DEPLOYING PHP APPLICATIONS

WITH ANSIBLE, ANSIBLE VAULT AND ANSISTRANO

- Full Stack Web Developer & System Administrator
- Senior Developer at Microserve
- Part-time freelancer
- Acquia certified Drupal 8 Grand Master
- Drupal 7 & 8 core contributor
- Symfony, Laravel, Sculpin
- @opdavies
- www.oliverdavies.uk

- https://microserve.io
- https://www.drupal.org/microserve
- https://github.com/microserve-io
- https://twitter.com/microserveltd
- https://www.linkedin.com/company/microserve-ltd

THINGS WE'LL BE

LOOKING AT


ANSIBLE


VAGRANT


COMPOSER

1. Ansible crash course
2. Initial setup and provisioning
3. Basic deployment setup
4. Using Ansible Vault for variables
5. Adding and configuring Ansistrano

WHEN SHOULD I USE THIS?

- Dedicated hosting: probably has this already
- Shared hosting: probably not flexible enough
- VPS or dedicated server

WHAT IS

ANSIBLE?

Ansible is open source software
that automates software provisioning,
configuration management,
and application deployment.

https://en.wikipedia.org/wiki/Ansible_(software)

WHAT IS ANSIBLE?

- YAML
- Batteries included
- Executes remote commands
- Installs software packages
- Performs deployment steps

WHAT IS ANSIBLE?

- Hosts
- Commands
- Playbooks
- Tasks
- Roles

WHY ANSIBLE?

- Familiar syntax
- Easily readable
- No server dependencies
- Easy to add to an existing project
- Includes relevant modules (e.g. Composer)

ANSIBLE

HOSTS

```
# hosts.ini

[dransible]
192.168.33.10
```

ANSIBLE

COMMANDS

```
ansible all -m ping

ansible all -m command -a 'git pull --chdir=/var/www/app'
```

ANSIBLE

TASKS AND PLAYBOOKS

```yaml
# playbook.yml

---
- hosts: all

  tasks:
    - name: Update the code
      command: git pull
        args:
            chdir: /var/www/app
```

```
ansible-playbook playbook.yml -i hosts.ini
```

ANSIBLE

ROLES

```
# requirements.yml

---
- src: geerlingguy.apache
- src: geerlingguy.composer
- src: geerlingguy.mysql
- src: geerlingguy.php
- src: geerlingguy.php-mysql
```

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

```yaml
# playbook.yml

---
- hosts: all

  roles:
    - geerlingguy.apache
    - geerlingguy.mysql
    - geerlingguy.php
    - geerlingguy.php-mysql
    - geerlingguy.composer
```

LET'S TAKE A LOOK AT THE CODE

BASIC DEPLOYMENT

ANSIBLE

```yaml
# ansible/provision.yml

tasks:
  - name: Create a database
    mysql_db:
      name: mydatabase
      state: present

  - name: Add the database user
    mysql_user:
      name: drupal
      password: secret
      priv: '*.*:ALL'
      state: present
```

```yaml
# ansible/deploy.yml

tasks:
  - name: Creating project directory
    file:
      path: /var/www/app
      state: directory

  - name: Uploading application
    synchronize:
      src: "{{ playbook_dir }}/../"
      dest: /var/www/app

  - name: Installing Composer dependencies
    composer:
      command: install
      working_dir: /var/www/app
```

DISADVANTAGES

- Single point of failure
- No ability to roll back
- Sensitive data stored in plain text

KEEPING SECRETS

ANSIBLE VAULT

```
ansible-vault create ansible/vault.yml
```

```
$ANSIBLE_VAULT;1.1;AES256
3665623332353961633639383839613734393962323339333866653031373037323366326363306531336333353163643063663335396139363762393831333330a356365666232623537333730663638373932646161346131636637626664643737336637373830393161633362633235383935333232663432346662613438330a386435393432323761386137333736363436386466643031386662353933303936313864633132656538626338666635306264396230633939346532356665306564626431353936643135376238343466353666376132356561656433613166633965303832633330643332626431623539643166626234663736656337633036323837333137343533386165366531626462643662666266316639306262666539373236343662313265376261316636623963353933613366353737363435
```

```
# ansible/vars/vault.yml

---
vault_app_db_name: mydatabase
vault_app_db_user: drupal
vault_app_db_password: secret
```

