

# PowerShell Scripting Guide

Copy sites and lists, run link reports, inventories and other tasks from PowerShell and the command line

Module `Cognillo.SPE` version 1.0

This guide shows how to run SharePoint Essentials Toolkit from PowerShell instead of clicking through the app, so you can repeat a task, run it for many sites at once from a spreadsheet, or run it on a schedule. Most of the guide covers copying sites, lists and libraries, which has its own set of commands. Section 10 shows how to script everything else: link reports, inventories, permission and storage reports, and any other task you can save in the app.

It is written for two kinds of readers:

- **EVERYONE** Sections 1–11 and 16: install the module, preview a copy, run it, copy from a CSV file, run reports, read the results, fix common problems. If you can open PowerShell and paste a command, you can follow these.
- **ADVANCED** Sections 12–15: automation, scheduling, the job spec file format, the executable's command line and the result file. For administrators and developers building pipelines.

- |                                |                                      |
|--------------------------------|--------------------------------------|
| 1. How it works                | 9. Templates and user mappings       |
| 2. Before you start            | 10. Link reports and other tasks     |
| 3. Install the module          | 11. Results, errors and exit codes   |
| 4. Quick start                 | 12. Automation and scheduling        |
| 5. Copying lists and libraries | 13. Job spec file reference          |
| 6. Copying sites               | 14. Command line reference (SPE.exe) |
| 7. Bulk copies from a CSV file | 15. PowerShell command reference     |
| 8. Copy settings               | 16. Troubleshooting and FAQ          |

## 1. How it works EVERYONE

You describe the copy with plain site URLs and list names. The toolkit signs in with the connection you already use in the app, looks the sites and lists up in SharePoint, checks your permissions, then runs the same copy engine as the app.

### PowerShell commands

`Copy-SPEList` , `Copy-SPESite` and friends. Best for interactive use and most scripts. They return a result object you can inspect, filter or export.

### Job spec file + command line

A JSON file that describes the job, run with `SharePoint Essentials Toolkit.exe --run-spec` . Best for scheduled tasks, CI/CD and other tools. The PowerShell commands write this file for you.

Every copy you run from a script:

- appears on the app's **History** page with its full report, exactly like a job started from the app;
- can be previewed first with `-WhatIf` , which reads both sites and shows what would be copied without copying anything;
- follows the same license rules as jobs you run in the app.

Copies are the most flexible to script because you describe them entirely in the command. Every other task (link reports, inventories, permission reports and so on) is set up once in the app, saved as a template, and then run from a script, optionally against other sites. See section 10.

## 2. Before you start EVERYONE

| You need                          | Details                                                                                                                                                                                       |
|-----------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| SharePoint Essentials Toolkit     | Installed and activated on the machine that runs the scripts.                                                                                                                                 |
| A saved connection to your tenant | Open the app once and sign in to your SharePoint tenant with the account that will run the copies. Scripts reuse that saved connection, so run them as the <b>same Windows user</b> .         |
| Permissions                       | Read access on the source; edit rights on the target. Site owner (or site collection administrator) on the target is recommended, and required to copy site permissions, groups and features. |
| Target sites                      | The target site must already exist. Copy Sites copies <i>into</i> an existing site; it does not create the site collection.                                                                   |
| PowerShell                        | Windows PowerShell 5.1 (built into Windows) or PowerShell 7+.                                                                                                                                 |

### 3. Install the module EVERYONE

The module is a folder named `Cognillo.SPE` containing `Cognillo.SPE.psd1` and `Cognillo.SPE.psm1`.

#### Option A – install for your user (recommended)

Copy the `Cognillo.SPE` folder into your personal modules folder. PowerShell then loads it automatically.

```
# Windows PowerShell 5.1
$modules = Join-Path ([Environment]::GetFolderPath('MyDocuments')) 'WindowsPowerShell\Modules'
# PowerShell 7: use 'PowerShell\Modules' instead of 'WindowsPowerShell\Modules'

New-Item -ItemType Directory -Force $modules | Out-Null
Copy-Item -Recurse -Force '.\Cognillo.SPE' $modules
Get-ChildItem (Join-Path $modules 'Cognillo.SPE') | Unblock-File
```

#### Option B – load it from any folder

```
Import-Module 'C:\Tools\Cognillo.SPE'
```

#### Check it works

```
Get-Command -Module Cognillo.SPE
Get-SPEExecutable          # shows where SharePoint Essentials Toolkit.exe was found
```

##### The executable was not found?

Point the module at your install folder once per session, or set it for every session with an environment variable:

```
Set-SPEExecutable -Path 'C:\Program Files\Cognillo\SharePoint Essentials Toolkit'
[Environment]::SetEnvironmentVariable('SPE_EXE', 'C:\...\SharePoint Essentials Toolkit.exe', 'User')
```

##### "Running scripts is disabled on this system"

Allow local scripts for your user: `Set-ExecutionPolicy -Scope CurrentUser RemoteSigned`, then run `Unblock-File` on the module files as shown above.

