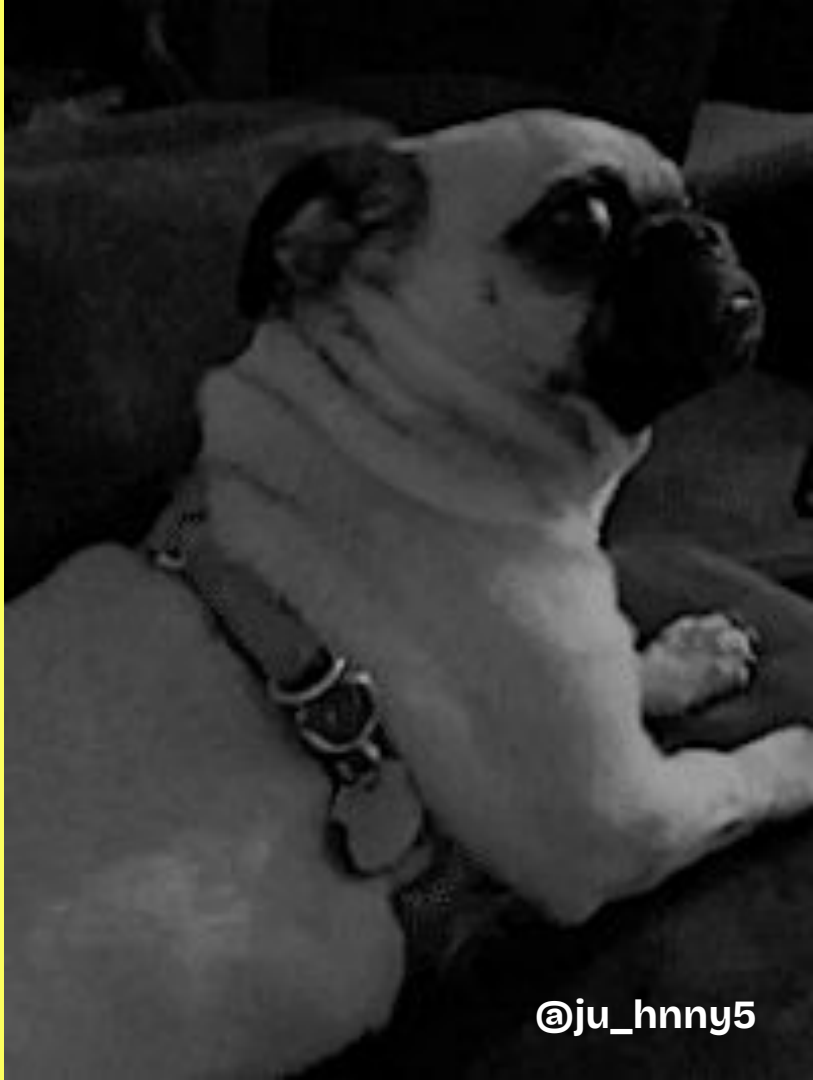
Pulumi

Gérer son infra avec son langage de programmation préféré*

Julien Briault - Deezer



~#whoami

Julien Briault
(ex) SecOps consultant chez **Rudder**

IT/Infrastructure Manager (bénévole) aux Restaurants du Cœur
Network Engineer / SRE chez **DEEZER**

Auteur principal sur blog.jbriault.fr

#Networking #FOSS #Dev #Music

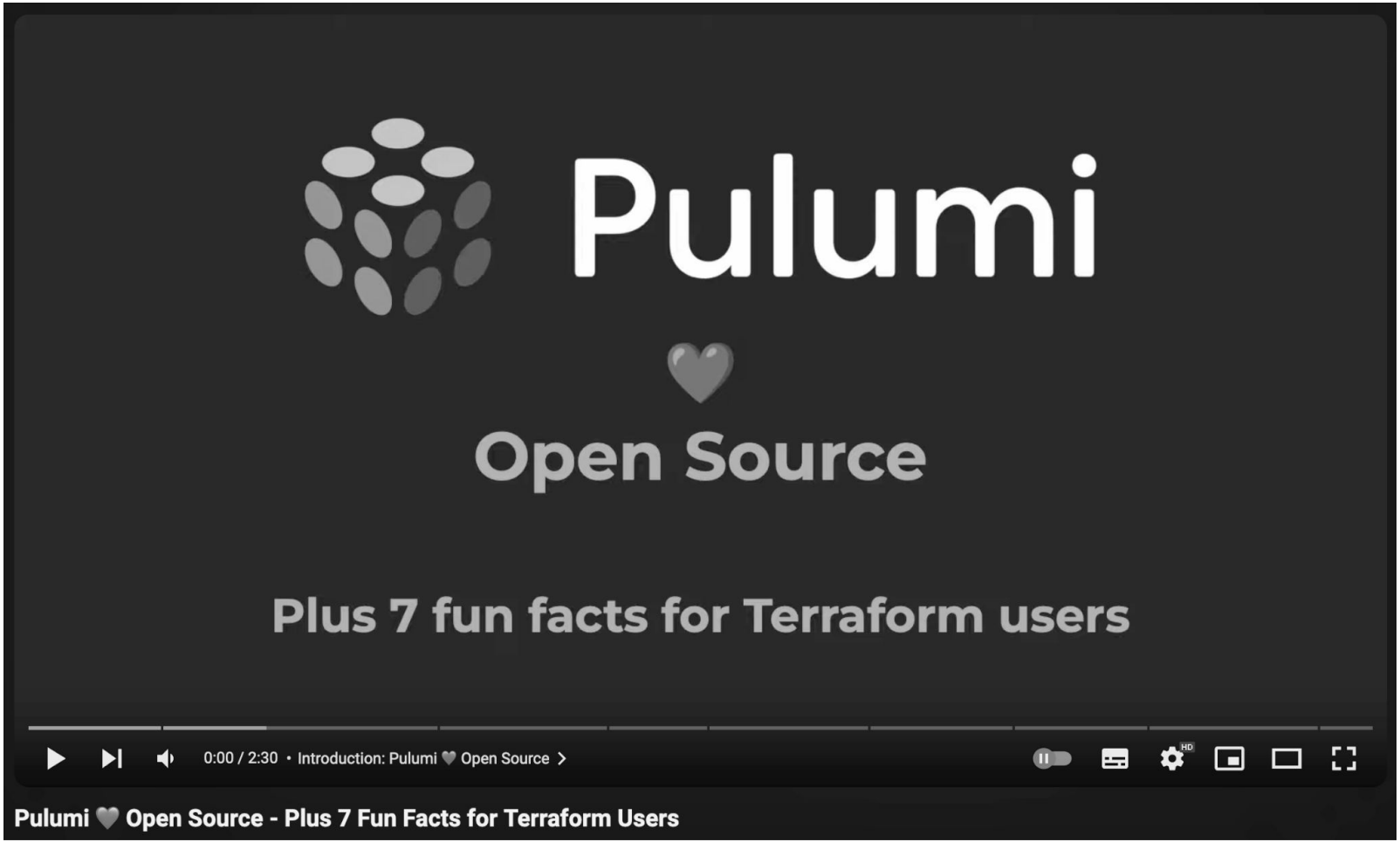
@ju_hnny5


Et oui, je ne dors pas beaucoup...
@ju_hnny5


#TALLBOYZ
Julien Briault
@ju_hnny5
Actuellement aux @restosducoeur nous recherchons des dons de pc portables, d'écran (avec HDMI), raspberry, etc.
Si jamais vous connaissez des entreprises ouvertes aux dons, n'hésitez pas à me contacter. 🙏
WE NEED YOUR HELP!
@ju_hnny5