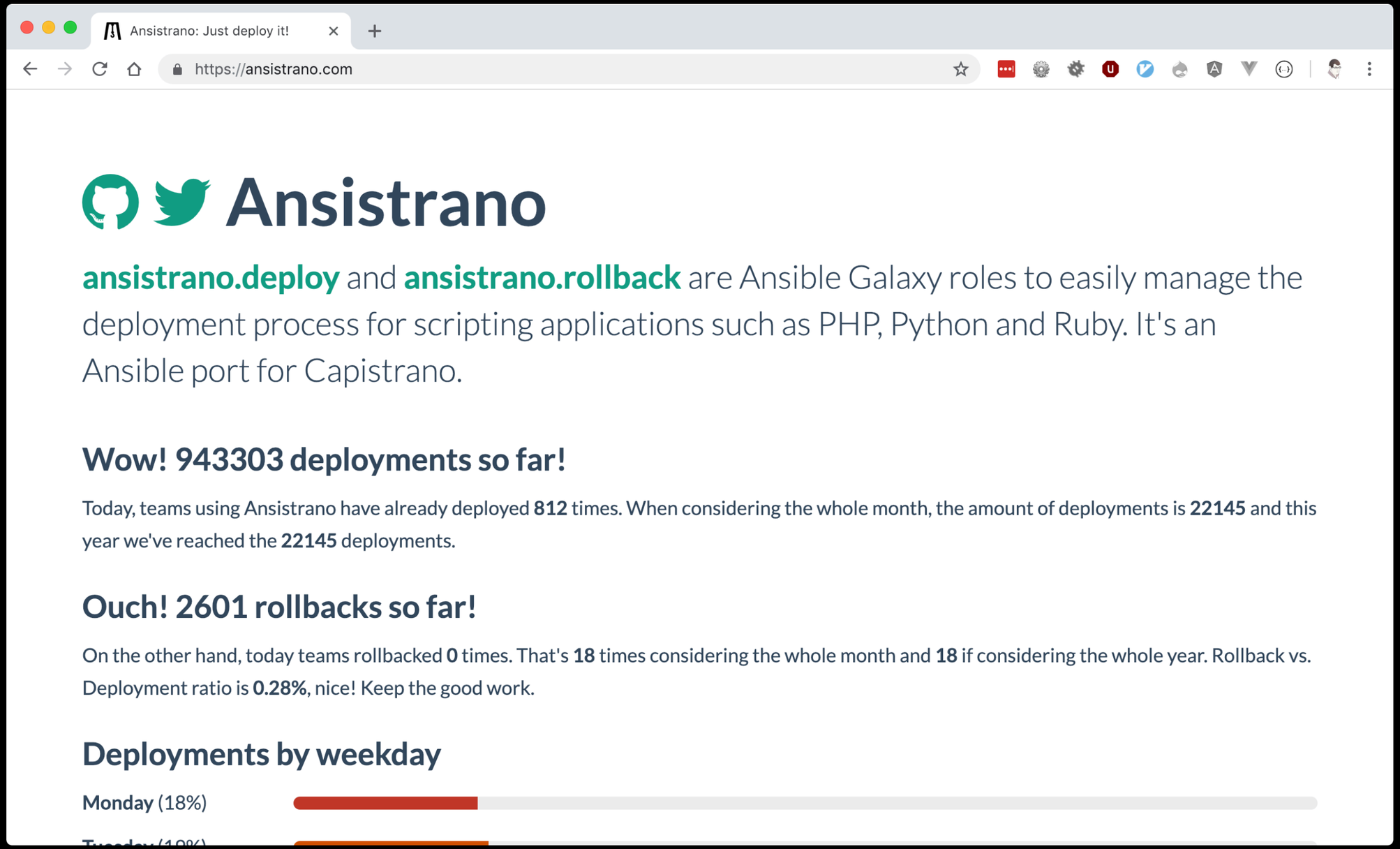
```yaml
# ansible/vars/vars.yml

---
app_db_name: "{{ vault_app_db_name }}"
app_db_user: "{{ vault_app_db_user }}"
app_db_password: "{{ vault_app_db_password }}"
```

```yaml
# ansible/provision.yml

tasks:
  - name: Create a database
    mysql_db:
      name: '{{ app_db_name }}'
      state: present
```

```
ansible-vault edit ansible/vault.yml
```