## 4. Quick start EVERYONE

- 1 **Preview the copy.** `-WhatIf` signs in, reads both sites and lists exactly what would be copied. Nothing is changed.

```
Copy-SPEList -SourceSite 'https://contoso.sharepoint.com/sites/HR' `
              -DestinationSite 'https://contoso.sharepoint.com/sites/HR-Archive' `
              -List 'Documents', 'Policies' -WhatIf
```

```
https://contoso.sharepoint.com/sites/HR -> https://contoso.sharepoint.com/sites/HR-Archive
Documents (exists on target)
Policies
[HEADLESS] What-if: nothing was copied.

Status          : Planned
JobName          : Copy Lists (script) 2026-10-15 11:40
...
```

- 2 **Run it.** Remove `-WhatIf`. Progress is shown in the console; press Ctrl+C to cancel safely.

```
$result = Copy-SPEList -SourceSite 'https://contoso.sharepoint.com/sites/HR' `
                      -DestinationSite 'https://contoso.sharepoint.com/sites/HR-Archive' `
                      -List 'Documents', 'Policies'
```

- 3 **Check the result.**

```
$result | Format-List *
$result.Status          # Completed, CompletedWithWarnings, Error, Cancelled ...
$result.ItemsCompleted
$result.Messages        # errors and warnings found while preparing the job
```

The full report (every item copied, skipped or failed) is on the app's **History** page.

## 5. Copying lists and libraries EVERYONE

`Copy-SPEList` copies lists and document libraries from one site to another. Lists are named by their **title** ( `Documents` ) or their **URL name** ( `Shared Documents` ). Names are not case-sensitive.

### One or more named lists

```
Copy-SPEList -SourceSite $src -DestinationSite $dst -List 'Documents'
Copy-SPEList -SourceSite $src -DestinationSite $dst -List 'Documents', 'Policies', 'Team Calendar'
```

### Wildcards

\* matches any text and ? matches one character.

```
Copy-SPEList -SourceSite $src -DestinationSite $dst -List 'Project*'      # Project A, Projects 2025 ...
Copy-SPEList -SourceSite $src -DestinationSite $dst -List 'Report 202?'
```

### Every list on the site

Leave out `-List` to copy every list and library a user would normally copy. SharePoint's own system lists and galleries (Style Library, Master Page Gallery, hidden lists, etc.) are left out. Wildcards follow the same rule; name a system list explicitly if you really want it.

```
Copy-SPEList -SourceSite $src -DestinationSite $dst
Copy-SPEList -SourceSite $src -DestinationSite $dst -ExcludeList 'Archive*', 'Old Documents'
```

### Copy under a new name

When exactly one list is copied, you can give it a new title and URL on the target.

```
Copy-SPEList -SourceSite $src -DestinationSite $dst -List 'Projects' `
-NewListTitle 'Projects (2025 Archive)' -NewListUrlName 'Projects2025'
```

### Copy into a folder of the target list

```
Copy-SPEList -SourceSite $src -DestinationSite $dst -List 'Documents' -TargetFolder '/Imported/HR'
```

### Lists that may not exist

By default a named list that is not found stops the job before anything is copied. With `-IgnoreMissingLists` it is reported as a warning and the other lists are copied. Unmatched wildcards are always just warnings.

```
Copy-SPEList -SourceSite $src -DestinationSite $dst -List 'Documents', 'Contracts' -IgnoreMissingLists
```

## Same lists to several targets

```
'https://contoso.sharepoint.com/sites/Sales-EU', 'https://contoso.sharepoint.com/sites/Sales-US' |  
ForEach-Object { [pscustomobject]{ SourceSite = $src; DestinationSite = $_ } } |  
Copy-SPEList -List 'Price Lists'
```

## 6. Copying sites EVERYONE

`Copy-SPESite` copies a site's structure (features, site columns, content types, groups, permissions, navigation, settings, lists and libraries) and its content into an existing target site.

```
# Full copy: structure and content
Copy-SPESite -SourceSite 'https://contoso.sharepoint.com/sites/HR' `
             -DestinationSite 'https://contoso.sharepoint.com/sites/HR-New'

# Structure only: lists and libraries are created empty (staging, new environments)
Copy-SPESite -SourceSite $src -DestinationSite $dst -StructureOnly

# Only some of the lists
Copy-SPESite -SourceSite $src -DestinationSite $dst -ExcludeList 'Archive*'
```

### Copy-SPESite asks for confirmation

Because a site copy changes a lot on the target, PowerShell asks "*Are you sure you want to perform this action?*" before it runs. In unattended scripts add `-Confirm:$false`.

## 7. Bulk copies from a CSV file EVERYONE

Keep the list of copies in a spreadsheet, save it as CSV and pipe it into the command. **Everything piped into one command runs as one job**, with one entry in History and one report.

### Lists CSV

```
SourceSite,DestinationSite,List,NewListTitle,TargetFolder
https://contoso.sharepoint.com/sites/HR,https://contoso.sharepoint.com/sites/HR2,Documents,,
https://contoso.sharepoint.com/sites/HR,https://contoso.sharepoint.com/sites/HR2,Policies;Forms,,
https://contoso.sharepoint.com/sites/Finance,https://contoso.sharepoint.com/sites/Fin2,Invoices,Invoices 2025,
https://contoso.sharepoint.com/sites/Legal,https://contoso.sharepoint.com/sites/Legal2,,
```

```
Import-Csv .\lists.csv | Copy-SPEList -WhatIf      # check first
Import-Csv .\lists.csv | Copy-SPEList
```

### Sites CSV

```
SourceSite,DestinationSite
https://contoso.sharepoint.com/sites/HR,https://contoso.sharepoint.com/sites/HR-New
https://contoso.sharepoint.com/sites/IT,https://contoso.sharepoint.com/sites/IT-New
```

```
Import-Csv .\sites.csv | Copy-SPESite -JobName 'Wave 1 migration' -Confirm:$false
```

### Recognized columns

| Column          | Also accepted                                              | Meaning                                                   |
|-----------------|------------------------------------------------------------|-----------------------------------------------------------|
| SourceSite      | Source, SourceUrl, SourceSiteUrl                           | Source site URL (required)                                |
| DestinationSite | Target, TargetSite, TargetUrl, Destination, DestinationUrl | Target site URL (required)                                |
| List            | Lists, ListName, Name                                      | List names; separate several with ; . Empty = every list. |
| ExcludeList     | Exclude                                                    | Lists to leave out; separate with ;                       |
| NewListTitle    | TargetListTitle                                            | Lists only: new title (one list per row)                  |
| NewListUrlName  | TargetListUrlSegment                                       | Lists only: new URL name (one list per row)               |
| TargetFolder    |                                                            | Lists only: folder in the target list                     |

#### One list per row is fine

Rows with the same source and target (and no new name or folder) are combined automatically, so the first two rows of the lists CSV above become one copy of *Documents, Policies, Forms*.

