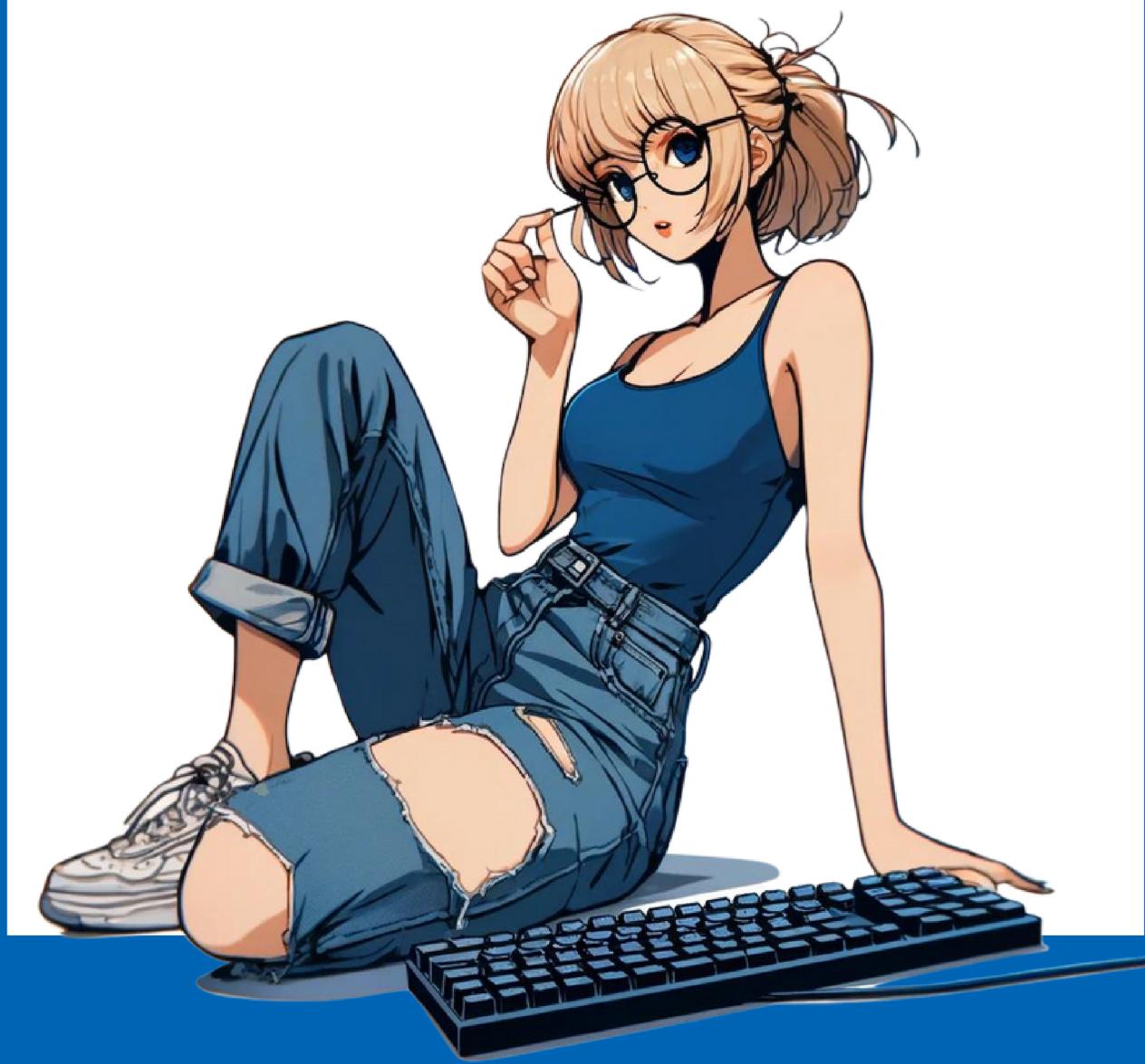


Antony Kervazo-Canut

Vagrant for Teenagers

**AUTOMATISATION DES
MACHINES VIRTUELLES**



SOMMAIRE

Introduction	3
Installation de Vagrant	4
Vagrantfile	5
Commandes Vagrant	6
Vagrant Cloud	7
Gestion des boxes	8
Réseau & Communication	9
Partage des dossiers	10
Provisionning Shell	11
Provisionning Ansible	12
Provisionning Puppet	13
Provisionning Chef	14
Vagrant Plugins	15
Plugin vagrant-triggers	16
Snapshots	17
Multi-Machine	18
Packages boxes	19

Introduction



Vagrant est un outil de gestion de machines virtuelles qui permet de configurer des environnements de développement reproductibles et portables. Il utilise des configurations simples codées en format de texte (principalement des fichiers Vagrantfile) pour automatiser la configuration des machines virtuelles, afin que vous puissiez facilement partager et collaborer sur des environnements de développement sans se soucier des différences spécifiques à chaque système d'exploitation ou infrastructure.

