

PowerShell – Get-ChildItem

Cmdlets permet de naviguer et manipuler le système de fichiers. Les cmdlets présentées ici fonctionnent de manière cohérente et prévisible, que l'on travaille dans le dossier courant ou sur un chemin précis. Elles constituent le socle de toute automatisation système.

6TQ

5TQ

 niveau

Lister le contenu d'un dossier : `Get-ChildItem`

```
1 | Get-ChildItem
```

Liste le contenu du **dossier courant** : fichiers et sous-dossiers. C'est l'équivalent moderne de `dir` ou `ls`.

Tu peux également indiquer **explicitement un chemin** :

```
1 | Get-ChildItem C:\Windows
2 | Get-ChildItem C:\Users\Public
```

Ou filtrer le contenu :

```
1 | Get-ChildItem *.txt
2 | Get-ChildItem C:\Logs\*.log
```

Dans tous les cas, `Get-ChildItem` renvoie des **objets** (`FileInfo` et `DirectoryInfo`) exploitables dans un pipeline.

Voici une **liste claire, structurée et pédagogique** des **principales propriétés et méthodes** des objets retournés par `Get-ChildItem`. Elle est adaptée à un cours pour techniciens débutants.

Objets retournés par `Get-ChildItem`

La cmdlet `Get-ChildItem` renvoie principalement deux types d'objets :

- **FileInfo** → pour les fichiers
- **DirectoryInfo** → pour les dossiers

Ces deux objets partagent plusieurs propriétés communes.

Propriétés communes (fichiers et dossiers)

Propriété	Description
<code>Name</code>	Nom de l'élément
<code>FullName</code>	Chemin complet
<code>Extension</code>	Extension du fichier (vide pour les dossiers)
<code>Exists</code>	Indique si l'élément existe
<code>CreationTime</code>	Date de création
<code>LastWriteTime</code>	Date de dernière modification
<code>LastAccessTime</code>	Date du dernier accès
<code>Attributes</code>	Attributs (Hidden, ReadOnly, Directory...)
<code>PSIsContainer</code>	Indique si l'objet est un dossier (<code>True</code>)

Propriétés spécifiques aux fichiers (`FileInfo`)

Propriété	Description
<code>Length</code>	Taille du fichier en octets
<code>IsReadOnly</code>	Fichier en lecture seule
<code>DirectoryName</code>	Dossier parent
<code>BaseName</code>	Nom du fichier sans extension

Propriétés spécifiques aux dossiers (`DirectoryInfo`)

Propriété	Description
Parent	Dossier parent
Root	Racine du lecteur
Exists	Indique si le dossier existe

Méthodes principales (fichiers et dossiers)

Les méthodes sont des **actions** que l'on peut appliquer à l'objet.

Méthodes communes

Méthode	Rôle
Delete()	Supprimer l'élément
MoveTo()	Déplacer l'élément
Refresh()	Mettre à jour les informations

Voici l'extension **claire et pédagogique** à ajouter pour `Get-ChildItem -Directory`, parfaitement cohérente avec le chapitre précédent.

Le paramètre `-Directory`

Par défaut, `Get-ChildItem` affiche **les fichiers et les dossiers**.

Si tu veux limiter l'affichage **uniquement aux dossiers**, tu peux utiliser le paramètre `-Directory` :

```
1 | Get-ChildItem -Directory
```

Cette commande :

- liste uniquement les **dossiers**
- renvoie des objets de type **DirectoryInfo**
- évite tout filtrage manuel supplémentaire

Exemple avec un chemin spécifique

```
1 | Get-ChildItem C:\Users -Directory
```

Affiche uniquement les dossiers présents dans `C:\Users`.

Comparaison avec la méthode objet

Sans `-Directory`, on pourrait écrire :

```
1 | Get-ChildItem | Where-Object { $_.PSIsContainer }
```

Mais `-Directory` est :

- plus lisible
- plus rapide
- plus explicite

👉 Toujours préférer `-Directory` quand le besoin est clair.

Complément utile : `-File`

PowerShell propose également :

```
1 | Get-ChildItem -File
```

Cette commande :

- liste uniquement les fichiers
- renvoie des objets **FileInfo**

Résumé rapide

Commande	Résultat
<code>Get-ChildItem</code>	Fichiers + dossiers
<code>Get-ChildItem -Directory</code>	Dossiers uniquement
<code>Get-ChildItem -File</code>	Fichiers uniquement

À retenir

- `Get-ChildItem` fonctionne **toujours selon le même principe** : dossier courant **ou** chemin explicitement fourni
- Les fichiers et dossiers sont manipulés comme des **objets**
- Les chemins relatifs dépendent du dossier courant
- `-Directory` est un **paramètre natif**, pas un filtre
- Il travaille directement sur le type d'objet
- Il rend le code plus simple et plus professionnel