glueck kanja

Marvin Bangert, CollabDays Bremen, 22.02.2025

Terrafying Power Platform

Power Platform - Infrastructure as Code

CollabDays Bremen 2025
Power Platform Bootcamp North Germany

Bremen, Germany – February 21-22, 2025

POWER PLATFORM BOOTCAMP

HEC
ECHTE MENSCHEN.
DIGITALE LÖSUNGEN.

GEDL
make your work flow
1010
1010

CLOVERFIELD
IT & COMMUNICATIONS

OASE

THANK YOU!

glueck kanja

Marvin Bangert

Cloud Architect mit Schwerpunkt auf Modern Collaboration und Power Platform

- Architektur, Migration und Betrieb
- Modern Collaboration Lösungen
- Microsoft Power Platform Lösungen
- Power Platform User Group Cologne Organisator
- Power Automate Community Super User

.../marvin-bangert/

Microsoft® Most Valuable Professional

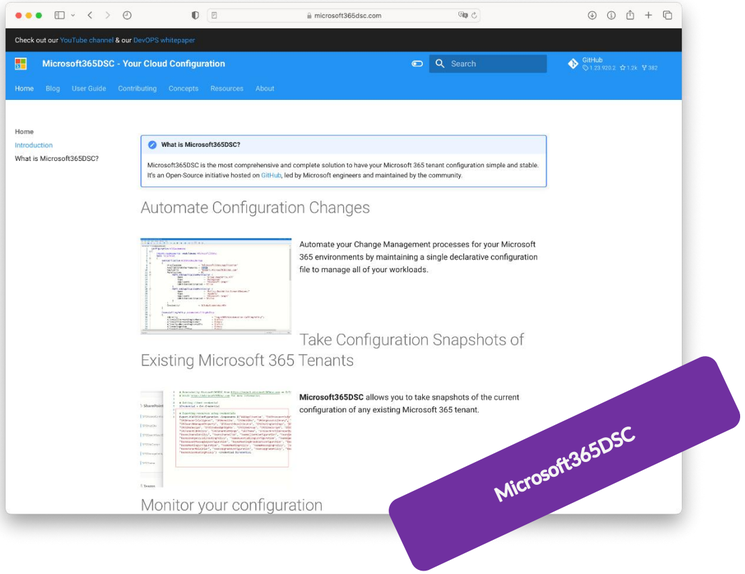
Which problem are we trying to solve?

- Power Platform configuration is a complex beast
- If you are alone and have just one tenant, you may be ok with 'clicks in portals', but ...
- If you are a **team of admins**, have **multiple environments or tenants** (staging/production | MSP) to manage, **'clicks in portals' is not a scalable option**.
  - documentation
  - change tracking / versioning
  - auditing
  - detect manual changes
  - blueprint environment creation / automation

