

PowerShell handles input and output through several cmdlets and techniques, including reading from the console, writing to the console, and formatting output for readability.

## Input

- **Read-Host:** This cmdlet prompts the user for input and stores it as a string.

```
$name = Read-Host "Please enter your name"  
Write-Host "Hello, $name!"
```

- **Get-Content:** Reads the content of a file.

```
$content = Get-Content -Path "C:\example.log"
```

## Output

- **Write-Host:** Displays output directly to the console.

```
Write-Host "This is a message."
```

- **Out-File:** Writes output to a file.

```
Get-Process | Out-File -FilePath "processes.txt"
```

## Formatting

- **Format-Table:** Displays output in a table format.

```
Get-Process | Format-Table -Property Name, CPU, StartTime
```

- **Format-List:** Displays output as a list of properties.

```
Get-Service | Format-List -Property Name, Status, DisplayName
```

- **Format-Wide:** Displays output in a wide format, showing only one property per line.

```
Get-ChildItem | Format-Wide -Column 3
```

## String Formatting

- The -f format operator: Allows for composite formatting, similar to string interpolation in other languages.

```
$name = "John"  
$age = 30  
"My name is {0} and I am {1} years old" -f $name, $age
```

## Pipelines

- PowerShell uses pipelines (|) to pass the output of one command as input to the next. This is fundamental for processing and formatting data.

```
Get-Process | Sort-Object CPU -Descending | Select-Object -First 5 | Format-Table Name, CPU
```

These cmdlets and techniques provide a robust way to handle input, output, and formatting in PowerShell, making it a powerful tool for scripting and automation.