Vagrant fournit une interface simple pour gérer et provisionner des machines virtuelles. Voici quelques raisons d'utiliser Vagrant :

- Consistance : Les environnements Vagrant sont hautement reproductibles, ce qui réduit les "mais ça marche sur ma machine" en développement.
- Portabilité : Avec un simple `vagrant up`, n'importe qui peut démarrer un environnement de développement complet sans effort, sur n'importe quel système d'exploitation (Windows, MacOS, Linux).
- Flexibilité : Compatible avec de nombreux fournisseurs de machines virtuelles (VirtualBox, VMware, Hyper-V, Parallels, etc.) et outils de provisioning (Shell scripts, Chef, Puppet, Ansible, etc.).

Installation de Vagrant



Vagrant peut s'installer via la CLI. Pour windows cependant, vous devrez passer par le site :

<https://developer.hashicorp.com/vagrant/downloads>

```
● ● ●

# Installation de Vagrant sur MacOS
brew install vagrant

# Installation de Vagrant sur Linus
sudo apt install vagrant
```

Ensute il faudra installer un fournisseur. Sur Linux, vous pouvez installer VirtualBox ou KVM, sur Windows VirtualBox fonctionnera mais il faudra désactiver HyperV. Pour MacOS, sur les derniers modèle il faudra se tourner vers Parallels Desktop.

```
● ● ●

# MacOS - plugin pour parallels desktop
vagrant plugin install vagrant-parallel
```

Il sera bien sûr possible de mettre des fournisseurs directement dans le cloud.

Vagrantfile



Le Vagrantfile est le fichier de configuration -- écrit en Ruby -- principal de Vagrant. Il définit les paramètres et les ressources pour vos machines virtuelles. Lorsque vous exécutez vagrant init, un Vagrantfile par défaut est créé dans votre répertoire de projet.

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

# Définir la box Vagrant à utiliser
Vagrant.configure("2") do |config|
  config.vm.box = "hashicorp/bionic64"

  # Configurer les options de base de la machine virtuelle
  config.vm.provider "virtualbox" do |vb|
    vb.memory = "1024"
    vb.cpus = 2
  end

  # Configurer le réseau
  config.vm.network "private_network", ip: "192.168.50.4"

  # Configurer le partage de dossiers
  config.vm.synced_folder ".", "/vagrant", type: "virtualbox"

  # Configurer le provisionnement
  config.vm.provision "shell", inline: <--SHELL
    echo "Hello, World!"
  SHELL
end
```

Commandes Vagrant



Vagrant utilise une interface en ligne de commande (CLI) pour gérer les machines virtuelles.