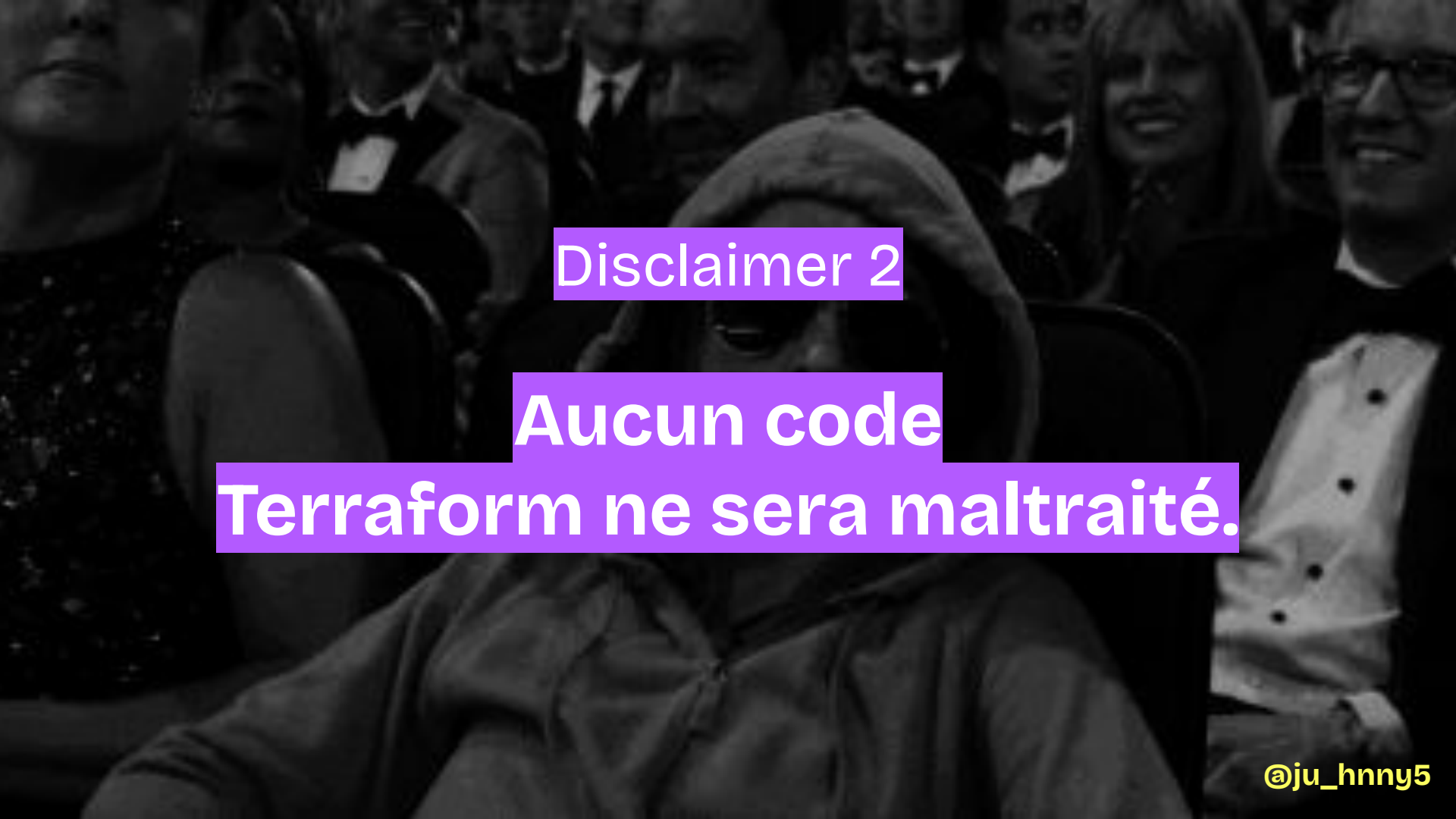
Disclaimer

Ce talk a été produit avant le drama d'Hashicorp et la Business Source Licence.



https://www.youtube.com/watch?v=E_F71Niwq_8

@ju_hnny5


Disclaimer 2
Aucun code
Terraform ne sera maltraité.
@ju_hnny5

Retour aux sources


Dev - Ops
@ju_hnny5


Agile + DevOps
@ju_hnny5


Pet vs Cattle
@ju_hnny5


"Infra is hard..."
@ju_hnny5

Infra as Code ?

L’infra as Code ?

L’*Infrastructure as Code* (IaC) est **une pratique** qui **consiste à gérer et à provisionner des infrastructures informatiques en utilisant** des **fichiers de définition**.

Source : https://shorturl.at/aeoqO

L’infra as Code ?

L’*Infrastructure as Code* apporte plusieurs bénéfices :

1. Elle permet de gérer le *versionning* de l’infrastructure à l’aide de logiciels comme `Git`.

2. Le ***Shadow IT*** est réduit, c'est-à-dire l’ensemble des systèmes informatiques qui ne sont pas officiellement répertoriés.

Source : https://shorturl.at/aeoqO

L’infra as Code ?

Infrastructure Provisioning

=/=

Infrastructure (configuration) Management

L’infra as Code ?

Stateful vs Stateless

L’infra as Code ?

Imperative

Procedural
Structured
Object-oriented

vs

Declarative

Functional
Logic

L’infra as Code ?