**One list selection per source site, per job**

Within one job, a source site can be copied to several targets, but always with the same lists. If the same source needs different lists for different targets, or a renamed list plus other lists, run them as separate commands.

## 8. Copy settings EVERYONE

Create settings with `New-SPECopySettings` and pass them with `-CopySettings`. Only the settings you specify are changed; everything else keeps the template's value (see section 9) or the app's default.

```
$settings = New-SPECopySettings -OnItemExists Update -IncludeVersionHistory $true -VersionLimit 10 `
    -PreserveAuthor $true -IncludeItemPermissions $false

Copy-SPEList -SourceSite $src -DestinationSite $dst -List 'Documents' -CopySettings $settings
```

| Setting                               | Values                  | What it does                                                                                                                                           |
|---------------------------------------|-------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>When the target already has it</b> |                         |                                                                                                                                                        |
| <code>OnItemExists</code>             | Overwrite, Update, Skip | Item or file already on the target.                                                                                                                    |
| <code>OnListExists</code>             | Overwrite, Update, Skip | List already on the target.                                                                                                                            |
| <b>Copy method</b>                    |                         |                                                                                                                                                        |
| <code>MigrationApi</code>             | NativeAPI, MigrationAPI | <i>NativeAPI</i> : standard item-by-item copy. <i>MigrationAPI</i> : high-fidelity bulk copy through the SharePoint Migration API (SharePoint Online). |
| <code>CopySiteMode</code>             | FullCopy, StructureOnly | Sites only. Same as <code>-StructureOnly</code> .                                                                                                      |
| <b>Content and versions</b>           |                         |                                                                                                                                                        |
| <code>IncludeContent</code>           | \$true / \$false        | Copy items and files (off = structure only).                                                                                                           |
| <code>IncludeVersionHistory</code>    | \$true / \$false        | Copy earlier versions.                                                                                                                                 |
| <code>VersionLimit</code>             | 0 – 50000               | Maximum versions per item; 0 copies all.                                                                                                               |
| <code>MajorVersionsOnly</code>        | \$true / \$false        | Skip minor (draft) versions.                                                                                                                           |
| <b>Items and structure</b>            |                         |                                                                                                                                                        |
| <code>PreserveAuthor</code>           | \$true / \$false        | Keep Created By / Modified By and dates.                                                                                                               |
| <code>RetainItemIds</code>            | \$true / \$false        | Keep item IDs the same on the target.                                                                                                                  |
| <code>IncludeFolders</code>           | \$true / \$false        | Copy folders.                                                                                                                                          |
| <code>IncludeListViews</code>         | \$true / \$false        | Create the list views on the target.                                                                                                                   |
| <code>AutoActivateFeatures</code>     | \$true / \$false        | Lists: activate on the target the features the lists need.                                                                                             |
| <b>Permissions and labels</b>         |                         |                                                                                                                                                        |
| <code>IncludeSitePermissions</code>   | \$true / \$false        | Copy site permissions.                                                                                                                                 |
| <code>IncludeListPermissions</code>   | \$true / \$false        | Copy unique list permissions.                                                                                                                          |
| <code>IncludeItemPermissions</code>   | \$true / \$false        | Copy unique item and file permissions.                                                                                                                 |
| <code>IncludeRetentionLabels</code>   | \$true / \$false        | Apply the source retention labels.                                                                                                                     |

|                                                                                                                                                                                                                                                                             |                                            |                                                                                                          |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------|----------------------------------------------------------------------------------------------------------|
| <code>IncludeSensitivityLabels</code>                                                                                                                                                                                                                                       | <code>\$true</code> / <code>\$false</code> | Apply the source sensitivity labels.                                                                     |
| <b>Sites only</b>                                                                                                                                                                                                                                                           |                                            |                                                                                                          |
| <code>IncludeSiteFeatures</code> ,<br><code>IncludeSiteColumns</code> ,<br><code>IncludeSiteContentTypes</code> ,<br><code>IncludeSiteGroups</code> ,<br><code>IncludeSiteGroupMembers</code> ,<br><code>IncludeSiteNavigation</code> ,<br><code>IncludeSiteSettings</code> | <code>\$true</code> / <code>\$false</code> | Choose which parts of the site structure are copied.                                                     |
| <b>Other</b>                                                                                                                                                                                                                                                                |                                            |                                                                                                          |
| <code>FallbackUserAccount</code>                                                                                                                                                                                                                                            | e-mail or <code>''</code>                  | Account used when a source user does not exist on the target.<br><code>''</code> turns the fallback off. |

## 9. Templates and user mappings EVERYONE

### Start from a job template

Set up a Copy Sites or Copy Lists job in the app the way you like it, save it as a **job template**, then reuse all of its options from a script. The template provides the options; your command provides the sites and lists.

```
Copy-SPEList -Template 'HR nightly copy' -SourceSite $src -DestinationSite $dst -List 'Documents'

# Template options, with one change
Copy-SPEList -Template 'HR nightly copy' -SourceSite $src -DestinationSite $dst `
    -CopySettings (New-SPECopySettings -OnItemExists Skip)
```

The template name is the name shown on the app's Job Templates page (the file name without `.json`). By default templates are stored under `Documents\SharePoint Essentials Toolkit\Job Templates`.