glueck kanja

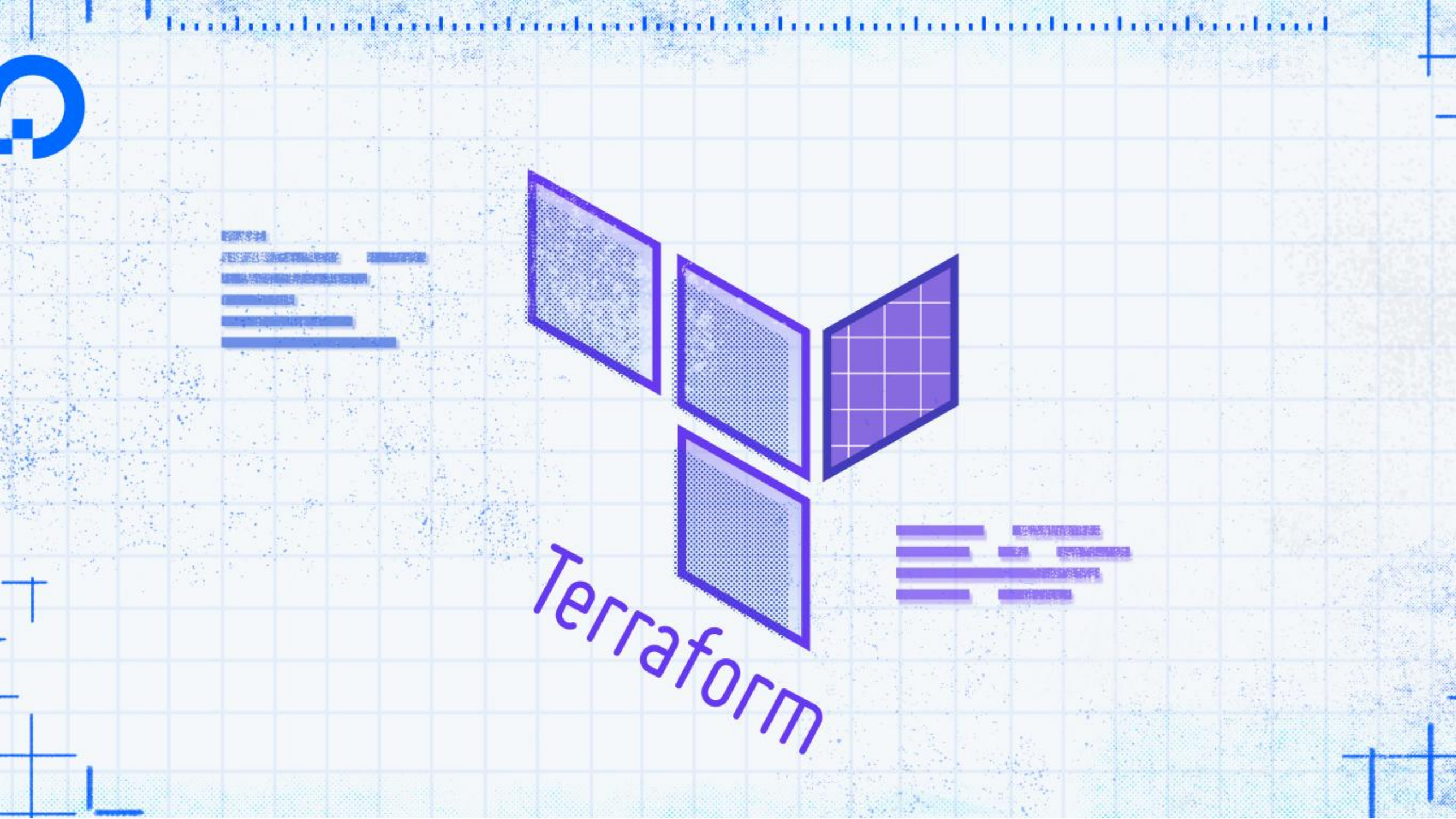
There are already solutions for this problem

- The general idea is called **Configuration-as-Code**
  (or Desired State Configuration / Infrastructure-as-Code)
- If you search for it, you'll find an amount of community projects
- Some are kind of creative, some are kind of professional including DevOps integration



glueck kanja

We ❤️ Powershell – but…


Terraform

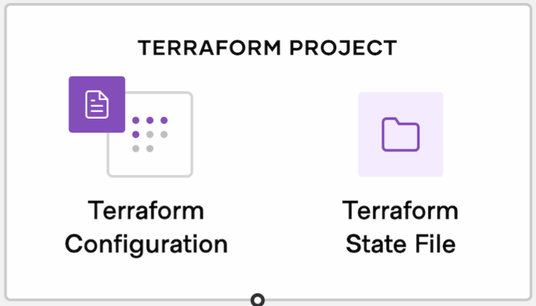
glueck kanja

Terraform Basics

HashiCorp Terraform

Write

Define infrastructure in configuration files



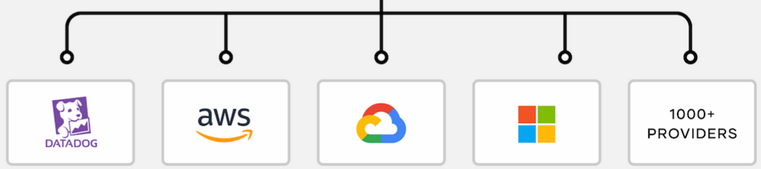
Plan

Review the changes Terraform will make to your infrastructure

```
$ terraform plan
...
Terraform will perform
the following actions
```

Apply

Terraform provisions your infrastructure and updates the state file.