Imperative

Step by step instructions

- Create a server
- Add a server
- Make this change

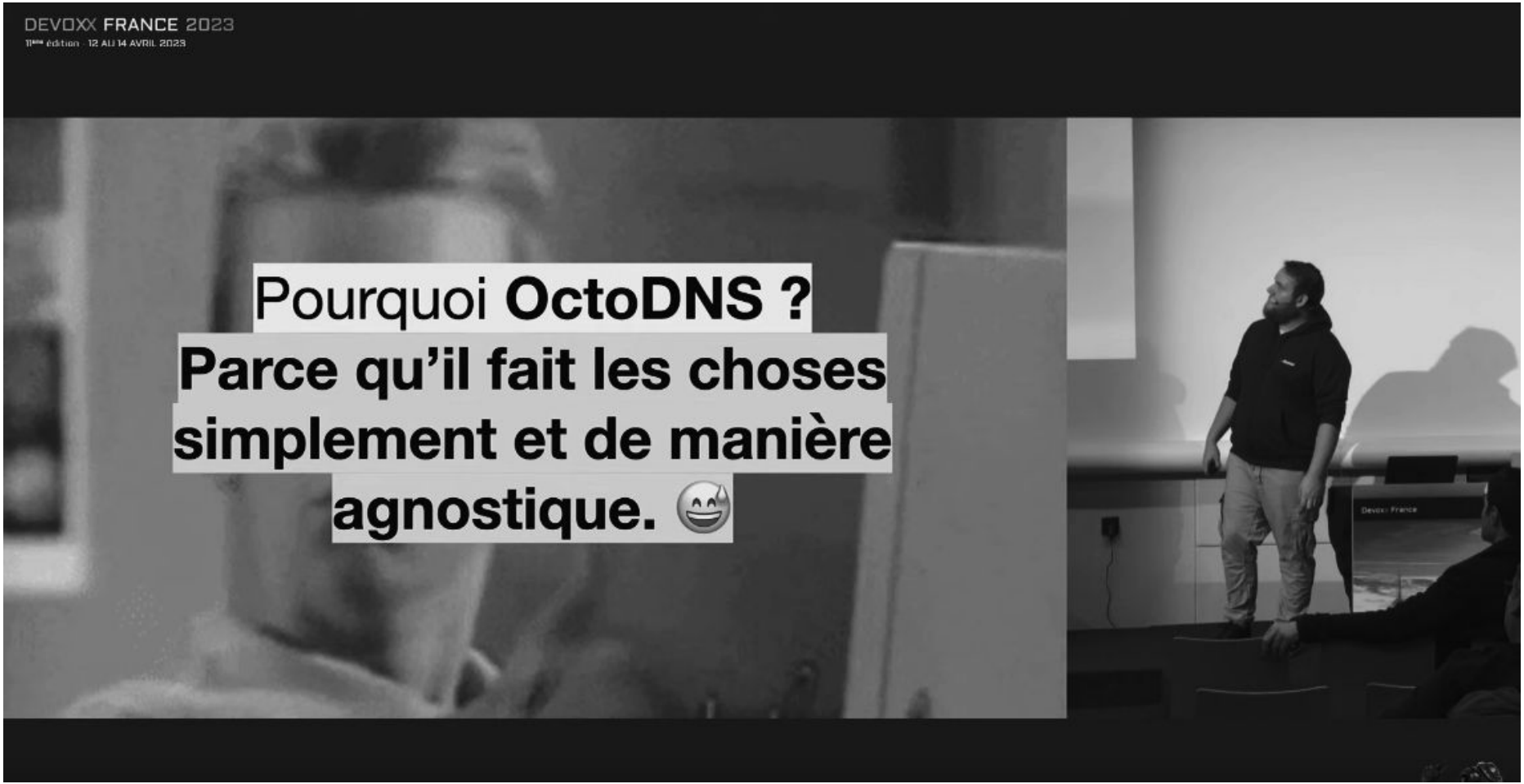
vs

Declarative

Declare and result

- 2 servers

"When devs want to do Ops".



https://www.youtube.com/watch?v=aZvsplOAd7c

On commence ?

DEEZER


Legacy solutions
@ju_hnny5

Legacy Solutions : AWS CloudFormation

Quelques désavantages :

1. Complexité de la syntaxe
2. N’est lié qu’à AWS (obviously)
3. Limitation des modèles
4. Ressources orphelines
    a. Si une mise à jour de template n'inclut pas toutes les ressources existantes, certaines peuvent devenir "orphelines", c'est-à-dire qu'elles ne sont pas supprimées lors de la mise à jour, ce qui peut entraîner des coûts inattendus.

AWS CloudFormation

Legacy Solutions : Azure Bicep

Quelques désavantages :

1. La courbe d'apprentissage
2. N'est lié qu'à Microsoft Azure
3. La maturité
4. Ecosystème et la communauté

Legacy Solutions : OpenStack Heat

Quelques désavantages :

1. La complexité au démarrage (lié principalement au DSL)
2. L'héritage de valeurs
3. Le système de modules

Legacy Solutions : Terraform

Quelques désavantages :

1. La complexité au démarrage (lié principalement au DSL)
2. L'héritage de valeurs
   a. Le système de modules
3. Lisible mais logique de dev difficilement applicable

HashiCorp
Terraform

Le combat que tout oppose

@ju_hnny5

Le standard de facto

Le petit nouveau*

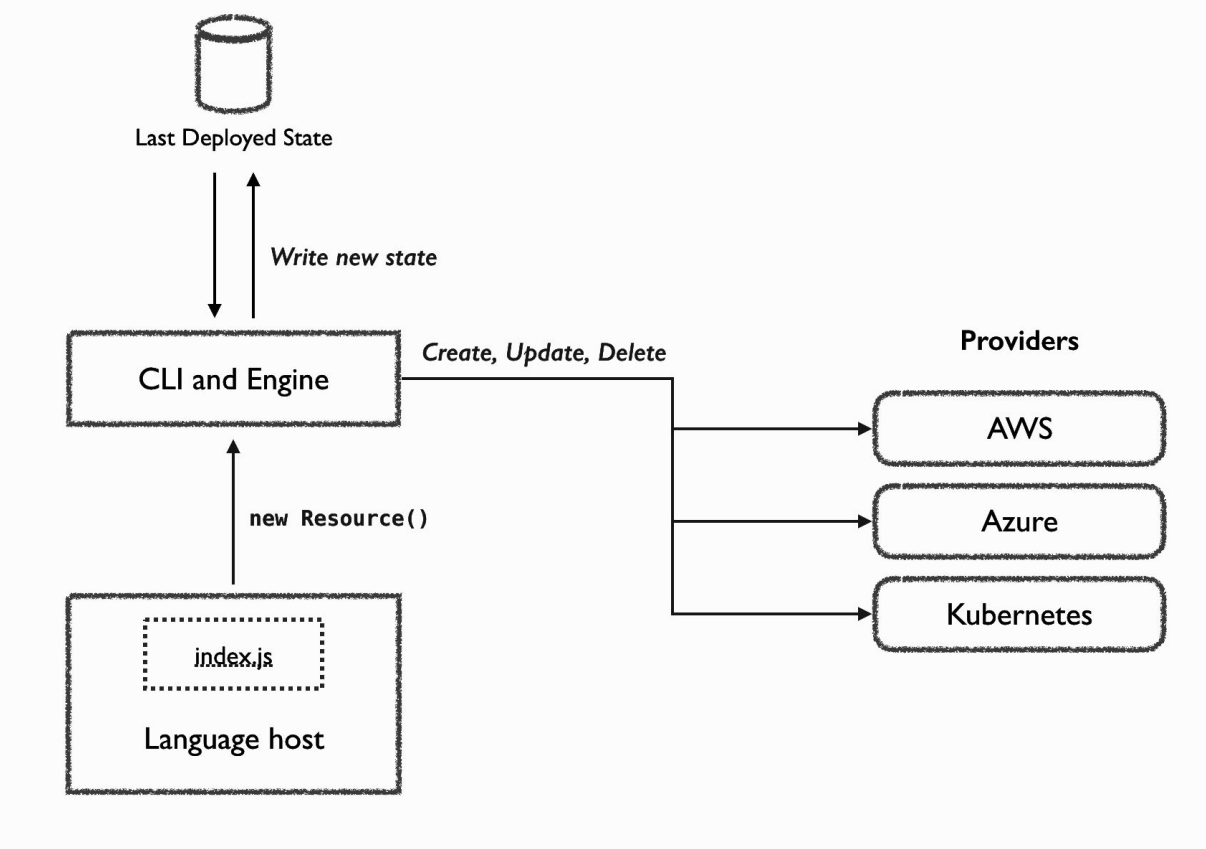
@ju_hnny5

Pulumi, Késako ?