BETTER DEPLOYMENTS

ANSISTRANO


Ansistrano: Just deploy it!
https://ansistrano.com
Ansistrano
ansistrano.deploy and ansistrano.rollback are Ansible Galaxy roles to easily manage the deployment process for scripting applications such as PHP, Python and Ruby. It's an Ansible port for Capistrano.
Wow! 943303 deployments so far!
Today, teams using Ansistrano have already deployed 812 times. When considering the whole month, the amount of deployments is 22145 and this year we've reached the 22145 deployments.
Ouch! 2601 rollbacks so far!
On the other hand, today teams rollbacked 0 times. That's 18 times considering the whole month and 18 if considering the whole year. Rollback vs. Deployment ratio is 0.28%, nice! Keep the good work.
Deployments by weekday
Monday (18%)

Capistrano is an open-source tool for running scripts on multiple servers; its main use is deploying web applications.

It automates the process of making a new version of an application available on one or more web servers, including supporting tasks such as changing databases.

FEATURES

- Multiple release directories
- Shared paths and files
- Customisable
- Multiple deployment strategies
- Multi-stage environments
- Prune old releases
- Rollbacks

```
# ansible/requirements.yml

---
...
- ansistrano.deploy
- ansistrano.rollback
```

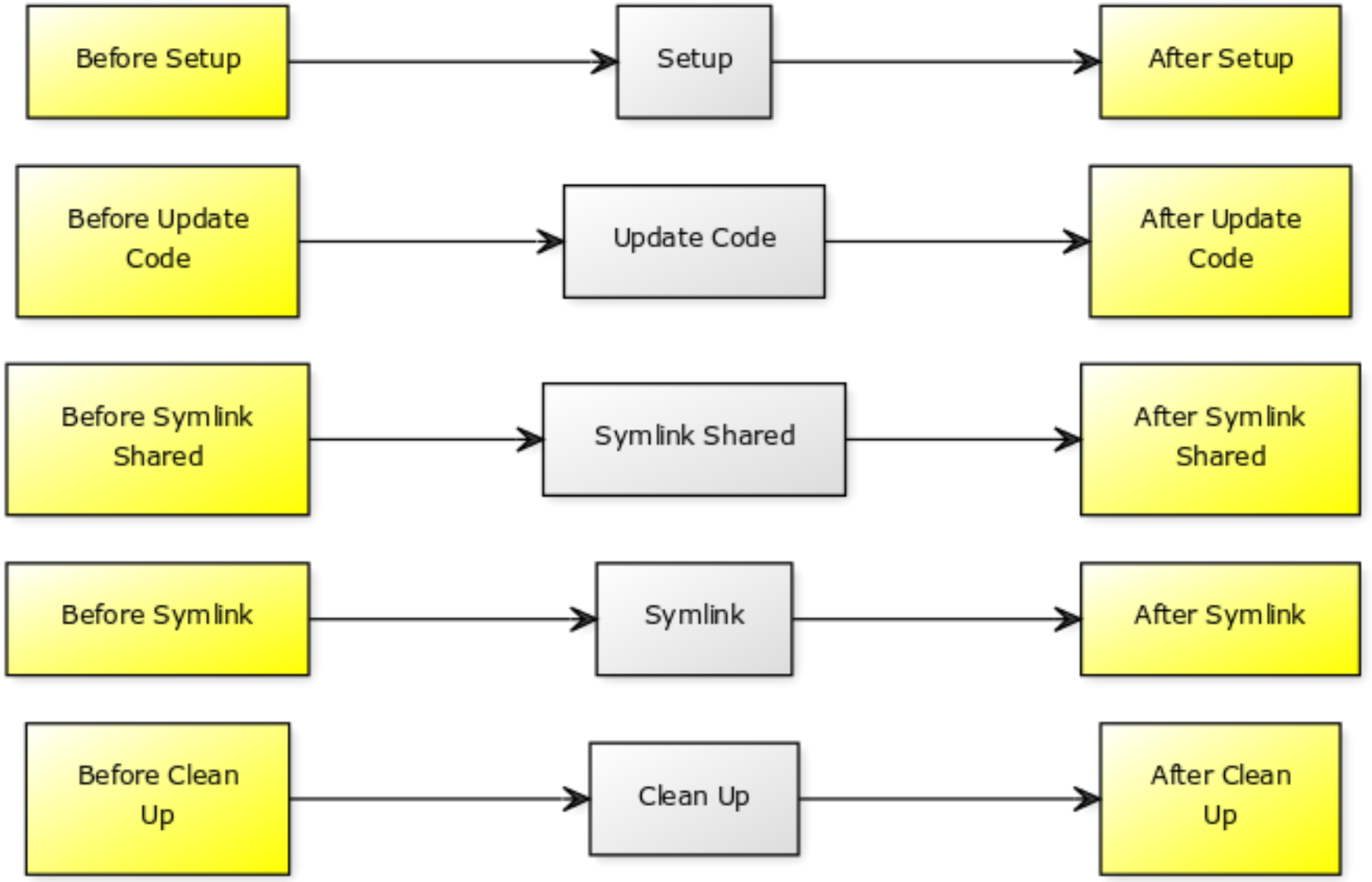
```yaml
# ansible/deploy.yml

---
- hosts: all

  roles:
    - ansistrano.deploy
```

```yaml
# ansible/deploy.yml

---
  ...
  vars:
    ansistrano_deploy_to: /var/www