glueck kanja

Terraform Configuration Language (.tf)

```
terraform {
  required_providers {
    powerplatform = {
      source = "microsoft/power-platform“
      version = "3.0.0"
    }
  }
}

provider "powerplatform" {
  use_cli = true
}

resource "powerplatform_environment" "development" {
  display_name     = "example_environment"
  location         = "europe"
  environment_type = "Sandbox"
  dataverse = {
    language_code     = "1033"
    currency_code     = "USD"
    security_group_id = "00000000-0000-0000-0000-000000000000"
  }
}
```

glueck kanja

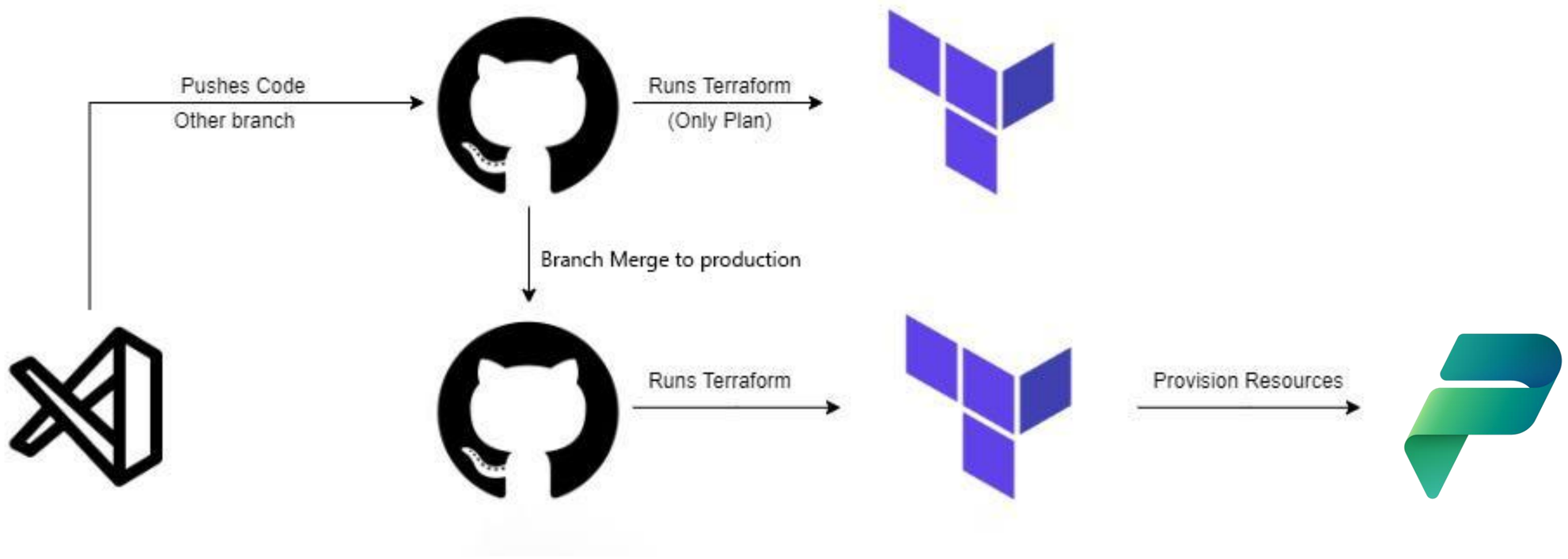
Terraform Command Line

```
# Initialize Terraform
terraform init

# Example Output:
# Initializing the backend...
# Initializing provider plugins...
# Terraform has been successfully initialized!

# Generate an execution plan

terraform plan

# Example Output:
# Refreshing Terraform state in-memory prior to plan...
# ...
# Plan: 2 to add, 0 to change, 0 to destroy.
```

```
# Apply the configuration
terraform apply

# Example Output:
# ...
# ...
# Apply complete! Resources: 2 added, 0 changed, 0 destroyed.

# Destroy resources (optional)

terraform destroy

# Example Output:
# ...
# ...
# Destroy complete! Resources: 2 destroyed.
```

