Module 7 : Automatisation et écriture de scripts

Formation Azure DevOps CI/CD

Automatisation des tâches avec Azure CLI et PowerShell
Personnalisation et extension d'Azure DevOps avec des scripts et des API

Outils d'automatisation dans Azure DevOps

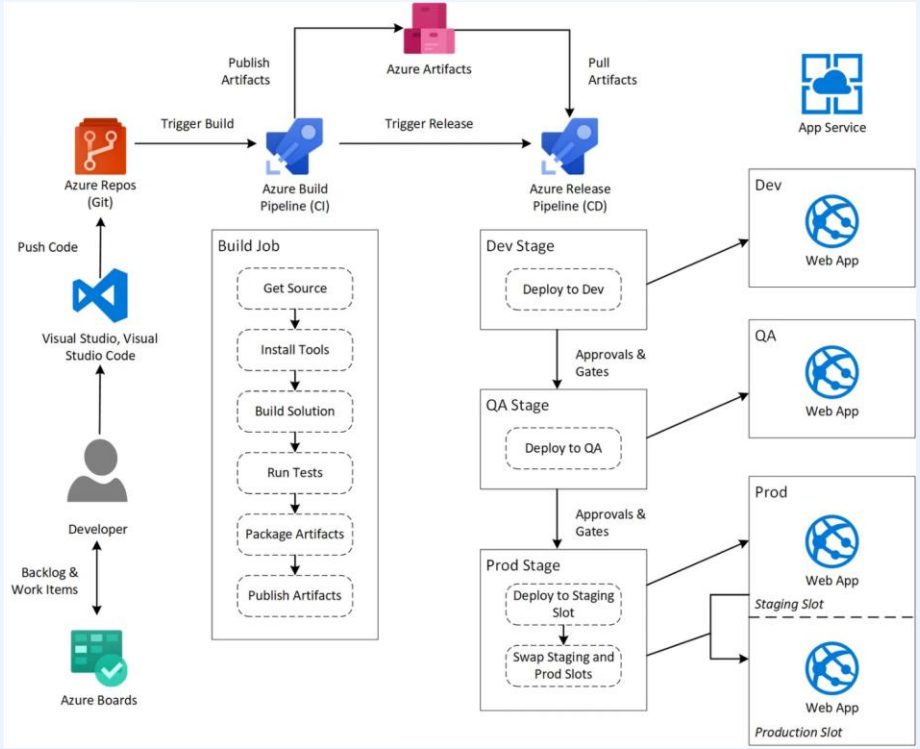
Azure CLI
Interface en ligne de commande pour gérer les ressources Azure

PowerShell
Langage de script et shell pour l'automatisation des tâches

REST API
Interface programmatique pour interagir avec Azure DevOps

Extensions et plugins
Modules complémentaires pour étendre les fonctionnalités

Pourquoi automatiser ?
- Réduction des erreurs humaines
- Gain de temps et d'efficacité
- Standardisation des processus
- Facilité de maintenance et de mise à l'échelle



Introduction à Azure CLI

Qu'est-ce qu'Azure CLI ?

Interface en ligne de commande multiplateforme pour gérer les ressources Azure et Azure DevOps.

- Compatible avec Windows, macOS et Linux
- Intégration avec scripts et automatisation
- Sortie formatée (JSON, table, TSV)
- Mode interactif et autocomplétion

Installation

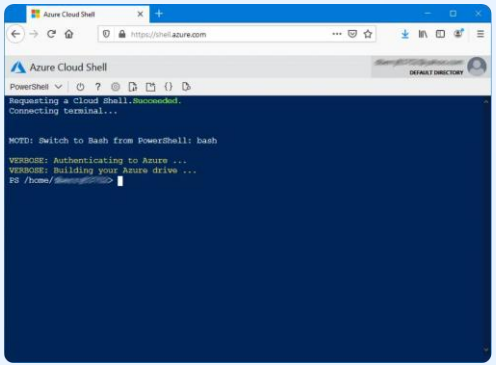
```
# Windows (PowerShell)
 Invoke-WebRequest -Uri https://aka.ms/installazurecliwindows -
OutFile .\AzureCLI.msi

# Linux
 curl -sL https://aka.ms/InstallAzureCLIDeb | sudo bash
```