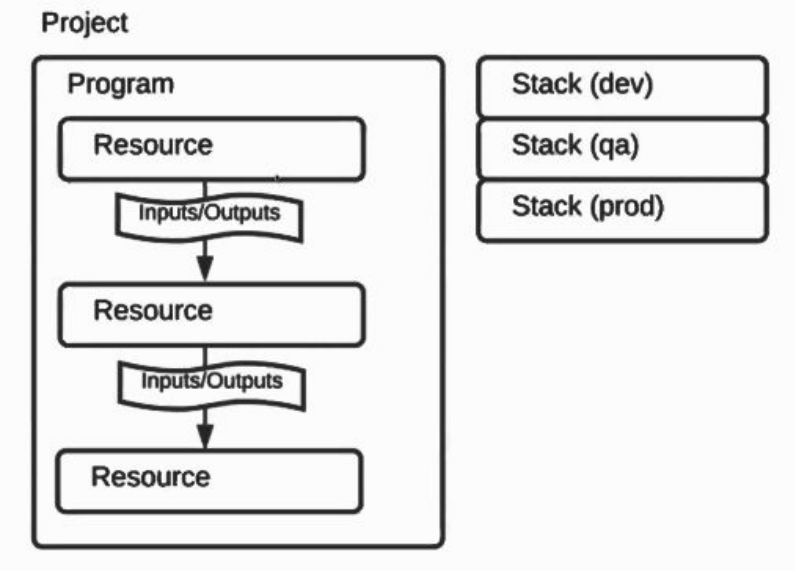
- La première version stable est sortie le **3 septembre 2019**.
  - Projet très jeune par rapport à Terraform sorti en 2014
- Projet écrit en **Go**(lang)
- Possède une communauté large et active
- Beaucoup de tooling fournit directement par l'outil
  - Vs Terraform où beaucoup d'améliorations sont apportées par des outils tiers (exemple Terragrunt)
- Pour bien commencer :
  - https://www.pulumi.com/docs/get-started/

Pulumi

Pulumi, Késako ?



Pulumi, Késako ?



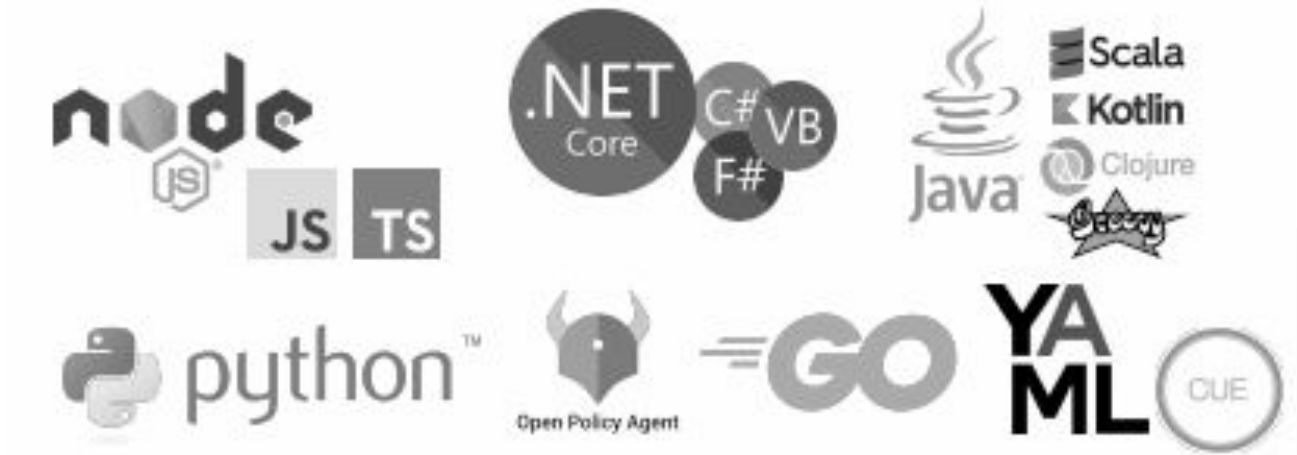
Infra as Code : Apprendre de nouveau langages ?

DevXP


eader(); ?>
iv class="container">
<hl class="te
xt-prima
info"></i>
<hr>
<div cla
<?ph
@ju_hnny5

DevXP : Les langages supportés

A l'inverse de Terraform qui propose d'utiliser son DSL, **Pulumi** propose un SDK permettant d'utiliser plusieurs langages comme :



DevXP : Le code

```
  resource "openstack_compute_instance_v2" "basic" {
     name = "basic"
     image_name = "Ubuntu Server 22.04"
     flavor_name = "m1.small"
     key_pair = "jbriault"
     security_groups = ["default"]
     network { name = "tenant_network" }
}
```

Terraform
HCL

```
instance = openstack.compute.Instance("basic",
   flavor_name="m1.small",
   image_name="Ubuntu Server 22.04",
   key_pair="jbriault",
   security_groups=["default"],
   networks=[{"name": "tenant_network"}],
)
```

Pulumi
Python

L’Infrastructure as Code, bah… c’est du code !

@ju_hnny5

DevXP : Les langages supportés

Des ressources intéressantes :

- https://www.pulumi.com/docs/concepts/vs/terraform/
- https://www.youtube.com/watch?v=PqAP4BunQZU