```
● ● ●

# Initialise un nouveau répertoire Vagrant avec un fichier
# Vagrantfile par défaut.
vagrant init

# Démarré et configure une machine virtuelle.
vagrant up

# Se connecte à une machine virtuelle en utilisant SSH.
vagrant ssh

# Arrête une machine virtuelle en cours d'exécution.
vagrant halt

# Supprime une machine virtuelle et libère les ressources
# associées.
vagrant destroy
```

Vagrant Cloud



Vagrant Cloud est un service en ligne qui permet de stocker, partager et découvrir des boxes Vagrant. Vagrant Cloud facilite la collaboration entre les utilisateurs de Vagrant en leur offrant un espace centralisé pour gérer leurs boxes.

V HashiCorp **Vagrant Cloud** Search Pricing Vagrant Help Create an Account Sign In

Discover Vagrant Boxes

Search for boxes by operating system, included software, architecture and more

Provider any ▾ Architecture any ▾ Sort by Downloads ▾

 ubuntu/trusty64 20190514.0.0	virtualbox	Downloads 30,776,811	Released over 4 years ago
Official Ubuntu Server 14.04 LTS (Trusty Tahr) builds (End of standard support)			
 laravel/homestead 14.0.2	libvirt parallels virtualbox vmware_desktop	Downloads 14,578,770	Released 3 months ago
Official Laravel local development box.			
 hashicorp/precise64 1.1.0	hyperv virtualbox vmware_fusion	Downloads 6,813,293	Released about 10 years ago
A standard Ubuntu 12.04 LTS 64-bit box.			
 centos/7 2004.01	hyperv libvirt virtualbox vmware and 3 more providers	Downloads 6,008,701	Released almost 4 years ago
CentOS Linux 7 x86_64 Vagrant Box			
 ubuntu/xenial64 20211001.0.0	virtualbox	Downloads 3,618,923	Released over 2 years ago
Official Ubuntu 16.04 LTS (Xenial Xerus) builds (End of standard support)			

Gestion des boxes



Les boxes Vagrant sont des images de machines virtuelles préconfigurées qui peuvent être utilisées pour créer rapidement et facilement des environnements de développement cohérents. Les boxes contiennent généralement un système d'exploitation, des logiciels et des configurations spécifiques. Tout peut être changé au sein du Vagrantfile.

```
● ● ● Vagrantfile

config.vm.provider "virtualbox" do |vb|
  # Définit le nom de la machine virtuelle affiché dans
  l'interface du fournisseur.
  vb.name = "ma-machine-virtuelle"
  # Définit la quantité de mémoire allouée à la machine
  # virtuelle.
  vb.memory = "2048"
  # Définit le nombre de processeurs virtuels alloués à la
  machine virtuelle.
  vb.cpus = 2
  # Définit l'emplacement où le disque dur virtuel de la
  machine virtuelle sera stocké sur la machine hôte.
  vb.customize ["modifyvm", :id, "--hddfolder",
  "/chemin/vers/dossier"]
  # Démarre la machine virtuelle sans interface graphique.
  vb.gui = false
  # Active l'accélération 3D pour la machine virtuelle
  vb.gui = true
  vb.customize ["modifyvm", :id, "--accelerate3d", "on"]
end
```

Réseau & Communication



Vagrant offre plusieurs options de configuration du réseau pour les machines virtuelles.

```

● ● ● Vagrantfile

config.vm.provider "virtualbox" do |vb|
  # Créer un réseau privé isolé entre la machine hôte et la
  machine virtuelle
  config.vm.network "private_network", ip: "192.168.50.4"

  # Créer un réseau public accessible depuis d'autres
  machines sur le réseau local.
  config.vm.network "public_network"

  # Créer un réseau bridge permet à la machine virtuelle
  d'être connectée directement au réseau local, comme si elle
  était une machine physique.
  # Vous pouvez vérifier les interfaces réseau disponibles
  sur votre système en exécutant la commande ifconfig sur Linux
  ou macOS, ou ipconfig sur Windows.
  config.vm.network "public_network", bridge: "en1: Wi-Fi
  (AirPort)"

  # Créer un réseau NAT est le type de réseau par défaut
  utilisé par Vagrant. Il permet à la machine virtuelle
  d'accéder à Internet et aux réseaux locaux via la machine
  hôte, mais les autres appareils sur le réseau ne peuvent pas
  accéder directement à la machine virtuelle.
  # Le port 80 de la machine virtuelle est redirigé vers le
  port 8080 de la machine hôte.
  config.vm.network "forwarded_port", guest: 80, host: 8080
end

```

Partage des dossiers



Le partage de dossiers est une fonctionnalité importante de Vagrant qui permet de partager des répertoires entre la machine hôte et la machine virtuelle. Cela facilite le développement, car vous pouvez modifier des fichiers sur la machine hôte et les utiliser directement dans la machine virtuelle, sans avoir à les copier manuellement.