    ansistrano_deploy_via: git
    ansistrano_git_branch: master
    ansistrano_git_repo: 'git@github.com:foo/bar.git'
```

```
# ansible/rollback.yml

---
- hosts: all

  roles:
    - ansistrano.rollback

  vars:
    ansistrano_deploy_to: /var/www
```

CUSTOMISING ANSISTRANO

BUILD HOOKS


Before Setup
Setup
After Setup
Before Update Code
Update Code
After Update Code
Before Symlink Shared
Symlink Shared
After Symlink Shared
Before Symlink
Symlink
After Symlink
Before Clean Up
Clean Up
After Clean Up

```
# ansible/deploy.yml

vars:
  ...
  ansistrano_after_symlink_shared_tasks_file: "{{ playbook_dir }}/deploy/after-symlink-shared.yml"
  ansistrano_after_symlink_tasks_file: "{{ playbook_dir }}/deploy/after-symlink.yml"
  ansistrano_after_update_code_tasks_file: "{{ playbook_dir }}/deploy/after-update-code.yml"

  release_web_path: "{{ ansistrano_release_path.stdout }}/web"
  release_drush_path: "{{ ansistrano_release_path.stdout }}/vendor/bin/drush"
```

```yaml
# ansible/deploy/after-update-code.yml

---
- name: Install Composer dependencies
  composer:
    command: install
    working_dir: '{{ ansistrano_release_path.stdout }}'
```

```
# ansible/deploy/after-symlink-shared.yml

---
- name: Run database updates
  command: '{{ release_drush_path }} --root {{ release_web_path }} updatedb'
```

```
# ansible/deploy/after-symlink.yml

---
- name: Clear Drupal cache
  command: '{{ release_drush_path }} --root {{ release_web_path }} cache-rebuild'
```

QUESTIONS?

THANKS

@OPDAVIES

OLIVERDAVIES.UK