### Map users between tenants or accounts

When people have different accounts on the target (tenant-to-tenant moves, renamed accounts), map them so authors, editors, people fields and permissions point to the right person.

```
users.csv
Source,Target
anna@contoso.com,anna.berg@fabrikam.com
jsmith@contoso.com,john.smith@fabrikam.com
```

```
$users = Import-SPEUserMapping .\users.csv
Copy-SPEList -SourceSite $src -DestinationSite $dst -UserMapping $users -Confirm:$false

# A few mappings inline
Copy-SPEList -SourceSite $src -DestinationSite $dst -UserMapping @{ 'old@contoso.com' = 'new@fabrikam.com' }
```

`Import-SPEUserMapping` also accepts the column pairs `SourceUser/TargetUser`, `SourceValue/DestinationValue` and `SourceEmail/TargetEmail`. Mappings are added to the template's; a mapping for the same source user replaces the template's entry.

## 10. Link reports and other tasks EVERYONE

Any task you can run in the app can also run from a script, not only copies. You set the task up once in the app, choose the options and the report you want, and save it as a **job template**. A script then runs that template, as saved or against other sites.

| Area                    | Examples of tasks you can script                                                                                |
|-------------------------|-----------------------------------------------------------------------------------------------------------------|
| Links                   | Links report: the links in pages, documents and list items, including broken links                              |
| Inventory               | Site & List Inventory, Item Inventory, Field Inventory, Content Type Inventory, User Inventory, Alert Inventory |
| Permissions and sharing | Permissions, Check Account Permissions, Sharing Links, Open To Everyone                                         |
| Storage and usage       | Storage Report, Site Analytics, Activity, User Activity                                                         |
| Search                  | Content Search                                                                                                  |
| Copy                    | Copy Sites and Copy Lists (sections 5–9), and the other copy tasks through templates                            |

Discovery, and the change logs written when you edit permissions or items from the actions panel, can't be run from a script.

### Step 1 – save the task as a template

1. In the app, set up the task as usual. For example, a **Links** report on one site that looks for broken links.
2. Choose how you want the report delivered, if at all: by email or to a SharePoint library. Scripted runs deliver the report the same way.
3. Save it as a job template, for example *Broken links*. On the **Job Templates** page you can copy a ready-made PowerShell command for it.

### Step 2 – run it from PowerShell

```
# Run the template exactly as saved
Invoke-SPEJobTemplate -Name 'Broken links'

# Run it against a different site. The template's options are used; its sites are replaced by this one.
Invoke-SPEJobTemplate -Name 'Broken links' -Url 'https://contoso.sharepoint.com/sites/Intranet'
```

| Name                                                         | Status  | ExitCode |
|--------------------------------------------------------------|---------|----------|
| ----                                                         | -----   | -----    |
| Broken links (https://contoso.sharepoint.com/sites/Intranet) | Success | 0        |

The report itself (every link found, its status and where it is) is on the app's **History** page, and is delivered by email or to SharePoint if the template is set up to do so.

### Example: link report for many sites from a CSV file

Each site is run separately, so every site gets its own report.

```
sites.csv
SiteUrl
https://contoso.sharepoint.com/sites/Intranet
https://contoso.sharepoint.com/sites/HR
https://contoso.sharepoint.com/sites/Finance
```

```
$results = Import-Csv .\sites.csv | Invoke-SPEJobTemplate -Name 'Broken links'
$results | Where-Object ExitCode -ne 0      # the sites that need a look
```

### Example: weekly governance reports in one script

```
# weekly-reports.ps1: inventory, permissions and storage, one after the other
Import-Module Cognillo.SPE

$templates = 'Site and list inventory', 'Permissions - all sites', 'Storage report', 'Sharing links'
$results = foreach ($name in $templates) { Invoke-SPEJobTemplate -Name $name }

$results | Format-Table Name, Status, DurationSeconds
if ($results.ExitCode | Where-Object { $_ -notin 0, 2 }) { exit 1 }
```

### Example: run a task again from History

Every task in History has an ID. Select the task on the History page and choose **PowerShell > Copy command** or **Create script file...**, or run it with the module:

```
Invoke-SPEJob -Id '6f1b8e0a-3c1d-4c35-9a41-2f8f0c7d9b11'
```

### Example: collect user activity

If User Activity is set up in the app, collect the latest activity from the Microsoft 365 audit log, for example before running an Activity report:

```
Sync-SPEUserActivity
Invoke-SPEJobTemplate -Name 'Inactive sites'
```

### Copy templates with other sites

For Copy Sites and Copy Lists templates, `-Url` is the new **source** site and `-TargetUrl` the target site. Without `-TargetUrl` the template's own target is used. The template's lists are copied where the new source has them.

```
Invoke-SPEJobTemplate -Name 'HR nightly copy' -Url $newSource -TargetUrl $newTarget
```

#### Reports or copies: which commands?

For copies, prefer `Copy-SPEList` and `Copy-SPESite`. They work without a template, can preview with `-WhatIf`, and return detailed counts. For all other tasks use `Invoke-SPEJobTemplate` and `Invoke-SPEJob`. These return the task's status and exit code, and the details are in History.

