

# PowerShell TreeView Generator Scripts

## Overview

Three PowerShell scripts designed to generate directory structure views and extract file contents for documentation and analysis purposes. These scripts provide different levels of detail from simple tree views to complete content extraction.

**Important:** YOURPATHHERE, USERPROFILE, and filename are placeholders that must be replaced with your actual parameters before executing the commands.

## Script 1: TreeView Generator 1.0

### Purpose

Generates a simple directory tree structure showing all files and folders within a specified path, recursively scanning all subdirectories.

### Command

#### TreeView Generator 1.0

```
Get-ChildItem -Path "YOURPATHHERE" -Recurse | Select-Object FullName | clip
```

### Parameters

Parameter	Description	Example
YOURPATHHERE	The full path to the folder you want to scan	C:\Projects\MyProject

### Output

Copies the full path of every file and folder to the clipboard in a list format.

### Example Output

```
C:\Projects\MyProject\  
- README.md  
- package.json  
- index.js  
src\  
  - app.js  
  - Header.js  
  - styles.css
```

```
\docs\
- api.md
- guide.md

\tests\
- app.test.js
- integration.test.js
```

### Use Cases

- Quick directory structure documentation
- Project file inventory
- Folder organization analysis

## Script 2: TreeView with Content 1.0

### Purpose

Generates a tree structure with full file paths followed by the complete content of each file, copying everything to the clipboard.

### Command

#### TreeView with Content 1.0

```
Get-ChildItem -Path "YOURPATHHERE" -Recurse -File | ForEach-Object { $_.FullName; Get-Content $_.FullName } | clip
```

### Parameters

Parameter	Description	Example
<b>YOURPATHHERE</b>	The full path to the folder containing files to extract	C:\Projects\MyProject

### Output

Copies file paths and their contents to the clipboard. Each file path is followed by its complete content.

### Example Output

```
C:\Projects\MyProject\
- main.js
- config.json
\src\
- app.js
- styles.css

[FILE CONTENT]
C:\Projects\MyProject\main.js
console.log("Hello World");
function main() {
    // Application code
```

```
}

C:\Projects\MyProject\config.json
{
  "name": "MyProject",
  "version": "1.0.0"
}

C:\Projects\MyProject\src\app.js
import React from 'react';
const App = () => {
  return
App Component
;
}

C:\Projects\MyProject\src\styles.css
body {
  margin: 0;
  padding: 0;
}
```

Use Cases

- Code review preparation
- Complete project backup to clipboard
- Quick content migration

Script 3: TreeView + Content to File

Purpose

Generates a tree structure with file contents and saves everything to a text file on the desktop instead of clipboard.

Command

TreeView + Content to File

```
Get-ChildItem -Path "YOURPATHHERE" -Recurse -File | ForEach-Object { $_.FullName; Get-Content $_.FullName
} | Out-File "$env:USERPROFILE\Desktop\filename.txt"
```

Parameters

Parameter	Description	Example
YOURPATHHERE	The full path to the folder to scan	C:\Projects\MyProject
USERPROFILE	Windows user profile folder (usually auto-detected via \$env:)	\$env:USERPROFILE (automatic)
filename	Name for the output text file	project_backup

## Output Location

Creates a text file on the desktop with the specified filename containing all paths and file contents.

## Advantages

- No clipboard size limitations
- Permanent file output
- Better for large directory structures
- Can handle binary files without corruption