```
● ● ● Vagrantfile

config.vm.provider "virtualbox" do |vb|
  # Partager un dossier entre la machine hôte et la machine
  # virtuelle en utilisant VirtualBox
  config.vm.synced_folder "/chemin/vers/dossier/hote",
  "/chemin/vers/dossier/invite", type: "virtualbox"

  # Sur un système Windows avec Hyper-V comme fournisseur,
  # vous pouvez utiliser le partage de dossiers SMB.
  config.vm.synced_folder "/chemin/vers/dossier/hote",
  "/chemin/vers/dossier/invite", type: "smb"

  # Partage dont la synchronisation est unidirectionnelle,
  # de la machine hôte vers la machine virtuelle.
  config.vm.synced_folder "/chemin/vers/dossier/hote",
  "/chemin/vers/dossier/invite", type: "rsync"

  # Partage de dossiers SSHFS utilise le protocole SSH pour
  # monter un répertoire distant sur un répertoire local.
  # Cette méthode peut être utile lorsque vous souhaitez
  # partager des dossiers entre des machines virtuelles et des
  # hôtes distants.
  config.vm.synced_folder "/chemin/vers/dossier/hote",
  "/chemin/vers/dossier/invite", type: "sshfs"
end
```

Provisioning Shell



Le provisionnement est le processus d'installation et de configuration automatisées du logiciel sur une machine virtuelle. Vagrant prend en charge plusieurs méthodes de provisionnement, ce qui vous permet de personnaliser et de configurer vos machines virtuelles en fonction de vos besoins.

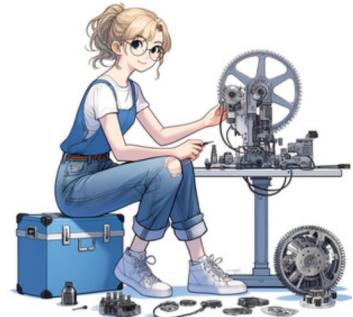
Le provisionnement avec des scripts shell est une méthode simple et courante pour configurer une machine virtuelle. Vous pouvez écrire un script shell qui exécute les commandes nécessaires pour installer et configurer le logiciel sur la machine virtuelle.

```
● ● ● Vagrantfile

config.vm.provider "virtualbox" do |vb|
  # Les commandes shell entre les délimiteurs <--SHELL et
  # SHELL seront exécutées sur la machine virtuelle lors du
  # provisionnement.
  config.vm.provision "shell", inline: <--SHELL
    # Commandes shell à exécuter
  SHELL

  # Vous pouvez spécifier un fichier de script shell
  # externe en utilisant l'option path
  config.vm.provision "shell", path: "script.sh"
end
```

Provisioning Ansible



Ansible est un outil de configuration et de déploiement populaire qui permet de gérer des machines à distance en utilisant des playbooks. Vagrant prend en charge le provisionnement avec Ansible, ce qui vous permet d'utiliser des playbooks Ansible pour configurer vos machines virtuelles.

```
● ● ● Vagrantfile

config.vm.provider "virtualbox" do |vb|
  config.vm.provision "ansible" do |ansible|
    ansible.playbook = "playbook.yml"
  end
end
```

Provisioning Puppet



Puppet est un autre outil de gestion de configuration populaire qui permet de gérer l'état des machines à distance en utilisant des manifestes. Vagrant prend en charge le provisionnement avec Puppet, ce qui vous permet d'utiliser des manifestes Puppet pour configurer vos machines virtuelles.

```
● ● ● Vagrantfile

config.vm.provider "virtualbox" do |vb|
  config.vm.provision "puppet" do |puppet|
    puppet.manifests_path = "manifests"
    puppet.manifest_file  = "site.pp"
  end
end
```

Provisioning Chef



Chef est un outil de gestion de configuration qui permet de gérer l'état des machines à distance en utilisant des cookbooks. Vagrant prend en charge le provisionnement avec Chef, ce qui vous permet d'utiliser des cookbooks Chef pour configurer vos machines virtuelles.

```
● ● ● Vagrantfile

config.vm.provider "virtualbox" do |vb|
  config.vm.provision "chef_solo" do |chef|
    chef.cookbooks_path = "cookbooks"
    chef.roles_path = "roles"
    chef.data_bags_path = "data_bags"
    chef.json = {
      # Données JSON pour votre configuration Chef
    }
  end
end
```

Vagrant Plugins



Vagrant prend en charge les plugins, qui sont des extensions développées par la communauté pour ajouter des fonctionnalités supplémentaires à Vagrant. Une liste est tenue par la communauté et disponible ici :

<https://github.com/hashicorp/vagrant/wiki/Available-Vagrant-Plugins>