Pulumi pour les utilisateurs de Terraform

@ju_hnny5

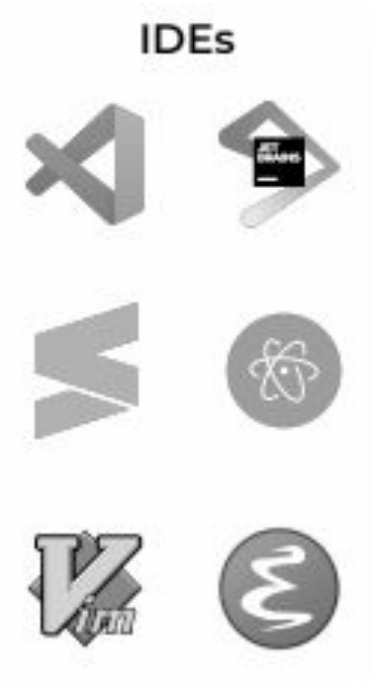
DevXP : **L’outillage**

1. Utiliser son environnement de dev préféré
   a. Pas besoin d’IDE ou de plugin spécifique pour Pulumi

2. Mode pour de la CICD (via l’option `-non-interactive`) **directement intégré** à la CLI Pulumi

@ju_hnny5

DevXP : **L'outillage**

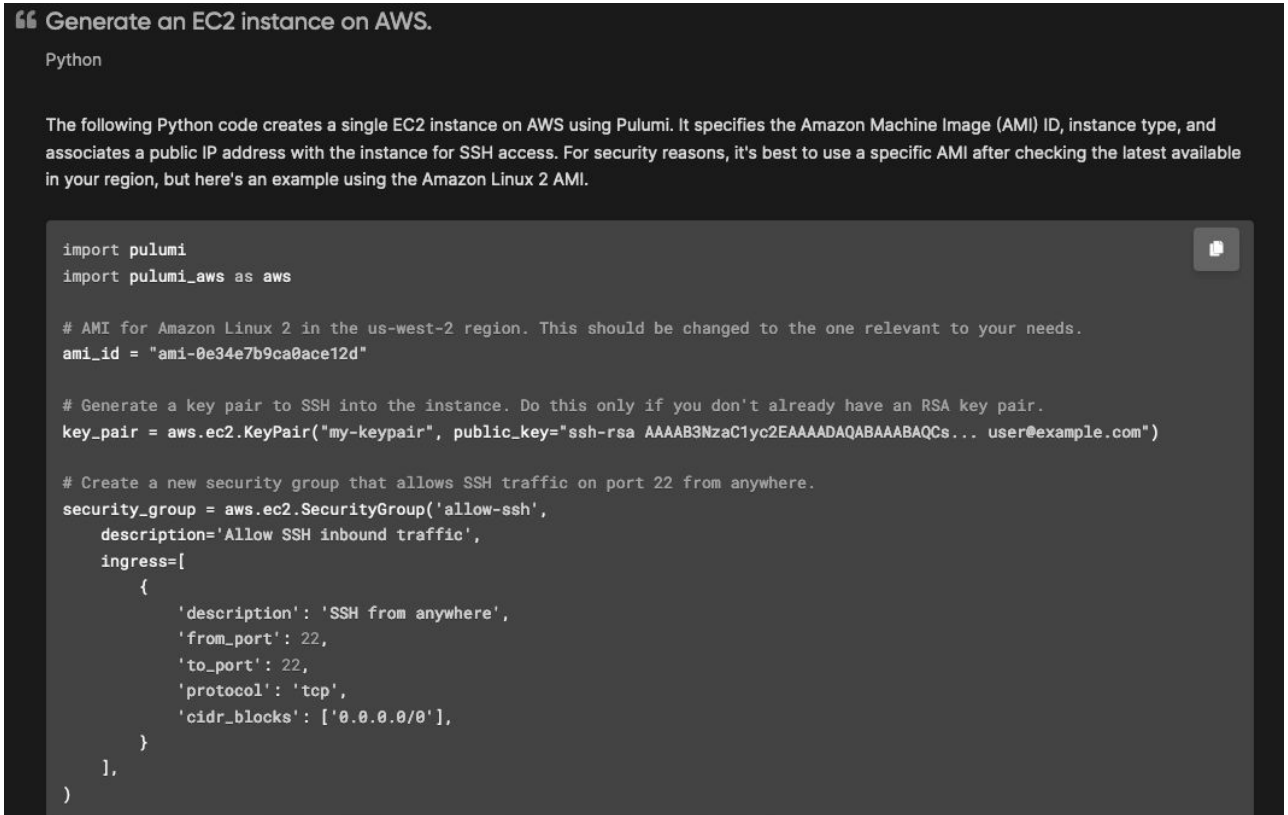


Source : https://shorturl.at/nHSV7

@ju_hnny5

DevXP : L’outillage

L’IA au service du bien.



@ju_hnny5

DevXP : La modularité

1. Vous pouvez organiser votre code en modules réutilisables, ce qui facilite la création et la gestion de configurations d'infrastructure complexes.

2. Vous pouvez facilement découper votre code sous forme de modules ou non (à l'inverse de Terraform qui n'offre qu'un système de modules).

3. Pas besoin d'outil comme `terragrunt` pour hériter de valeurs.

DevXP : Une couche d’abstraction

Pulumi permet d’utiliser les mécanismes de chaque langage.

Ainsi il est aisé d’apporter de l’abstraction sur certains éléments de son infrastructure.

DevXP : Une couche d'abstraction

Exemple, créer des objets sur un pare-feu FortiGate :

```
for key, value in content_services.items():
                if value['type'] == 'TCP':
                serviced = forti.FirewallServiceCustom(
                key,
                app_service_type="disable",
                category="General",
                check_reset_range="default",
                color=0,
                helper="auto",
                iprange="0.0.0.0",
                name=key,
                protocol="TCP/UDP/SCTP",
                protocol_number=6,
                proxy="disable",
                tcp_halfclose_timer=0,
                tcp_halfopen_timer=0,
                tcp_portrange=value['port_range'],
                tcp_timewait_timer=0,
                udp_idle_timer=0,
                visibility=value['visibility'],
                )
                elif value['type'] == 'UDP':
                serviced = forti.FirewallServiceCustom(
                key,
                app_service_type="disable",
                category="General",
                check_reset_range="default",
                color=0,
                helper="auto",
                iprange="0.0.0.0",
                name=key,
                protocol="TCP/UDP/SCTP",
                protocol_number=6,
                proxy="disable",
                udp_halfclose_timer=0,
                udp_halfopen_timer=0,
                udp_portrange=value['port_range'],
                udp_timewait_timer=0,
                udp_idle_timer=0,
                visibility=value['visibility'],
                )
                elif value['type'] == 'SCTP':
                serviced = forti.FirewallServiceCustom(
                key,
                app_service_type="disable",
                category="General",
                check_reset_range="default",
                color=0,
                helper="auto",
                iprange="0.0.0.0",
                name=key,
                protocol="TCP/UDP/SCTP",
                protocol_number=6,
                proxy="disable",
                sctp_halfclose_timer=0,
                sctp_halfopen_timer=0,
                sctp_portrange=value['port_range'],
                sctp_timewait_timer=0,
```

```
# TCP port range
myport:
  type: "TCP" # or UDP/SCTP
  port_range: "223-332"
  visibility: "enable"
  category: "General"

ssh_port:
  type: "TCP"
  port_range: "22-22"
  visibility: "enable"
  category: "General"
```

https://github.com/juhnny5/pulumi-fortigate-example

@ju_hnny5


The CLI
@ju_hnny5

The CLI : `pulumi`

Une seule ligne de commande pour tout contrôler.

Verbes différents de ceux de Terraform.

- Liste disponible ici : https://www.pulumi.com/docs/concepts/vs/terraform/terminology/#commands

The CLI : `pulumi`

Exemples

Afficher les modifications à apporter :

`pulumi preview`
`terraform plan`

Appliquer les modifications :

`pulumi update`
`terraform apply`

Détruire les ressources :

`pulumi destroy`
`terraform destroy`

The CLI : Une migration facilitée

Une commande :

```
pulumi convert \
-from terraform \
-language python \
-out pulumi
```

Disponibles depuis la version **v3.71.0**.