## 11. Results, errors and exit codes EVERYONE ADVANCED

### The result object

Every command returns one result per job. The default view shows the main counts; `Format-List *` shows everything.

| Property                                                    | Meaning                                                                                                                                                                           |
|-------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Status                                                      | <i>Completed, CompletedWithWarnings, CompletedWithErrors, Error</i> (could not start), <i>Cancelled, Planned for -WhatIf</i> ; <i>Invalid</i> when the job could not be prepared. |
| ExitCode                                                    | See the table below.                                                                                                                                                              |
| JobId , JobName , JobType                                   | Identify the job in History.                                                                                                                                                      |
| Started , Completed , DurationSeconds                       | Timing.                                                                                                                                                                           |
| ListsTotal , ListsCompleted , ListsWithError                | List counts.                                                                                                                                                                      |
| ItemsTotal , ItemsCompleted , ItemsSkipped , ItemsWithError | Item and file counts.                                                                                                                                                             |
| Errors , Warnings                                           | Number of errors and warnings in the job report.                                                                                                                                  |
| Plan                                                        | One entry per source list: source and target site, target list title and URL, whether the target list already exists, and the permission check result.                            |
| Messages                                                    | Problems found while preparing the job, prefixed <code>Error:</code> or <code>Warning:</code> .                                                                                   |
| ErrorMessage                                                | Why the job failed or could not start.                                                                                                                                            |
| JobHistoryFolder                                            | Folder holding the job's report data and logs.                                                                                                                                    |

```
# Save the plan of a preview to Excel-friendly CSV
(Import-Csv .\lists.csv | Copy-SPEList -WhatIf).Plan | Export-Csv .\plan.csv -NoTypeInformation

# Lists that already exist on the target
$r = Copy-SPEList -SourceSite $src -DestinationSite $dst -WhatIf
$r.Plan | Where-Object TargetListExists | Select-Object SourceList, TargetSite
```

### Errors and warnings in PowerShell

- Warnings (for example an unmatched wildcard, or a job that completed with warnings) are written with `Write-Warning` .
- A failed or invalid job writes a PowerShell error **and still returns the result object**. Use `-ErrorAction Stop` to stop the script instead, or `-ErrorVariable` to collect the error.
- `$LASTEXITCODE` is set to the job's exit code after each run.

### Results of reports and other tasks

`Invoke-SPEJobTemplate` , `Invoke-SPEJob` and `Sync-SPEUserActivity` return a smaller result: `Name` , `Status` , `ExitCode` , `Started` , `Completed` and `DurationSeconds` . They report warnings and errors the same way. The full report is on the History page.

## Exit codes

| Code | Name                  | Meaning                                                                                                                                   |
|------|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------|
| 0    | Success               | Completed, or a <code>-WhatIf</code> preview that found no problems.                                                                      |
| 1    | Failed                | The job failed or completed with errors.                                                                                                  |
| 2    | CompletedWithWarnings | The job finished; the report has warnings.                                                                                                |
| 3    | Cancelled             | Cancelled with Ctrl+C.                                                                                                                    |
| 4    | InvalidInput          | Copy jobs: the job could not be prepared (bad URL, list not found, template missing, permission check failed). <b>Nothing was copied.</b> |

Reports and other tasks use codes 0–3; a template that can't be found returns 1. User activity sync returns 0 or 1. Jobs started by Windows Task Scheduler from the app's own scheduler only ever return 0 or 1, because Task Scheduler reports every other code as a failure.

## 12. Automation and scheduling ADVANCED

### Save a job once, run it many times

`-ExportSpec` writes the job to a file instead of running it. Keep the file in source control, review it, and run it whenever you need with `Invoke-SPECopyJob` or the command line.

```
Import-Csv .\wave1.csv | Copy-SPEList -Template 'Migration defaults' -JobName 'Wave 1' -ExportSpec .\wave1.json

Invoke-SPECopyJob -Path .\wave1.json -WhatIf
Invoke-SPECopyJob -Path .\wave1.json
```

### A robust unattended script

```
# nightly-copy.ps1
Import-Module Cognillo.SPE
$ErrorActionPreference = 'Stop'
$log = "C:\SPE\Logs\nightly-$(Get-Date -Format yyyyMMdd-HH:mm).json"

try {
    $result = Import-Csv 'C:\SPE\nightly.csv' |
        Copy-SPEList -Template 'Nightly' -JobName "Nightly $(Get-Date -Format d)" -ErrorAction Continue
}
catch {
    Write-Error "Copy could not start: $_"
    exit 4
}

$result | ConvertTo-Json -Depth 5 | Set-Content $log

switch ($LASTEXITCODE) {
    0 { 'Copy completed.' }
    2 { Write-Warning "Completed with $($result.Warnings) warning(s)." }
    default {
        # send an alert, open a ticket ...
        Write-Error "Copy failed ($($result.Status)): $($result.ErrorMessage)"
    }
}
exit $LASTEXITCODE
```

### Run it with Windows Task Scheduler

```
$action = New-ScheduledTaskAction -Execute 'powershell.exe' `
    -Argument '-NoProfile -ExecutionPolicy Bypass -File "C:\SPE\nightly-copy.ps1"'
$trigger = New-ScheduledTaskTrigger -Daily -At 1am
Register-ScheduledTask -TaskName 'SPE nightly copy' -Action $action -Trigger $trigger `
    -User "$env:USERDOMAIN\$env:USERNAME" -RunLevel Limited
```

The same approach schedules reports. For example, the *weekly-reports.ps1* script from section 10 every Monday morning:

```
$action = New-ScheduledTaskAction -Execute 'powershell.exe' `
        -Argument '-NoProfile -ExecutionPolicy Bypass -File "C:\SPE\weekly-reports.ps1"'
$trigger = New-ScheduledTaskTrigger -Weekly -DaysOfWeek Monday -At 6am
Register-ScheduledTask -TaskName 'SPE weekly reports' -Action $action -Trigger $trigger
```

### Only need one task on a schedule?

The app can schedule a single task itself (it creates the Windows scheduled task for you). Use a script when you want several tasks in a row, a list of sites from a file, or your own alerts and checks.

