

TD : Gestion automatisée des comptes locaux

Objectif : Concevoir un outil robuste capable d'industrialiser la création d'utilisateurs à partir d'un fichier source, tout en gérant les erreurs et les journaux (logs).

1. Préparation de l'environnement

- Créez un dossier de travail dédié.
 - Créez un fichier `utilisateurs.csv` avec trois colonnes : `Nom`, `Description`, `Groupe`.
 - Remplissez-le avec au moins **3 utilisateurs fictifs**.
-

2. Structure du Script

Créez un fichier nommé `GestionComptes.ps1`. Le script doit obligatoirement respecter la structure suivante :

- ****Bloc `param()`**** : Avec validation (utilisez `ValidateSet` pour définir les actions autorisées).
 - ****Bloc `begin {}`**** : Dédié à la déclaration des fonctions internes.
 - ****Bloc `process {}`**** : Dédié au corps de l'exécution (boucles, logique métier).
 - ****Bloc `end {}`**** : Pour la clôture et le rapport final.
-

3. Exercices à réaliser

Exercice A : Les Fonctions (Bloc `begin`)

Définissez deux fonctions à l'intérieur du bloc `begin` :

1. **`Test-UserExists`** : Prend un nom d'utilisateur en paramètre et retourne `$true` s'il existe sur la machine, sinon `$false`.
 - *Indice : Utilisez la commande `Get-LocalUser`.*
2. **`Write-Log`** : Prend un message et un chemin de fichier en paramètres. Elle doit ajouter une ligne au fichier de log incluant l'horodatage (`Get-Date`).

Exercice B : Importation et Vérification (Bloc `process`)

Implémentez l'action **"Lecture"** :

- Le script doit importer le fichier CSV.
- Pour chaque utilisateur du fichier, affichez dans la console s'il est présent ou absent du système en appelant votre fonction `Test-UserExists`.

Exercice C : Création Massive (Bloc `process`)

Implémentez l'action **"Creation"** :

- Pour chaque utilisateur absent du système :

1. Créer le compte local (sans mot de passe pour le test).
 2. L'ajouter au groupe local spécifié dans la colonne **Groupe**.
 3. Appeler **Write-Log** pour consigner la réussite de l'opération.
- **Gestion d'erreurs** : Encapsulez la création dans un bloc `try { ... } catch { ... }` pour capturer les incidents (ex: groupe inexistant).

Exercice D : Nettoyage (Action "Suppression")

Ajoutez une action pour supprimer les utilisateurs listés dans le CSV.

- **Sécurité** : Le script doit demander une confirmation explicite avant chaque suppression (utilisez le paramètre `-Confirm` ou la commande `Read-Host`).

4. Bonus pour les experts

- **Sécurité** : Modifiez l'action de création pour générer un mot de passe aléatoire ou solliciter une saisie sécurisée via `Read-Host -AsSecureString`.
- **Rapport final** : Dans le bloc `end {}`, affichez un résumé indiquant le nombre total d'utilisateurs traités durant la session.

[!IMPORTANT] **Consigne de test** : Pour manipuler les comptes et groupes locaux, vous devez impérativement exécuter votre terminal PowerShell en tant qu'**Administrateur**.