The CLI : Une migration facilitée

Migrer ses ressources :

```
pulumi import --from terraform ./terraform.tfstate
```

Quelques ressources pour aider à la migration :

- **Migration Hub**
  - https://www.pulumi.com/blog/migration-hub/
  - https://www.pulumi.com/migrate/

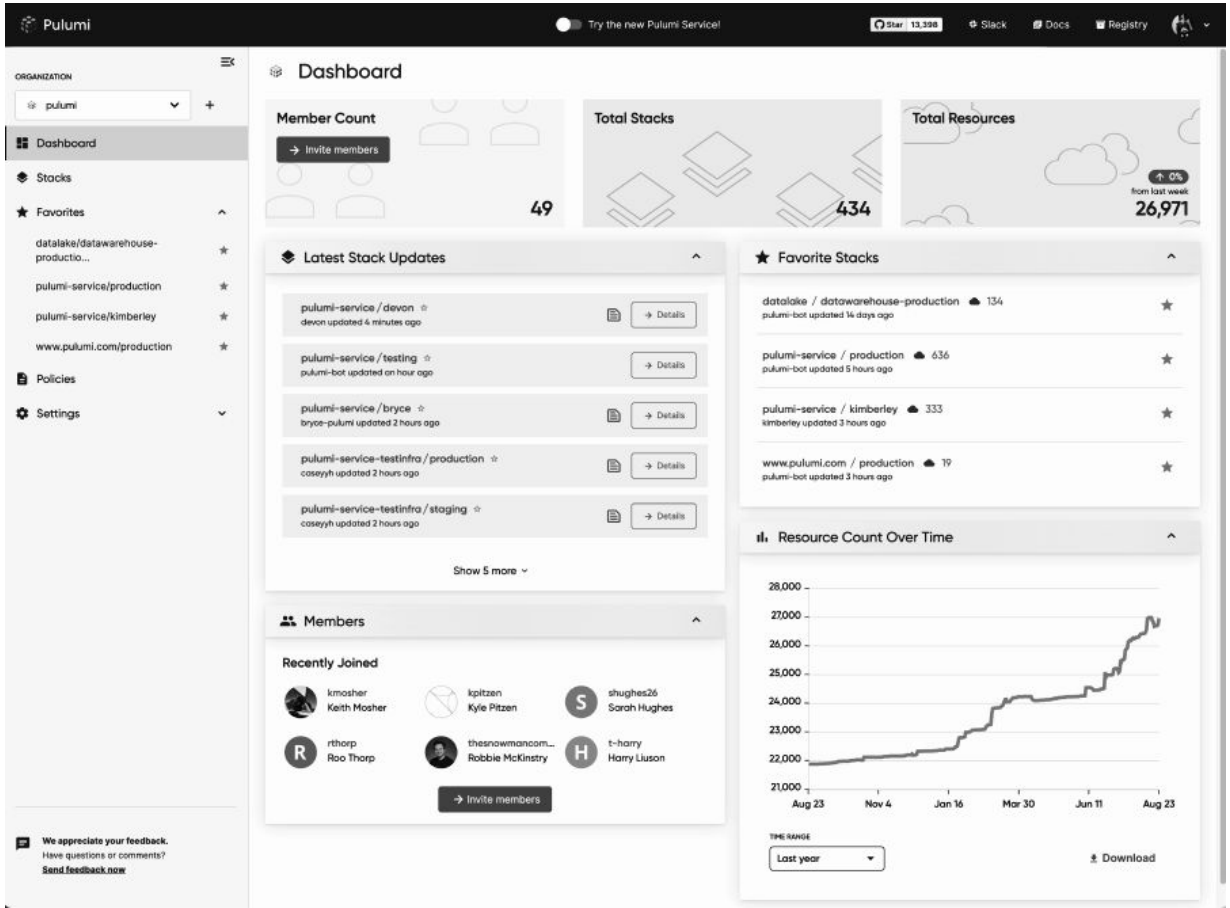
Pulumi Cloud Console

- Gestion de l'état de déploiement et des secrets dans **Pulumi Cloud** par défaut.
- Permet l'exécution à distance des déploiements.
- S'intègre avec votre CI/CD.
- Gestion des politiques/contrôles d'accès.

> Self-hosting is only available with **Pulumi Business Critical**. If you would like to evaluate the Self-Hosted Pulumi Cloud, sign up for the 30 day trial or contact us.
>
> To manage your state with a self-managed backend, such as a cloud storage bucket, see State and Backends.

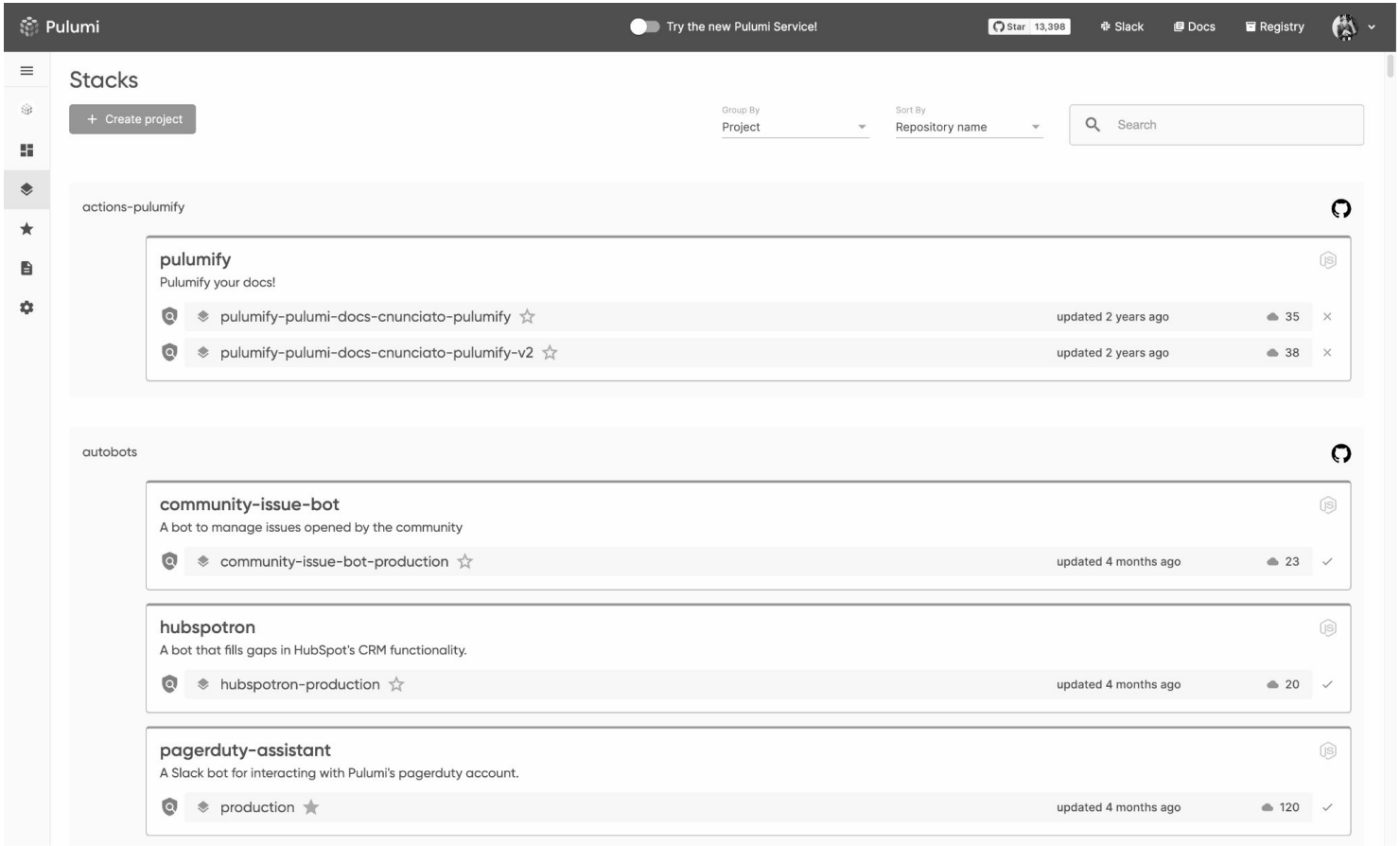
https://www.pulumi.com/docs/pulumi-cloud/self-hosted/components/console/

Pulumi Cloud Console



@ju_hnny5

Pulumi Cloud Console



@ju_hnny5

Pulumi ESC

(Environment, Secrets and Configuration)

Pulumi ESC

- Permet aux équipes d'agréger des secrets et des configurations provenant de nombreuses sources, de gérer des collections hiérarchiques de configurations et de secrets ("environnements"), et de consommer ces configurations et secrets à partir d'une variété de services d'infrastructure et d'application différents.