### Run as the user who signed in to the app

Scripted tasks use the SharePoint connection saved by the app for that Windows user. Register the task under the same account and choose "Run whether user is logged on or not" if needed.

### Copy jobs without PowerShell

Other tools (CI/CD pipelines, orchestration software, .NET or Python programs) can write a job spec file and call the executable directly, then read the result file. See sections 13 and 14.

## 13. Job spec file reference ADVANCED

A job spec is a JSON file. Property names are not case-sensitive; comments ( `//` ) and trailing commas are allowed; enum values are written as text.

### Complete example

```
{
  // Copy Lists job built on a template, two copies, a few option overrides
  "jobType": "CopyLists",           // CopyLists or CopySites; omit to use the template's type
  "jobName": "HR archive - October",
  "template": "HR nightly copy",    // optional
  "whatIf": false,                  // true = plan only (same as --whatif)

  "copies": [
    {
      "source": "https://contoso.sharepoint.com/sites/HR",
      "target": "https://contoso.sharepoint.com/sites/HR-Archive",
      "lists": ["Documents", "Polic*"],
      "excludeLists": ["Policies Draft"],
      "ignoreMissingLists": false
    },
    {
      "source": "https://contoso.sharepoint.com/sites/Finance",
      "target": "https://contoso.sharepoint.com/sites/Finance-Archive",
      "lists": ["Invoices"],
      "targetListTitle": "Invoices 2025",
      "targetListUrlSegment": "Invoices2025",
      "targetFolder": "/Imported"
    }
  ],

  "options": {
    "onItemExists": "Update",
    "onListExists": "Update",
    "migrationApi": "NativeAPI",
    "includeVersionHistory": true,
    "versionLimit": 10,
    "preserveAuthor": true,
    "includeItemPermissions": false,
    "fallbackUserAccount": "migration@contoso.com"
  },

  "userMappings": [
    { "source": "anna@contoso.com", "target": "anna.berg@fabrikam.com" }
  ]
}
```

### Top-level properties

| Property             | Type | Description                                                                                                               |
|----------------------|------|---------------------------------------------------------------------------------------------------------------------------|
| <code>jobType</code> | text | <code>CopyLists</code> or <code>CopySites</code> . Optional: defaults to the template's type, or <code>CopyLists</code> . |
| <code>jobName</code> | text | Name in History. Defaults to the template name, or job type and time.                                                     |

|                           |            |                                                                                                                                         |
|---------------------------|------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| <code>template</code>     | text       | Job template whose options are the starting point.                                                                                      |
| <code>copies</code>       | array      | At least one copy entry (below). Required.                                                                                              |
| <code>options</code>      | object     | Option overrides; same names as New-SPECopySettings (camelCase or PascalCase). Leave a property out to keep the template/default value. |
| <code>userMappings</code> | array      | <code>{ "source": "...", "target": "..."} </code> entries.                                                                              |
| <code>whatIf</code>       | true/false | Plan only.                                                                                                                              |

## Copy entry

| Property                          | Type          | Description                                                                                   |
|-----------------------------------|---------------|-----------------------------------------------------------------------------------------------|
| <code>source</code>               | URL           | Absolute URL of the source site. Required.                                                    |
| <code>target</code>               | URL           | Absolute URL of an existing target site. Required.                                            |
| <code>lists</code>                | array of text | Titles or URL names, wildcards allowed. Empty = every list a user would copy.                 |
| <code>excludeLists</code>         | array of text | Removed from the selection; wildcards allowed.                                                |
| <code>ignoreMissingLists</code>   | true/false    | Unmatched names are warnings instead of errors.                                               |
| <code>targetListTitle</code>      | text          | New title on the target. Requires exactly one list in <code>lists</code> , without wildcards. |
| <code>targetListUrlSegment</code> | text          | New URL name (no slashes). Same requirement.                                                  |
| <code>targetFolder</code>         | text          | List-relative folder on the target, e.g. <code>/Archive/2025</code> .                         |

## Validation rules

The job is checked before anything is sent to SharePoint, and again against SharePoint before copying. Any of these stops the job with exit code 4:

- no copies, or a source/target that is not an absolute `http(s)` URL;
- a rename with zero, several or wildcard lists; a URL name containing `/`;
- the same source site listed more than once with different list selections;
- a user mapping missing its source or target; `versionLimit` above 50000;
- a named list not found (unless `ignoreMissingLists`), or a list copied onto itself;
- the target permission check failing; the template not found or not a copy template.

## 14. Command line reference (SPE.exe) ADVANCED

The executable is `SharePoint Essentials Toolkit.exe` in the install folder. It is a Windows application, so from PowerShell start it with `Start-Process -Wait` (or pipe it, e.g. `| Out-Null`) to wait for it and get the exit code.