Structure des commandes

```
az [groupe] [sous-groupe] [commande] [paramètres]
```

Exemples de groupes

az login - Connexion à Azure
az account - Gestion des abonnements
az group - Gestion des groupes de ressources
az devops - Commandes Azure DevOps
az pipelines - Gestion des pipelines



Commandes Azure CLI pour Azure DevOps

Syntaxe de base pour Azure DevOps

```
az devops [sous-groupe] [commande] --org https://dev.azure.com/organisation --project "Nom du projet"
```

Commandes principales

Groupe	Description
az devops	Commandes générales Azure DevOps
az repos	Gestion des dépôts Git
az pipelines	Gestion des pipelines CI/CD
az boards	Gestion des tableaux et éléments de travail
az artifacts	Gestion des artefacts et packages

Exemples d'utilisation

Gestion des dépôts

```
# Lister les dépôts
az repos list --org https://dev.azure.com/monorg

# Créer un dépôt
az repos create --name "mon-repo" --project "MonProjet"
```

Gestion des pipelines

```
# Lister les pipelines
az pipelines list --org https://dev.azure.com/monorg

# Exécuter un pipeline
az pipelines run --id 123 --branch main
```

Exemples pratiques Azure CLI

Gestion des projets

```
# Créer un nouveau projet
az devops project create --name "MonProjet"
--description "Description du projet"
--org https://dev.azure.com/organisation
```

Gestion des dépôts

```
# Lister les dépôts
az repos list --project "MonProjet"
--org https://dev.azure.com/organisation

# Créer un nouveau dépôt
az repos create --name "mon-repo"
--project "MonProjet"
```

Gestion des pipelines

```
# Lister les pipelines
az pipelines list --project "MonProjet"
--org https://dev.azure.com/organisation

# Exécuter un pipeline
az pipelines run --id 123
--project "MonProjet"
```

Gestion des éléments de travail

```
# Créer un élément de travail
az boards work-item create --title "Nouvelle tâche"
--type "Task" --project "MonProjet"
--org https://dev.azure.com/organisation

# Mettre à jour un élément de travail
az boards work-item update --id 456
--state "Active"
```

Introduction à PowerShell pour Azure DevOps

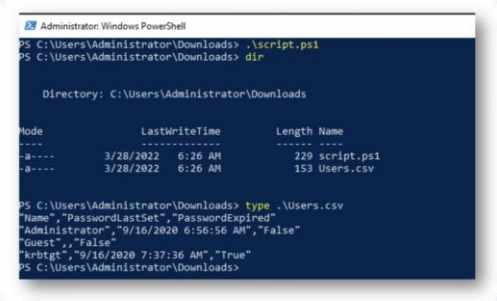
Qu'est-ce que PowerShell ?

Langage de script et shell conçu pour l'automatisation des tâches et la gestion de configuration.

Caractéristiques principales

- Orienté objet et basé sur .NET Framework
- Puissant traitement de texte et manipulation de données
- Intégration native avec Azure et Azure DevOps
- Multiplateforme (Windows, Linux, macOS)

```
# Exemple de base PowerShell
$connection = Connect-AzAccount
Get-AzResourceGroup | Where-Object { $_.Location -eq "WestEurope" }
```



Avantages pour Azure DevOps

Automatisation avancée
Scripts complexes avec logique conditionnelle et traitement d'erreurs

Intégration complète
Accès à toutes les fonctionnalités d'Azure DevOps via les modules dédiés

Gestion des pipelines
Création et modification programmatique des pipelines CI/CD

Sécurité renforcée
Gestion des secrets et authentification sécurisée

Modules PowerShell pour Azure

Modules principaux

Az

Module principal pour gérer les ressources Azure. Remplace AzureRM.

```
# Installation
Install-Module -Name Az -AllowClobber -Scope CurrentUser
```

Az.DevOps

Module dédié à l'automatisation des tâches Azure DevOps.

```
# Installation
Install-Module -Name Az.DevOps -Scope CurrentUser
```

Az.Accounts

Gestion des comptes, authentification et contexte Azure.

```
# Connexion à Azure
Connect-AzAccount
```

