Powershell App Deploy Toolkit & MSI Tables

GROUP C

What is Powershell App Deploy Toolkit?

Overview

Key features

- Standardized deployment framework
- User interaction
- Process and application handling
- Supports any installer type
- Pre and Post installation actions
- Built in logging
- Error handling and exit codes
- Compatibility with enterprise deployment tools
- Highly customizable

How Powershell App Deploy ToolKit works?

- Powershell App Deploy Kit typically involves a main PowerShell script (Deploy-Application.ps1) that orchestrates the deployment process. This script utilizes functions provided by the toolkit to perform tasks such as:
- Installation/Uninstallation: Executing vendor installers (MSI, EXE) with appropriate switches.
- File Operations: Copying, moving, or deleting files and folders.
- Registry Modifications: Creating, modifying, or deleting registry keys and values.
- Service Management: Starting, stopping, or configuring Windows services.
- User Interaction: Displaying messages, prompts, and progress to the user.
- Logging: Recording deployment actions and outcomes for auditing and troubleshootin

Why PSADT in Application Packaging

- Reduces custom scripting effort
- Improves deployment consistency
- Provides pre-built functions (logging, prompts, UI)
- Handles user interaction & detection logic
- Easy to maintain

Benefits

- Saves time in packaging projects
- Standardized deployment across apps
- Better user experience with clear prompts
- Simplified troubleshooting via logs

What is an MSI?

- Microsoft Installer (MSI) is a package format for installing Windows applications.
- Works like a mini database containing installation instructions.
- Controlled by Windows Installer service.

What Are MSI Tables?

- Structured data sheets inside an MSI file.
- Each table stores specific instructions for installation.
- Similar to Excel sheets — rows & columns define install details.

Key Tables and Their Roles

Table Name	Purpose
File	Lists the files to install and their locations
Component	Groups files, registry keys, etc., into installable units
Feature	Defines installable features for the user to choose
Directory	Defines target folders for installation.
Registry	Lists registry changes to be made during installation.
Custom Action	Defines custom scripts or commands to run
Property	Stores variables like installation paths or product names

Why MSI Tables Matter

- **Customization:**
- Change install paths or options using transforms (MST).
- **Troubleshooting:**
- Identify missing files or errors.
- **Automation:**
- Works with deployment tools like SCCM/Intune.
- **Standardization:**
- Same format across applications.

Tools to View/Edit

- Orca – Official Microsoft MSI table editor.
- InstEd – User-friendly alternative.
- LessMSI – View table data without installing.
- WiX Toolset – Create and modify MSI packages

Conclusion

- **MSI files are more than just installers** —
- They are structured databases that define every step of how an application is installed on Windows. Inside, MSI tables act like blueprints, detailing which files to copy, folders to create, registry changes to apply, and custom actions to run.
- By understanding these tables, we gain the ability to customize installations, troubleshoot problems, and automate deployments with tools like SCCM or Intune. Using editors like Orca or InstEd, we can inspect and modify these instructions to meet specific business or technical needs.
- In short, mastering MSI tables means gaining precise control over application deployment, ensuring installations are consistent, efficient, and tailored to requirements