### Run a job spec

```
"SharePoint Essentials Toolkit.exe" --run-spec <spec.json> [--result <result.json>] [--whatif]
```

| Argument                             | Description                                                                                                        |
|--------------------------------------|--------------------------------------------------------------------------------------------------------------------|
| <code>--run-spec &lt;path&gt;</code> | The job spec file. Must be the first argument.                                                                     |
| <code>--result &lt;path&gt;</code>   | Where to write the result file. Written for every outcome, including invalid input. Folders are created if needed. |
| <code>--whatif</code>                | Plan only; overrides <code>"whatIf": false</code> in the file.                                                     |

```
# PowerShell
$exe = 'C:\Program Files\Cognillo\SharePoint Essentials Toolkit\SharePoint Essentials Toolkit.exe'
$p = Start-Process $exe -ArgumentList '--run-spec "C:\SPE\job.json" --result "C:\SPE\result.json"' -NoNewWindow -
Wait -PassThru
$p.ExitCode
Get-Content C:\SPE\result.json | ConvertFrom-Json

:: Command prompt / batch
start "" /wait "C:\...\SharePoint Essentials Toolkit.exe" --run-spec C:\SPE\job.json --result C:\SPE\result.json
echo %ERRORLEVEL%
```

### Result file

JSON with camelCase names; properties without a value are left out. Same fields as the result object.

```
{
  "jobId": "6f1b8e0a-3c1d-4c35-9a41-2f8f0c7d9b11",
  "jobName": "HR archive - October",
  "jobType": "CopyLists",
  "status": "CompletedWithWarnings",
  "exitCode": 2,
  "whatIf": false,
  "started": "2026-10-15T01:00:03",
  "completed": "2026-10-15T01:42:51",
  "durationSeconds": 2568.2,
  "listsTotal": 3, "listsCompleted": 3, "listsWithError": 0,
  "itemsTotal": 18240, "itemsCompleted": 18190, "itemsSkipped": 50, "itemsWithError": 0,
  "errors": 0, "warnings": 50,
  "jobHistoryFolder": "C:\\Users\\...\\jobs\\6f1b8e0a-...",
  "plan": [
    {
      "sourceSite": "https://contoso.sharepoint.com/sites/HR",
      "targetSite": "https://contoso.sharepoint.com/sites/HR-Archive",
      "sourceList": "Documents",
      "targetList": "Documents",
      "targetListUrl": "/sites/HR-Archive/Shared Documents",
      "targetListExists": true,
      "preflightStatus": "Success"
    }
  ],
  "messages": []
}
```

## Run a saved job template (any task type)

```
"SharePoint Essentials Toolkit.exe" --run-template "<template name>" [-url <site>] [-target <target site>]
```

- Without `-url` the template runs exactly as saved.
- Reports and other tasks: `-url` replaces the template's sites with that one site.
- Copy templates: `-url` is the **new source site**, and the template's lists are copied where the new source has them (missing ones are warnings). `-target` sets the target site; without it the template's own target sites are used.

```
"SharePoint Essentials Toolkit.exe" --run-template "Broken links" -url
https://contoso.sharepoint.com/sites/Intranet
"SharePoint Essentials Toolkit.exe" --run-template "HR nightly copy" -url
https://contoso.sharepoint.com/sites/HR-EU -target https://contoso.sharepoint.com/sites/HR-EU-Archive
```

The app's Job Templates page can copy a ready-to-paste `--run-template` command to the clipboard.

## Run a task from History

```
"SharePoint Essentials Toolkit.exe" --run-job "<task ID>"
```

Runs a task from History again with the same settings and sites. Use **PowerShell > Copy command** on the History page to get the command with the ID filled in.

## Collect user activity

```
"SharePoint Essentials Toolkit.exe" --run-useractivitysync
```

Collects SharePoint user activity from the Microsoft 365 audit log for the activity reports. User Activity must be set up in the app first.

## Logs

Each command-line run writes a log to `%APPDATA%\Cognillo\SharePoint Essentials Toolkit\Logs\PowerShell Jobs`. The full job report is in the app's History page.

## 15. PowerShell command reference ADVANCED

Run `Get-Help <command> -Full` for built-in help and examples.

### Copy-SPEList

```
Copy-SPEList -SourceSite <url> -DestinationSite <url> [-List <string[]>] [-ExcludeList <string[]>]
[-NewListTitle <string>] [-NewListUrlName <string>] [-TargetFolder <string>] [-IgnoreMissingLists]
[-Template <string>] [-JobName <string>] [-CopySettings <object>] [-UserMapping <object[]>]
[-ExportSpec <path>] [-WhatIf] [-Confirm]
```

### Copy-SPESite

```
Copy-SPESite -SourceSite <url> -DestinationSite <url> [-List <string[]>] [-ExcludeList <string[]>]
[-IgnoreMissingLists] [-StructureOnly]
[-Template <string>] [-JobName <string>] [-CopySettings <object>] [-UserMapping <object[]>]
[-ExportSpec <path>] [-WhatIf] [-Confirm]
```

| Parameter                                                    | Description                                                                                                       |
|--------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------|
| <code>-SourceSite</code>                                     | Source site URL. Pipeline by property name (aliases: Source, SourceUrl, SourceSiteUrl).                           |
| <code>-DestinationSite</code>                                | Target site URL. Pipeline by property name (aliases: Target, TargetSite, TargetUrl, Destination, DestinationUrl). |
| <code>-List</code>                                           | Lists to copy: titles or URL names, wildcards, ; -separated in CSV. Omit for every list.                          |
| <code>-ExcludeList</code>                                    | Lists to leave out.                                                                                               |
| <code>-NewListTitle</code> ,<br><code>-NewListUrlName</code> | Copy-SPEList: rename a single list on the target.                                                                 |
| <code>-TargetFolder</code>                                   | Copy-SPEList: folder in the target list.                                                                          |
| <code>-IgnoreMissingLists</code>                             | Lists not found are warnings, not errors.                                                                         |
| <code>-StructureOnly</code>                                  | Copy-SPESite: create lists and libraries empty.                                                                   |
| <code>-Template</code>                                       | Job template providing the options.                                                                               |
| <code>-JobName</code>                                        | Name in History.                                                                                                  |
| <code>-CopySettings</code>                                   | Output of <code>New-SPECopySettings</code> .                                                                      |
| <code>-UserMapping</code>                                    | Output of <code>Import-SPEUserMapping</code> , objects with Source/Target, or a hashtable.                        |
| <code>-ExportSpec</code>                                     | Write the job spec file instead of running; returns the file.                                                     |
| <code>-WhatIf</code>                                         | Plan only: reads SharePoint, copies nothing, returns the plan.                                                    |
| <code>-Confirm</code>                                        | Ask before running. Copy-SPESite asks by default; use <code>-Confirm:\$false</code> to skip.                      |