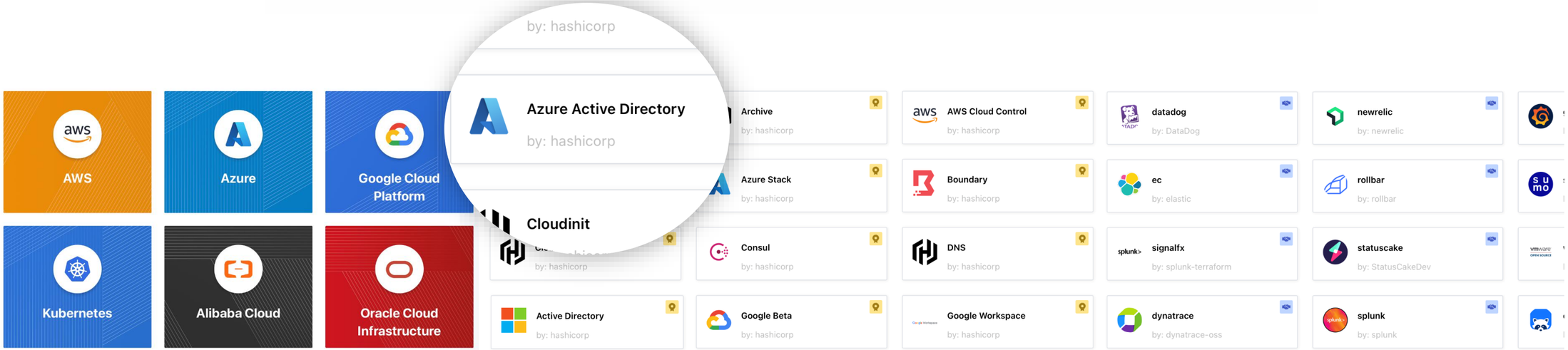
glueck kanja

Basic Terraform Lifecycle



glueck kanja

Terraform Provider



Prerequisites

glueck kanja

Create an App Registration to use the Power Platform Provider

Configured permissions

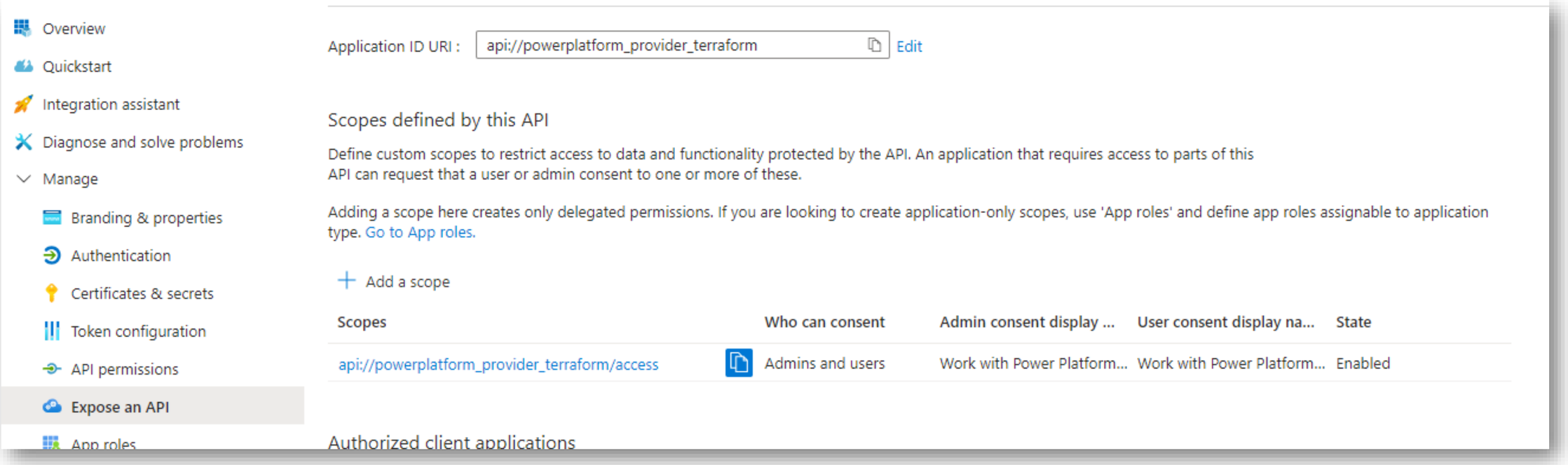
Applications are authorized to call APIs when they are granted permissions by users/admins as part of the consent process. The list of configured permissions should include all the permissions the application needs. Learn more about permissions and consent

