Powershell command-lets Intune Application Packaging

Group C

What are Powershell Cmdlets?

- Cmdlets are specialized commands in PowerShell used for system tasks.
- They are not standalone executables like ipconfig.exe or notepad.exe; instead, they run within the PowerShell environment.
- Cmdlets are .NET-based, which means they return objects (not plain text like old Command Prompt commands).

Categories of powershell cmdlets:

- 1. File system and Data management
- 2. Registry management
- 3. Process and service control
- 4. Software installation and package management
- 5. Networking
- 6. Security and permissions
- 7. System information and administration
- 8. Scripting and utility
- 9. Cloud and remote management

PowerShell Cmdlet Naming Conventions:

- **1.Structure:**
- A cmdlet name always follows this format:
- Verb → The action to perform
- Noun → The item, object, or resource you’re working with
- **2.Rules for Verbs:**
- Microsoft maintains an approved verb list to keep cmdlets consistent.
- Examples of approved verbs:
- Get → Retrieve information
- Set → Modify something
- New → Create something
- Remove → Delete something

PowerShell Cmdlet Naming Conventions

- Start → Begin a process or service
- Stop → End a process or service
- **3.Rules for Nouns**:
- Nouns should be singular, even if they refer to multiple items.
- □ Get-Process
- □ Get-Processes
- Nouns describe what the verb is acting on (process, file, registry, etc.).
- **Examples**: Get-Service - Shows details of services
- Start-Service - Starts a service

Why PowerShell Cmdlets Are Important in Application Packaging

- **Automation:** Speeds up repetitive tasks like installing, configuring, and validating applications.
- **Precision:** Gives granular control over system settings, registry entries, and file permissions.
- **Integration:** Works seamlessly with MSI, App-V, Azure, and other Microsoft deployment tools.
- **Flexibility**: Allows customization for unique application requirements.
- **Cross-Platform Support:** Can handle packaging in mixed OS environments with PowerShell Core.
- **Efficiency:** Reduces errors and ensures consistent deployments across sys

Powershell Cmdlets

- Cmdlets are small, task-focused PowerShell commands
- Always in Verb-Noun format (e.g., Get-Process)
- Automate repetitive tasks in application packaging
- Manage files, registry, permissions, and apps efficiently
- Help install, uninstall, and configure software easily
- Save time, reduce mistakes, and improve consistency
- Scripts can be reused for future packaging work
- Essential skill for modern IT and software deployment

What is Intune Application Packaging?

- In Microsoft Intune, application packaging means preparing your app in a format that Intune can upload, manage, and deploy to devices.
- Most Windows apps in Intune use the .intunewin format.

How It Works

- **1. Prepare the Installer**
- You start with your normal app installer (e.g., .msi, .exe, .msix).This installer should be ready for silent installation (so no pop-ups or clicks required).
- Example: setup.exe /silent or app.msi /quiet.

How It Works

- **2. Use the Win32 Content Prep Tool**
- Microsoft gives a free tool called IntuneWinAppUtil.exe.
- This tool wraps your installer into a special .intunewin package.
- **Command example:**
- IntuneWinAppUtil.exe -c "C:\SourceFolder" -s "setup.exe" -o "C:\OutputFolder“
- -c = folder where your installer is
- -s = installer file name
- -o = output location for .intunewin

How It Works

- **3. Upload to IntuneIn Intune Admin Center**:
- Go to Apps > Windows > Add.
- Select Windows app (Win32).
- Upload your .intunewin file.
- Fill in details like install command, uninstall command, requirements, and detection rules.

How It Works

- **4. Assign the App Decide who gets the app:**
- All devices
- All users
- Specific groups
- **Intune will push the app to those devices automatically**.

How It Works

- 5. **App Installs on Endpoints**
- When assigned devices check in with Intune, they:
- Download the .intunewin package.
- Install the app silently in the background.
- Report back to Intune if installation was successful.

Real-Life Example

- Your company needs to install Zoom on 500 laptops:
- You take the Zoom installer (ZoomInstallerFull.msi).
- Wrap it into .intunewin using the Content Prep Tool.
- Upload and assign it in Intune.
- Within hours, all laptops get Zoom automatically, without anyone manually installing it.