```
● ● ● Vagrantfile

# Installer un plugin, par exemple vagrant-triggers
vagrant plugin install vagrant-triggers

# Afficher la liste des plugins installés
vagrant plugin list

# Mettre à jour un plugin Vagrant
vagrant plugin update vagrant-triggers

# Désinstaller un plugin Vagrant
vagrant plugin uninstall vagrant-triggers
```

Les plugins Vagrant peuvent étendre les fonctionnalités de base de Vagrant et vous aider à personnaliser votre environnement de développement en fonction de vos besoins.

Plugin vagrant-triggers



vagrant-triggers est un plugin Vagrant qui permet d'exécuter des commandes ou des scripts personnalisés en réponse à des événements spécifiques dans le cycle de vie d'une machine virtuelle Vagrant. Ces événements, appelés déclencheurs (triggers), peuvent être utilisés pour automatiser davantage le processus de provisionnement et de configuration de vos machines virtuelles.

```
● ● ● Vagrantfile

config.vm.provider "virtualbox" do |vb|
  config.trigger.before :up do
    # Commandes ou scripts à exécuter avant le démarrage de la
    # machine virtuelle
    end

    config.trigger.after :halt do
      # Commandes ou scripts à exécuter après l'arrêt de la
      # machine virtuelle
      end
end
```

vagrant-triggers prend en charge de nombreux autres événements, tels que :provision, :reload, :suspend, et :resume.

Snapshots



Les snapshots sont une fonctionnalité utile de Vagrant qui vous permet de sauvegarder l'état d'une machine virtuelle à un moment donné et de la restaurer facilement si nécessaire. Les snapshots peuvent être utilisés pour tester différentes configurations, expérimenter des modifications ou simplement sauvegarder l'état actuel de votre machine virtuelle avant d'apporter des changements importants.

```
# Création d'un snapshot
vagrant snapshot save mon_snapshot

# Afficher la liste des snapshots disponibles pour une
# machine virtuelle
vagrant snapshot list

# Restauration d'un snapshot
vagrant snapshot restore mon_snapshot

# Suppression d'un snapshot
vagrant snapshot delete mon_snapshot
```

Multi-Machine



Vagrant propose un moyen de gérer plusieurs machines virtuelles à la fois en utilisant la fonctionnalité "Multi-Machine". Avec cette fonctionnalité, vous pouvez définir et gérer plusieurs machines virtuelles dans un seul Vagrantfile, et les démarrer, les arrêter et les provisionner en parallèle.

```
Vagrantfile

Vagrant.configure("2") do |config|


  config.vm.define "db" do |db|
    db.vm.box = "ubuntu/xenial64"
    db.vm.network "private_network", ip: "192.168.50.10"
    # Autres configurations spécifiques à la machine
    # virtuelle "db"
    end

  config.vm.define "web" do |web|
    web.vm.box = "ubuntu/xenial64"
    web.vm.network "private_network", ip: "192.168.50.11"
    # Autres configurations spécifiques à la machine
    # virtuelle "web"
    end

end
```

Si vous souhaitez gérer une machine virtuelle spécifique, vous pouvez ajouter le nom de la machine virtuelle à la commande Vagrant. Par exemple, pour démarrer uniquement la machine virtuelle "db", utilisez la commande `vagrant up db`.

Packages boxes



Vous pouvez utiliser l'outil vagrant package pour créer un fichier de box à partir d'une machine virtuelle existante pour pouvoir la partager sur un repository tel que Vagrant Cloud.



```
# Créer un fichier nommé "my-box.box" contenant votre machine  
virtuelle  
vagrant package --output my-box.box
```

Vagrant Cloud permet aux utilisateurs de partager et de distribuer leurs propres boxes personnalisées, facilitant ainsi la création d'environnements de développement cohérents et reproductibles. En publiant une box sur Vagrant Cloud, les utilisateurs peuvent rendre leurs configurations de machines virtuelles accessibles à d'autres personnes, favorisant la collaboration et l'efficacité dans les projets de développement. Les boxes publiées sur Vagrant Cloud peuvent être mises à jour et gérées avec des versions, offrant ainsi un contrôle total sur leur distribution et leur utilisation.

Dans la même collection

Orchestration et Gestion de Conteneurs



Infrastructure as Code



Sécurité & Gestion des secrets



Développement & CI/CD



↓ FOLLOW ME ↓



[ANTONYCANUT](#)



[ANTONY KERVAZO-CANUT](#)