- Peut fonctionner avec mais aussi sans Pulumi.

https://www.pulumi.com/docs/esc/get-started/


Terraform Bridge
@ju_hnny5

Pulumi Terraform Bridge

Adapter n'importe quel provider Terraform (construit à l'aide du Terraform Plugin SDK) en provider provider **Pulumi**.

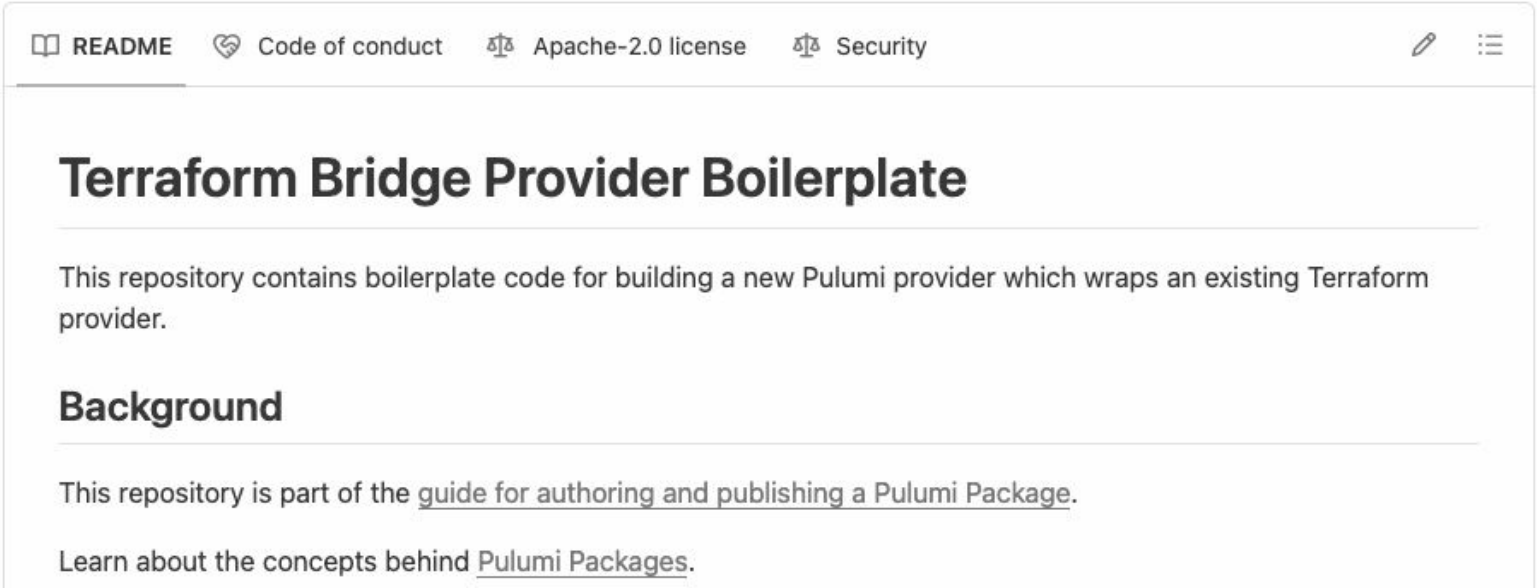
Les providers Terraform font des opérations **CRUD***, il est donc "facile" de les transposer en provider Pulumi.