+ Add a permission ✓ Grant admin consent for C4A8 Gallifrey

API / Permissions name	Type	Description	Admin consent req...	Status
Dynamics CRM (1)				
user_impersonation	Delegated	Access Common Data Service as organi...	No	Granted for C4A8 Gallifr...
Power Platform API (8)				
AppManagement.ApplicationPackages.Install	Delegated	Install Application Packages	No	Granted for C4A8 Gallifr...
AppManagement.ApplicationPackages.Read	Delegated	Read Application Packages	No	Granted for C4A8 Gallifr...
Licensing.Allocations.Read	Delegated	Read Currency Allocations	No	Granted for C4A8 Gallifr...
Licensing.Allocations.ReadWrite	Delegated	Read and Write Currency Allocations	No	Granted for C4A8 Gallifr...
Licensing.BillingPolicies.Read	Delegated	Read Billing Policies	No	Granted for C4A8 Gallifr...
Licensing.BillingPolicies.ReadWrite	Delegated	Read and Write Billing Policies	No	Granted for C4A8 Gallifr...
Licensing.IsvContracts.Read	Delegated	Read ISV Contracts	No	Granted for C4A8 Gallifr...
Licensing.IsvContracts.ReadWrite	Delegated	Read and Write ISV Contracts	No	Granted for C4A8 Gallifr...
PowerApps Service (1)				
User	Delegated	Access the PowerApps Service API	No	Granted for C4A8 Gallifr...

glueck kanja

Create an App Registration to use the Power Platform Provider



Register your app registration with Power Platform

PowerShell

```
Install-Module -Name Microsoft.PowerApps.Administration.PowerShell
Add-PowerAppsAccount
New-PowerAppManagementApp –ApplicationId $ApplicationId
```

PowerShell

```
Authorization: Bearer eyJ0eXAiOi...
Host: api.bap.microsoft.com
Accept: application/json
PUT
https://api.bap.microsoft.com/providers/Microsoft.BusinessAppPlatform/adminApplications/{CLIENT_ID
_FROM_AZURE_APP}?api-version=2020-10-01
```

Create an App Registration to use the Power Platform Provider

Certificates (0) Client secrets (0) **Federated credentials (2)**

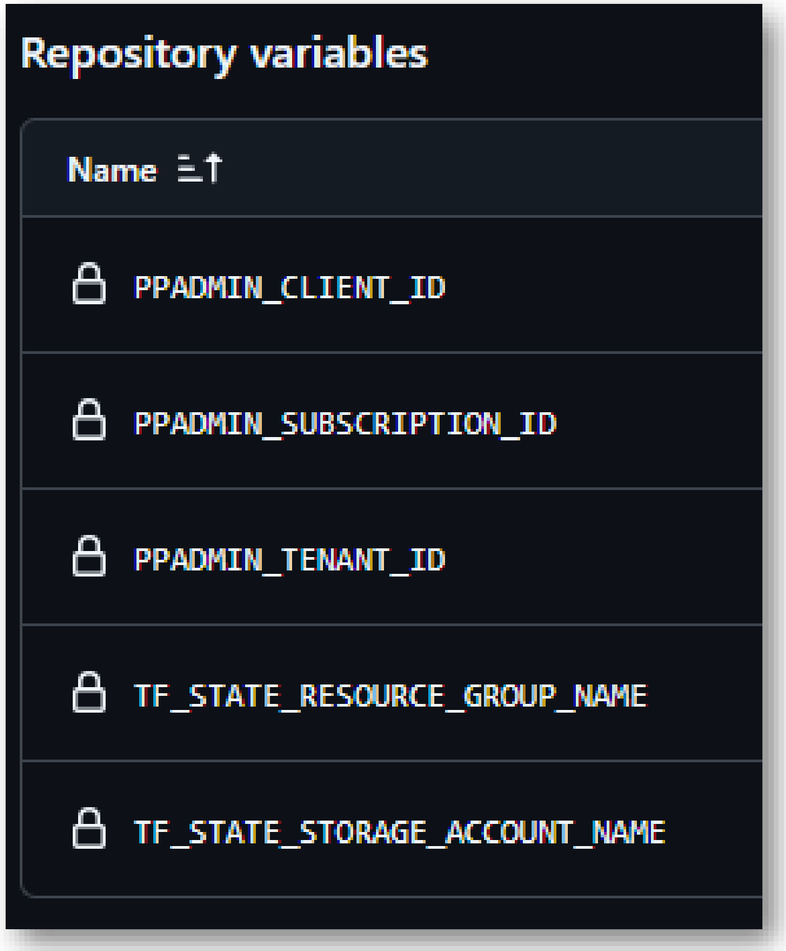
Allow other identities to impersonate this application by establishing a trust with an external OpenID Connect (OIDC) identity provider. This federation allows you to get tokens to access Microsoft Entra ID protected resources that this application has access to like Azure and Microsoft graph. Learn more