## Other commands

| Command                                                                                                 | Description                                                                                                                                                                                                                                                                          |
|---------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>New-SPECopySettings</code>                                                                        | Builds copy settings; see section 8 for every parameter.                                                                                                                                                                                                                             |
| <code>Import-SPEUserMapping -Path &lt;csv&gt; [-Delimiter &lt;char&gt;]</code>                          | Reads user mappings from CSV.                                                                                                                                                                                                                                                        |
| <code>Invoke-SPECopyJob -Path &lt;json&gt; [-WhatIf]</code>                                             | Runs a job spec file. Accepts files from the pipeline<br>( <code>Get-ChildItem *.json   Invoke-SPECopyJob</code> ).                                                                                                                                                                  |
| <code>Invoke-SPEJobTemplate -Name &lt;template&gt; [-Url &lt;site&gt;] [-TargetUrl &lt;site&gt;]</code> | Runs a saved job template of any type. Sites can be piped in as text or as objects with a <code>Url</code> , <code>SiteUrl</code> or <code>SourceSite</code> property; each site is a separate run. Copy templates also take <code>TargetUrl</code> / <code>DestinationSite</code> . |
| <code>Invoke-SPEJob -Id &lt;task ID&gt;</code>                                                          | Runs a task from History again. IDs can be piped in.                                                                                                                                                                                                                                 |
| <code>Sync-SPEUserActivity</code>                                                                       | Collects user activity from the Microsoft 365 audit log.                                                                                                                                                                                                                             |
| <code>Get-SPEExecutable</code>                                                                          | Path of the executable used (from <code>Set-SPEExecutable</code> , <code>\$env:SPE_EXE</code> , the installed product, or Program Files).                                                                                                                                            |
| <code>Set-SPEExecutable -Path &lt;file or folder&gt;</code>                                             | Uses a specific installation for this session.                                                                                                                                                                                                                                       |

## 16. Troubleshooting and FAQ EVERYONE

| Message or symptom                                                      | What to do                                                                                                                                                            |
|-------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| SharePoint Essentials Toolkit.exe was not found                         | Run <code>Set-SPEExecutable -Path '&lt;install folder&gt;'</code> or set <code>\$env:SPE_EXE</code> .                                                                 |
| No list named 'X' was found on '...'                                    | Check the title or URL name in the browser. Use a wildcard, or <code>-IgnoreMissingLists</code> if the list is optional.                                              |
| Source site '...' appears more than once with different list selections | Within one job a source can have only one list selection. Combine the rows, or run separate commands.                                                                 |
| targetListTitle and targetListUrlSegment need exactly one list          | Renaming works for one named list per row/entry, without wildcards.                                                                                                   |
| Permission check failed on the target                                   | The signed-in account needs edit rights (site owner recommended) on the target site.                                                                                  |
| Sign-in errors, or the job cannot reach SharePoint                      | Open the app as the same Windows user and sign in to the tenant again; scripts reuse that connection.                                                                 |
| No Copy Sites or Copy Lists job template named '...' was found          | Use the exact name from the Job Templates page; it must be a Copy Sites or Copy Lists template.                                                                       |
| Job template not found: '...'                                           | <code>Invoke-SPEJobTemplate</code> : use the exact name shown on the Job Templates page.                                                                              |
| A report ran but the results are not where I expected                   | Open the task on the History page. Email or SharePoint delivery happens only when the template is set up to deliver the report.                                       |
| The script stops at "Are you sure...?"                                  | Copy-SPESite asks for confirmation. Add <code>-Confirm:\$false</code> .                                                                                               |
| I need more detail about a failure                                      | See <code>\$result.Messages</code> , the job's report in History, and the log in <code>%APPDATA%\Cognillo\SharePoint Essentials Toolkit\Logs\PowerShell Jobs</code> . |

### Frequently asked questions

#### Does -WhatIf connect to SharePoint?

Yes. It signs in and reads both sites (read-only) so it can tell you exactly which lists would be copied, whether they already exist on the target and whether your permissions are sufficient. It never writes to SharePoint.

#### Can I run several copies at the same time?

Yes, but it is usually better to put them into one job (pipe them into one command); you get a single entry in History and a single report, and the copies don't compete with each other for SharePoint throttling limits.

#### Can I stop a running copy?

Press `Ctrl+C` in the console. The job stops cleanly, is marked *Cancelled* in History, and returns exit code 3.

#### Does the copy remove anything from the source?

No. Copies never change the source site.

#### Can I run a copy again to pick up changes?

Yes. Use `OnItemExists Update` (or `Skip`) in the copy settings so items already on the target are updated or left alone instead of overwritten.

### **Do scripted jobs count against my license?**

Scripted jobs follow the same license rules as jobs you start in the app.

---

© Cognillo. SharePoint Essentials Toolkit – PowerShell Scripting Guide, module version 1.0.  
SharePoint and Microsoft 365 are trademarks of Microsoft Corporation.