który definiuje pojedynczy, logiczny zbiór zadań (tasków) do wykonania w ramach jednego wywołania.

W ramach roli możemy wykonywać wszystkie standardowe zadania a także dodawać elementy, takie jak np. pliki konfiguracyjne czy domyślne zmienne.

PANDA ACADEMY

ANSIBLE GALAXY



Ansible-galaxy pozwala na tworzenie i zarządzanie rolami oraz instalację kolekcji. Jest to jednak przede wszystkim miejsce, w którym możemy odnaleźć role stworzone przez innych użytkowników.

https://galaxy.ansible.com/

PANDA ACADEMY

INSTALACJA ROLI

```
ansible-galaxy install <rola>
```

Wywołenie komendy ansible-galaxy pobiera role z oficjalnego repozytorium!

PANDA ACADEMY

INSTALACJA ROLI

```
ansible-galaxy install -r requirements.yml
```

Pobierz wiele ról jednocześnie
zdefiniowanych w pliku requirements.yml

PANDA ACADEMY

REQUIREMENTS.YML

```
---
roles:
  -name: geerlingguy.git
   version: 3.0.0
  -name: inna rola
   version: inna wersja
```

Plik requirements.yml ma składnię podobną do playbooków.

Różni się miedzy innymi brakiem definicji nazwy oraz hostów.

Nie jest to typowa lista i wymaga wpisania sekcji roles.

PANDA ACADEMY

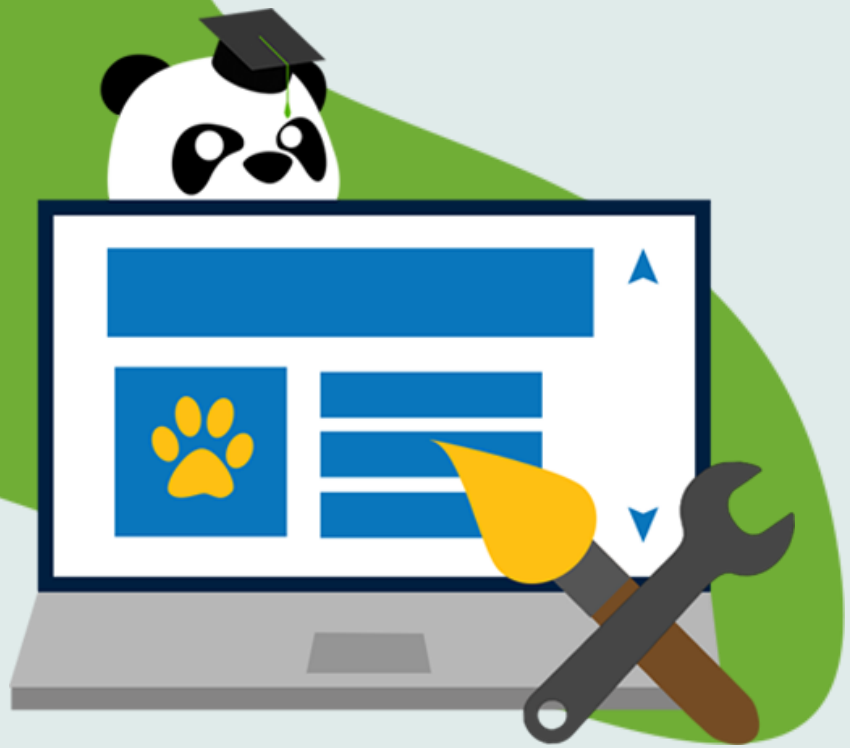
PLAYBOOK-ROLE.YML

```yaml
- name: How to use role
  roles:
    - NAZWA.ROLI1
    - NAZWA.ROLI2
```

Aby uruchomić rolę w playbooku musimy ją zdefiniować analogicznie do zadań (tasks:) w osobnej sekcji roles.

PANDA ACADEMY

Kod ćwiczenia znajdziesz tu:
https://pastebin.com/jaRa21dC



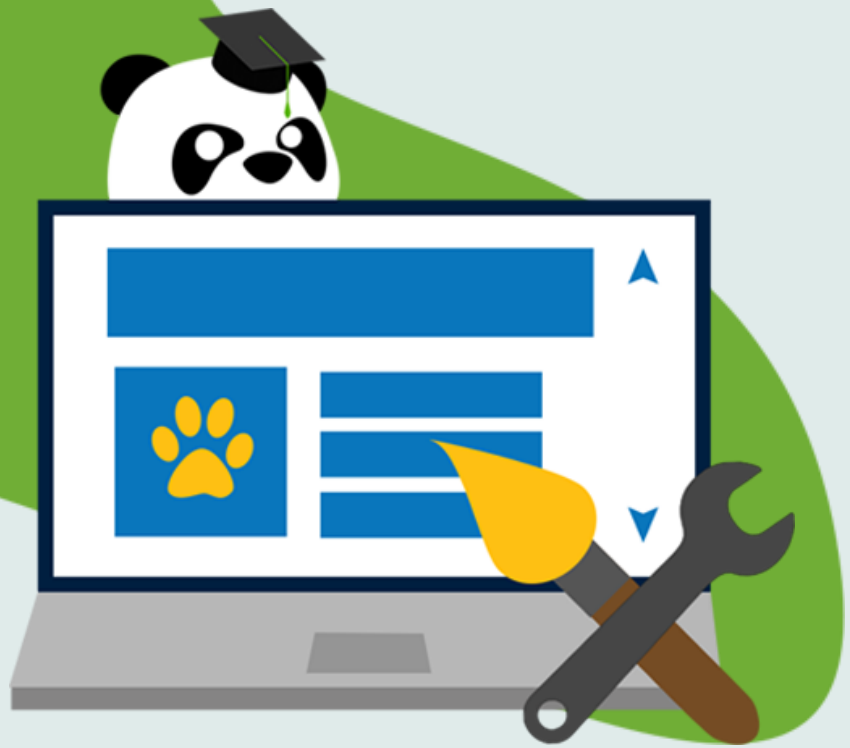
ZADANIA SAMODZIELNE

Wykonaj poniższe ćwiczenia

- Zainstaluj z ansible-galaxy rolę nginx
- Uruchom lokalnie instalację nginx za pomocą roli zdefiniowanej wewnątrz prostego playbooka, który:
  - uruchamia się wszędzie
  - używa połączenia lokalnego
  - uruchamia rolę nginx z ansible-galaxy

PANDA ACADEMY

Kod ćwiczenia znajdziesz tu:
https://pastebin.com/akLaGQij



ZADANIA SAMODZIELNE

Wykonaj poniższe ćwiczenia

- Utwórz plik requirements.yml w folderze ansible w katalogu domowym. Następnie znajdź poniższe role w ansible-galaxy i dodaj je do pliku:
  - Docker (w wersji 4.1.3)
  - Awscli (w wersji 1.0.2)
  - Awscli_configure (w wersji 0.0.5)
  - Terraform (w wersji diodonfrost 1.6.1)
- Utwórz nowe repozytorium github o nazwie Vagrant
- Wkopiuj do folderu z repozytorium (Vagrant) plik requirements.yml
- Wypchnij wszystkie zmiany za pomocą komend `git add`, `git commit` oraz `git push`.

PANDA ACADEMY