+ Add credential

Name	Description	Subject Identifier
TF-plan		repo:MarvinBangert/Power-Platform-Terraform:pu...
TF-apply		repo:MarvinBangert/Power-Platform-Terraform:re...

glueck kanja

Create an App Registration to use the Power Platform Provider



Demo

glueck kanja

Example Structure

```
terraform/
├── modules/
│   ├── power_platform_environment/
│   │   ├── main.tf
│   │   ├── variables.tf
│   │   ├── outputs.tf
│   ├── dlp_policy/
│       ├── main.tf
│       ├── variables.tf
│       ├── outputs.tf
├── environments/
│   ├── dev/
│   │   ├── main.tf
│   │   ├── terraform.tfvars
│   ├── test/
│   │   ├── main.tf
│   │   ├── terraform.tfvars
│   ├── prod/
│       ├── main.tf
│       ├── terraform.tfvars
├── providers.tf
├── backend.tf
└── variables.tf
```

```
# terraform/modules/power_platform_environment/main.tf
provider "microsoftpowerplatform" {
  # Add provider configuration if needed
}
resource "powerplatform_environment" "environment“ {
  display_name         = var.display_name
  location             = var.location
  environment_type     = var.environment_type
  dataverse = {
    language_code     = var.language_code
    currency_code     = var.currency_code
    security_group_id = var.environment_access_group_id
  }
output "environment_id" {
  value = powerplatform_environment.environment.id
}
```

glueck kanja

Example Structure

```
terraform/
├── modules/
│   ├── power_platform_environment/
│   │   ├── main.tf
│   │   ├── variables.tf
│   │   ├── outputs.tf
│   ├── dlp_policy/
│       ├── main.tf
│       ├── variables.tf
│       ├── outputs.tf
├── environments/
│   ├── dev/
│   │   ├── main.tf
│   │   ├── terraform.tfvars
│   ├── test/
│   │   ├── main.tf
│   │   ├── terraform.tfvars
│   ├── prod/
│       ├── main.tf
│       ├── terraform.tfvars
├── providers.tf
├── backend.tf
└── variables.tf
```

```
# terraform/modules/power_platform_environment/variables.tf
variable “display_name" {
    type = string
}
variable “location" {
    type = string
}
variable "environment_type" {
    type = string
}
variable “language_code" {
    type = string
}
...
```

glueck kanja

Example Structure

```
terraform/
├── modules/
│   ├── power_platform_environment/
│   │   ├── main.tf
│   │   ├── variables.tf
│   │   ├── outputs.tf
│   ├── dlp_policy/
│       ├── main.tf
│       ├── variables.tf
│       ├── outputs.tf
├── environments/
│   ├── dev/
│   │   ├── main.tf
│   │   ├── terraform.tfvars
│   ├── test/
│   │   ├── main.tf
│   │   ├── terraform.tfvars
│   ├── prod/
│       ├── main.tf
│       ├── terraform.tfvars
├── providers.tf
├── backend.tf
└── variables.tf
```

```
# terraform/modules/dlp_policy/main.tf
resource "microsoftpowerplatform_dlp_policy" "dlp" {
  name = var.policy_name
  environment_id = var.environment_id
  data_groups = var.data_groups
  connectors = var.connectors
}
output "dlp_policy_id" {
  value = microsoftpowerplatform_dlp_policy.dlp.id
}
```

glueck kanja

Example Structure

```
terraform/
├── modules/
│   ├── power_platform_environment/
│   │   ├── main.tf
│   │   ├── variables.tf
│   │   ├── outputs.tf
│   ├── dlp_policy/
│       ├── main.tf
│       ├── variables.tf
│       ├── outputs.tf
├── environments/
│   ├── dev/
│   │   ├── main.tf
│   │   ├── terraform.tfvars
│   ├── test/
│   │   ├── main.tf
│   │   ├── terraform.tfvars
│   ├── prod/
│       ├── main.tf
│       ├── terraform.tfvars
├── providers.tf
├── backend.tf
└── variables.tf
```

```
# terraform/modules/dlp_policy/variables.tf
variable "policy_name" { type = string }
variable "environment_id" { type = string }
variable "data_groups" { type = map(any) }
variable "connectors" { type = map(any) }
```

