PowerShell Syntax Overview

PowerShell syntax is designed to be intuitive and consistent, making it easier to construct and execute commands. Below is a concise guide to understanding its key components:

1. Cmdlet Structure

PowerShell commands, called **cmdlets**, follow a `Verb-Noun` naming convention:

```
Get-Process
Set-Item
New-Object
```

- **Verb**: Describes the action (e.g., Get, Set, New, Remove).
- **Noun**: Specifies the object being acted upon (e.g., Process, Item, Object).

2. Parameters

Cmdlets often include parameters to modify their behavior:

```
Get-Process -Name "notepad"
```

- **Named Parameters**: Explicitly specify options (e.g., `-Name`).
- **Positional Parameters**: Omit the parameter name if the position is known.

3. Variables

Variables are prefixed with `$` and can store data:

```
$myVariable = "Hello, PowerShell!"
```

4. Pipelines

Use `|` to pass output from one cmdlet to another:

```
Get-Process | Where-Object {$_.CPU -gt 100}
```

5. Comparison Operators

PowerShell uses specific operators for comparisons:

- `-eq` (equals), `-ne` (not equals)
- `-gt` (greater than), `-lt` (less than)
- `-like` (wildcard match), `-match` (regex match)

Example:

```
if ($value -eq 10) { Write-Output "Value is 10" }
```

6. Loops and Conditionals

- **ForEach**: Iterates through a collection.
- **If/Else**: Conditional logic.
- **While**: Repeats while a condition is true.

Example:

```
foreach ($item in $collection) {
    Write-Output $item
}
```

7. Comments

- Single-line: `# This is a comment`
- Multi-line: `<# This is a multi-line comment #>`

8. Common Cmdlets

- **Get-Help**: Displays help for a cmdlet.

  ```
  Get-Help Get-Process
  ```

- **Get-Command**: Lists available cmdlets.

  ```
  Get-Command
  ```

- **Get-Member**: Shows properties and methods of an object.

  ```
  Get-Process | Get-Member
  ```

This is just a starting point! PowerShell is a powerful scripting language with extensive capabilities for automation and system management.