Utilisation des modules

Avantages des modules PowerShell

- Intégration native avec l'écosystème Microsoft
- Gestion avancée des objets et du pipeline
- Fonctionnalités de scripting puissantes
- Support complet de l'API Azure
- Mise à jour régulière et documentation complète

Exemple de script avec modules

```
# Importer les modules
Import-Module Az.DevOps
Import-Module Az.Resources

# Définir l'organisation et le projet
$organization = "https://dev.azure.com/monorg"
$project = "MonProjet"

# Obtenir les pipelines
Get-AzDevOpsPipeline -Organization $organization -Project $project
```

Automatisation des tâches Azure DevOps avec PowerShell

Gestion des ressources Azure

Création de ressources

```
# Créer un groupe de ressources
New-AzResourceGroup -Name "MonGroupe" -Location "WestEurope"
```

Gestion des branches

```
# Créer une politique de branche
$policy = @{
MinimumApproverCount = 2
}
New-AzDevOpsBranchPolicy -RepositoryId $repoId -BranchName "main"
```

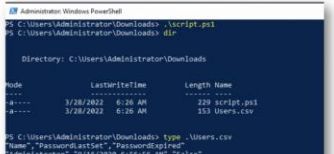
Intégration avec les pipelines

Gestion des variables

```
# Définir des variables de pipeline
$variables = @{
"BuildConfiguration" = "Release"
}
Set-AzDevOpsPipelineVariable -PipelineId 123 -Variables $variables
```

Automatisation des déploiements

```
# Fonction pour déployer vers plusieurs environnements
function Deploy-ToEnvironment {
param ($environment, $buildId)
Start-AzDevOpsRelease -BuildId $buildId -Environment $environment
}
```



API REST Azure DevOps

Présentation des API REST

Les API REST d'Azure DevOps permettent une intégration personnalisée et avancée avec d'autres systèmes et applications.

Avantages des API REST

- Intégration avec n'importe quel langage ou plateforme
- Accès complet à toutes les fonctionnalités d'Azure DevOps
- Automatisation avancée et workflows personnalisés
- Extensibilité et intégration avec des systèmes tiers

URL de base :

https://dev.azure.com/{organization}/_apis/{area}/{resource}?api-version=6.0

Créer un élément de travail

```
// Requête PATCH avec JSON
 PATCH
https://dev.azure.com/{organization}/{project}/_apis/wit/workitems/$Task?
api-version=6.0
 [
 {
"op"  "add"
"path"  "/fields/System.Title"
"value"  "Nouvelle tâche via API"
 }
```

Exemples d'utilisation

Récupérer les projets

```
// Requête GET
 GET https://dev.azure.com/{organization}/_apis/projects?api-version=6.0
```

Créer un dépôt Git

```
// Requête POST avec JSON
 POST https://dev.azure.com/{organization}/{project}/_apis/git/repositories?
api-version=6.0

 {
"name": "MonNouveauRepo" ,
"project": {
"id": "00000000-0000-0000-0000-000000000000"
 }
 }
```

Résumé et bonnes pratiques

Points clés à retenir

Choisir le bon outil

Azure CLI pour les commandes simples, PowerShell pour les scripts complexes, API REST pour l'intégration personnalisée.

Scripts modulaires et réutilisables

Créer des fonctions et modules réutilisables pour maximiser l'efficacité et la maintenabilité.

Sécurité avant tout

Gérer correctement les secrets, appliquer le principe du moindre privilège et journaliser les actions.

Automatisation de bout en bout

Intégrer les scripts dans les pipelines CI/CD pour une automatisation complète du cycle de vie.

Ressources supplémentaires

- Documentation Azure DevOps
- Référence Azure CLI
- Documentation PowerShell
- Référence API REST Azure DevOps
- Microsoft Learn - Parcours d'apprentissage

Questions fréquentes

Quand utiliser Azure CLI vs PowerShell ?

Azure CLI pour les commandes simples et rapides, PowerShell pour les scripts complexes avec logique avancée.

Comment gérer les secrets en toute sécurité ?

Utiliser Azure Key Vault ou les variables secrètes des pipelines Azure DevOps.

> L'automatisation n'est pas seulement un gain de temps, c'est une transformation de la façon dont nous travaillons.