Exemple :
https://github.com/juhnny5/pulumi-maas-python

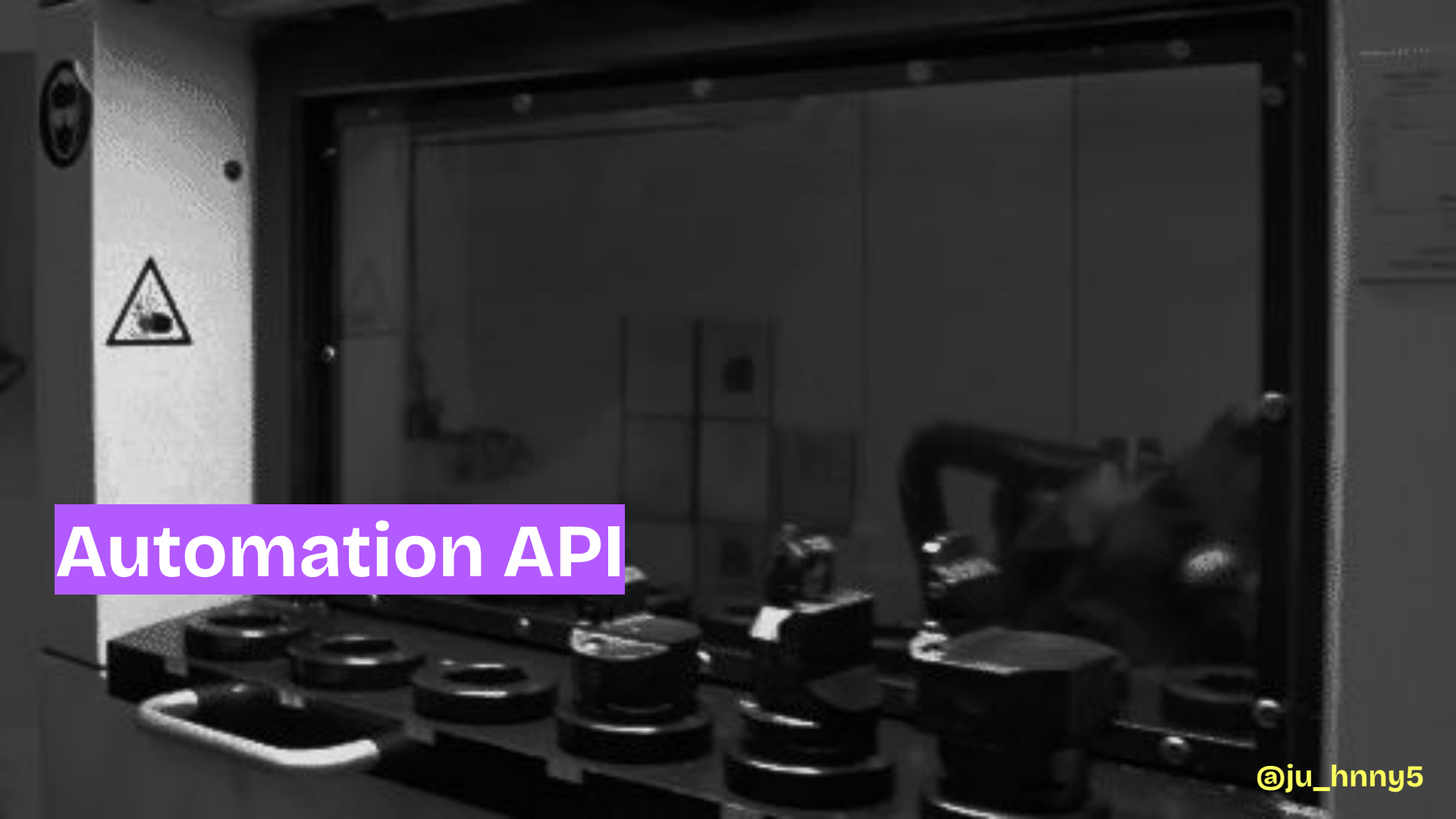
*Create, Read, Update, Delete

Pulumi Terraform Bridge

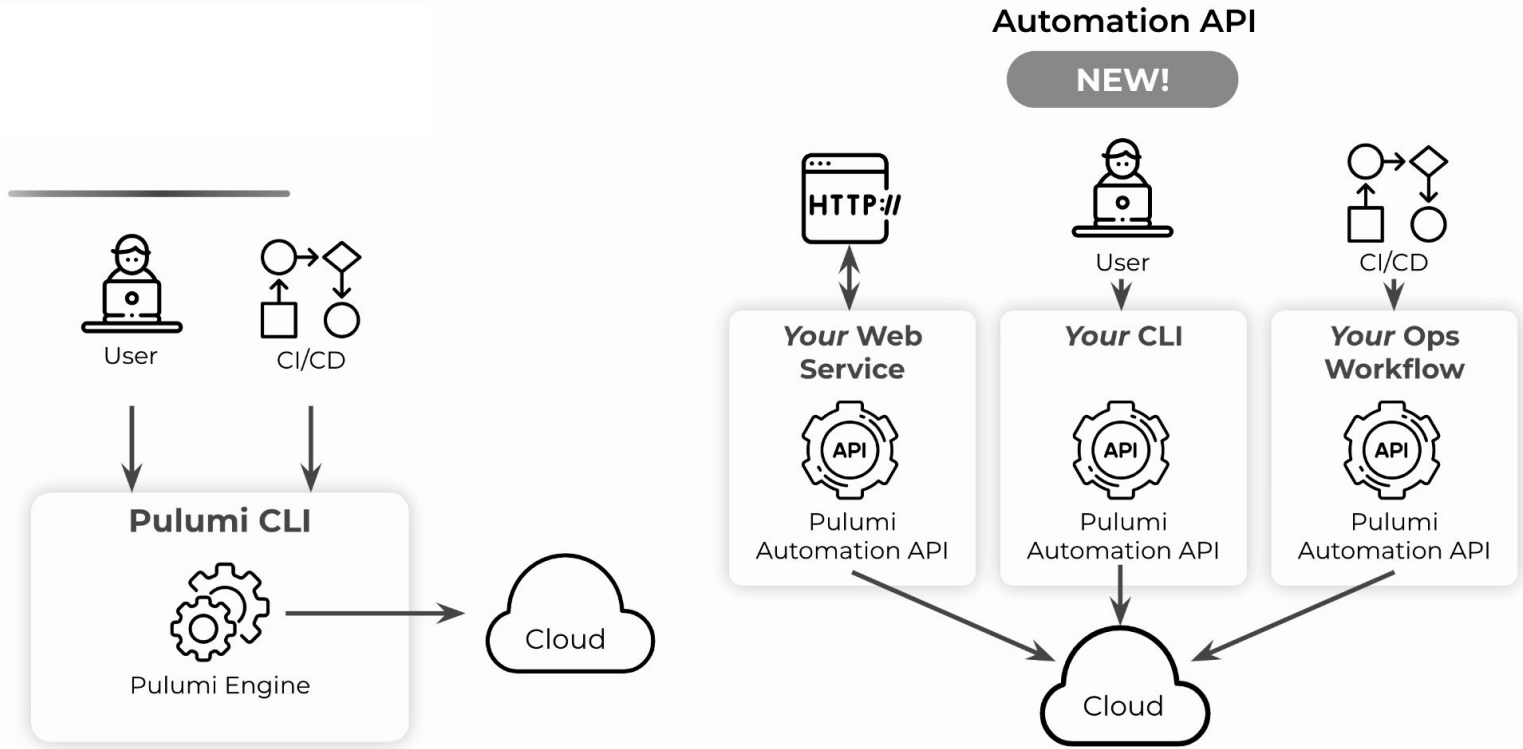
https://github.com/pulumi/pulumi-tf-provider-boilerplate



@ju_hnny5


Automation API
@ju_hnny5

Pulumi Automation API : **Intégrations et workflows automatisés**



@ju_hnny5


Community
@ju_hnny5

Community

- https://slack.pulumi.com/
- https://github.com/pulumi
- https://twitter.com/PulumiCorp
- Calendrier des conférences / Meetup : https://docs.google.com/spreadsheets/d/1TXphpxOOJN87ptH39JdSv32nzoTpBmhE2HhOGUAE4yw/edit


Demo time !
@ju_hnny5

Démo 1 : Créer une première instance sur OpenStack (avec Python)


Démo 2 : Créer une première instance sur OpenStack (avec Python) et comparer avec Terraform
@ju_hnny5

Démo 3 : Créer des ressources en apportant une couche d'abstraction (avec Python)

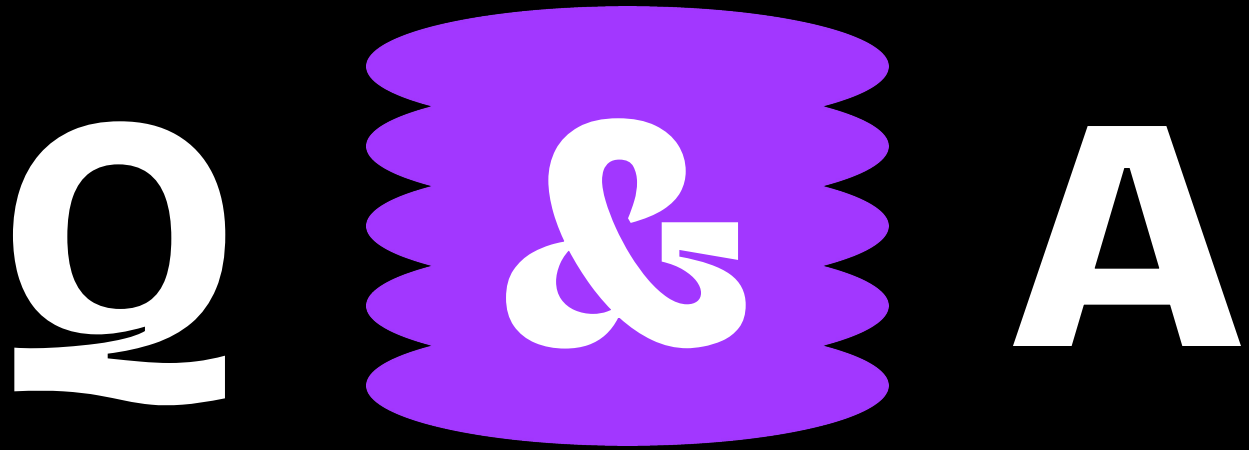
@ju_hnny5

Démo 4 : Créer des **ressources** (avec YAML)


Voila`
@ju_hnny5

One link : **https://t.ly/wxK0Q**

DEEZER


Q
&
A

Thank you

DEEZER

@ju_hnny5