glueck kanja

Example Structure

```
terraform/
├── modules/
│   ├── power_platform_environment/
│   │   ├── main.tf
│   │   ├── variables.tf
│   │   ├── outputs.tf
│   ├── dlp_policy/
│       ├── main.tf
│       ├── variables.tf
│       ├── outputs.tf
├── environments/
│   ├── dev/
│   │   ├── main.tf
│   │   ├── terraform.tfvars
│   ├── test/
│   │   ├── main.tf
│   │   ├── terraform.tfvars
│   ├── prod/
│       ├── main.tf
│       ├── terraform.tfvars
├── providers.tf
├── backend.tf
└── variables.tf
```

```
# terraform/environments/dev/terraform.tfvars

environment_name = "dev-environment"

location         = "North America"

environment_type = "Sandbox"

display_name     = "Development Environment"

policy_name      = "dev-dlp-policy"

data_groups      = { "business_data" = ["connector1"],
"non_business_data" = ["connector2"] }

connectors       = ["connector1", "connector2"]
```

glueck kanja

Terraform Licensing

HashiCorp updates licensing FAQ based on community questions

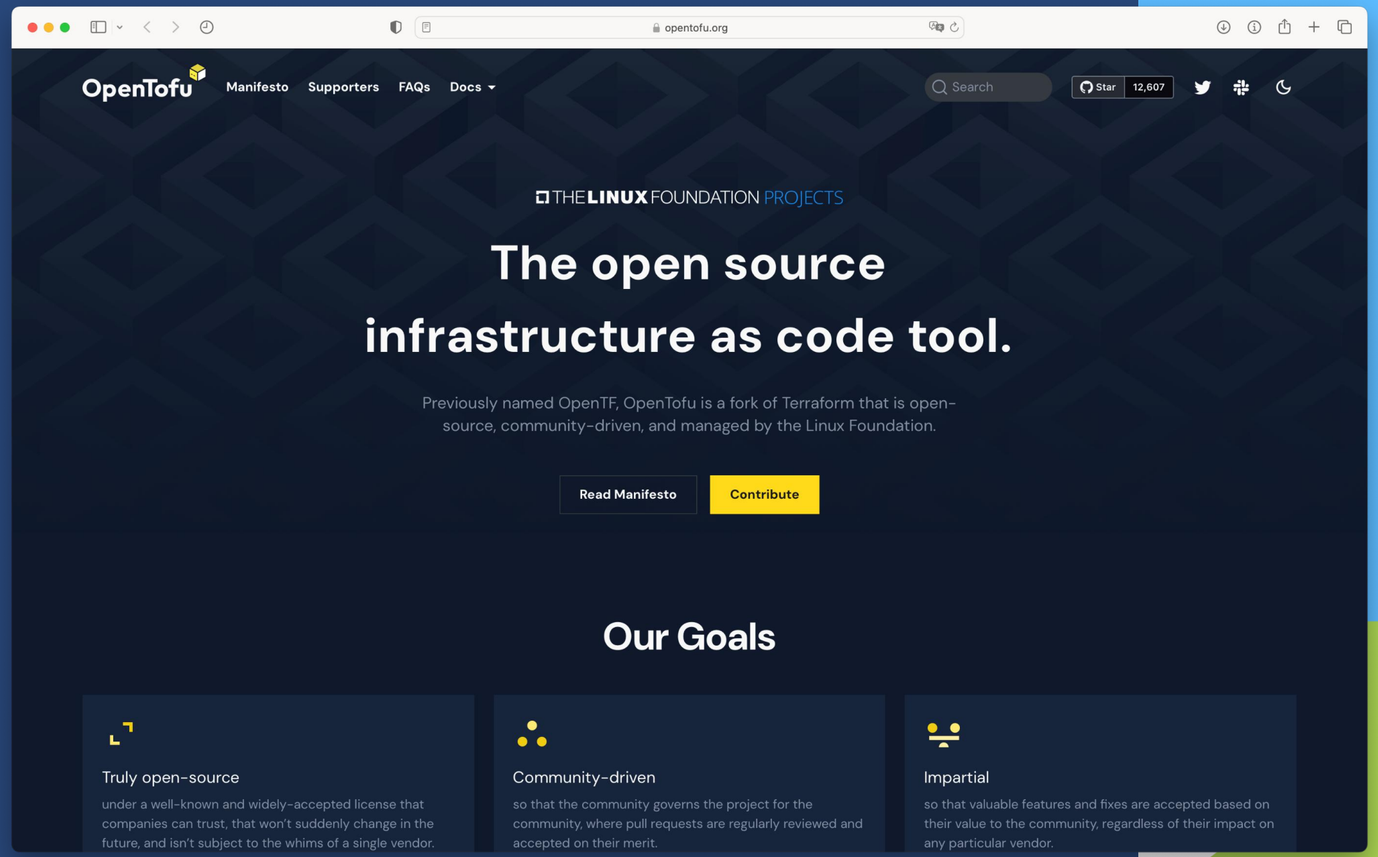
HashiCorp continues to update our licensing FAQ based on questions from the community about our change to the Business Source License for future releases of HashiCorp products.

AUG 21 2023 | ARMON DADGAR

HashiCorp recently announced that we have adopted the Business Source License (BSL, or BUSL) v1.1 for all future releases of HashiCorp products. HashiCorp team members have been answering questions about the licensing change in a thread on our Discuss forum and via our licensing@hashicorp.com email. Based on those questions, we have continued to update our FAQ to provide additional clarity.

Our goal with the BSL license was to make it short and simple. This means the FAQs play an important role in providing interpretive guidance to our users. We view the guidance in these FAQs as binding, so users of our software should feel assured in relying on them as our official positions now and in the future.

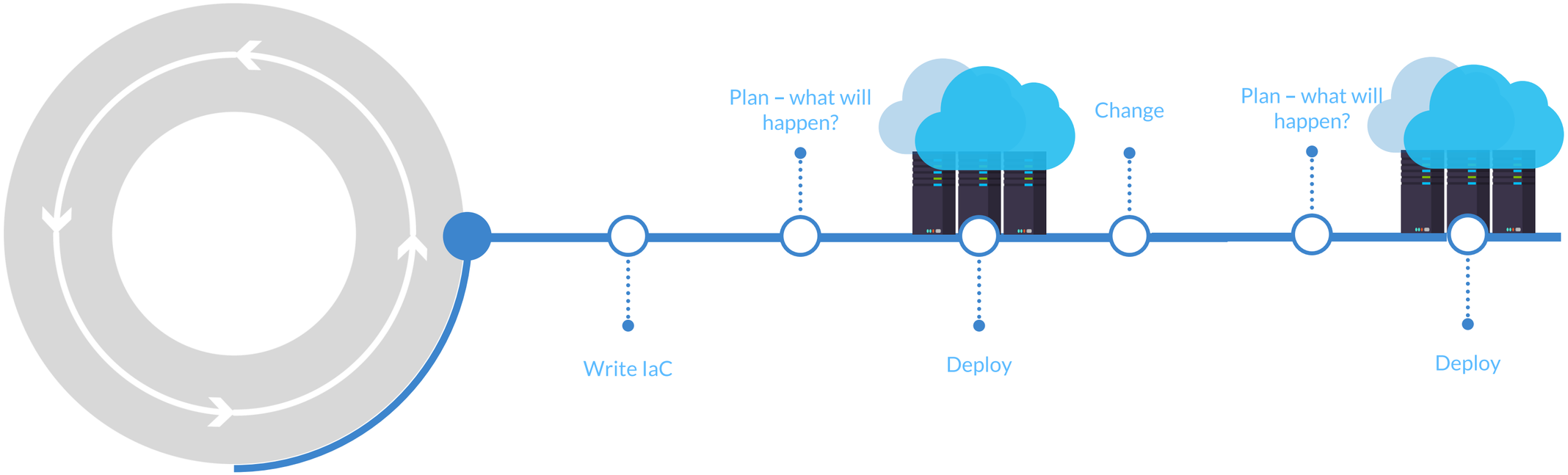
We have added additional questions and answers to the FAQs since August 10 based on



glueck kanja

Summary: Infrastructure as Code

- declarative description of the target infrastructure
- describe what you want in code (desired state configuration)
- write once - deploy many
- documentation of IT estate, standardized deployment model



glueck kanja

Resources to get started

- Power Platform Terraform Provider
- Examples for using the Power Platform Terraform Provider
- Power Platform Terraform Provider
- Terraform Registry Docs

More Providers to check out

- Terraform Provider Microsoft Fabric
- Terraform Provider Azure DevOps
- Terraform Provider AzureRM
- Configuration as Code for Microsoft 365 Policy Management

glueck kanja

Thanks.

Questions?

Socials: @MarvinBangert