Universal-AppInstaller v2.0.0 - Complete Guide

🎯 What's New in v2.0.0

Major Improvements:

1. ✅ **Relative Path Support** - Portable across environments (local, network, USB)
2. ✅ **SourcePath Parameter** - Explicitly specify installer location
3. ✅ **Better Registry Detection** - Separate parameters for registry key, value, and data
4. ✅ **Empty String Logging** - Allows blank lines in output
5. ✅ **Wildcard File Detection** - Support for version-specific folders
6. ✅ **Enhanced Error Handling** - Better error messages and validation

---

📋 Quick Start - Adobe Reader Example

Your Directory Structure:

```
C:\Deploy\
├── Installers\
│   └── Apps\
│       └── AdobeReader\
│           └── AcroRdrDC.exe
└── Scripts\
    ├── Orchestration-Config.ps1
    └── Phase4-Applications\
        └── Universal-AppInstaller.ps1
```

Configuration in Orchestration-Config.ps1:

```
powershell
```

```
@{
    TaskID = "APP-002"
    TaskName = "Install Adobe Acrobat Reader"
    ScriptPath = "Phase4-Applications\Universal-AppInstaller.ps1"
    Enabled = $true
    Timeout = 900
    RunAs = "SYSTEM"
    RequiresReboot = $false
    AllowRetry = $true
    Critical = $false
    Description = "Installs Adobe Acrobat Reader DC"
    Parameters = @{
        # Application Info
        AppName = "Adobe Acrobat Reader DC"

        # Installer Location (RELATIVE PATH!)
        SourcePath = "Installers\Apps\AdobeReader"  # ← Relative to Deploy root
        InstallerFileName = "AcroRdrDC.exe"
        InstallerType = "EXE"
        InstallArguments = "/sAll /rs /msi EULA_ACCEPT=YES"

        # Detection (prevents reinstalling)
        DetectionMethod = "File"
        DetectionPath = "C:\Program Files (x86)\Adobe\Acrobat Reader DC\Reader\AcroRd32.exe"

        # Logging
        LogPath = "C:\ProgramData\OrchestrationLogs\Apps"
    }
}
```

🔧 Parameter Reference

Required Parameters:

Parameter	Description	Example
`AppName`	Display name of application	`"Adobe Acrobat Reader DC"`
`InstallerFileName`	Installer file name	`"AcroRdrDC.exe"`

Parameter	Description	Example
`DetectionMethod`	How to detect if installed	`"File"` or `"Registry"`

Important Parameters:

Parameter	Description	Example
`SourcePath`	Path to installer (relative or absolute)	`"Installers\Apps\Chrome"`
`InstallerType`	MSI, EXE, MSIX, APPX, AUTO	`"EXE"`
`InstallArguments`	Silent install arguments	`"/S"` or `"/quiet"`

Detection Parameters:

File Detection:

```powershell
DetectionMethod = "File"
DetectionPath = "C:\Program Files\7-Zip\7z.exe"
```

Registry Detection (Simple):

```powershell
DetectionMethod = "Registry"
DetectionRegistry = "HKLM:\SOFTWARE\Microsoft\Office\ClickToRun\Configuration"
```

Registry Detection (Advanced):

```powershell
DetectionMethod = "Registry"
DetectionRegistry = "HKLM:\SOFTWARE\7-Zip"
DetectionRegistryValue = "Path"
DetectionRegistryData = "C:\Program Files\7-Zip\"
```

📂 Path Types

Relative Paths (Recommended):

Benefits: Portable across environments

```powershell
# Works from:
# - C:\Deploy\
# - \\Server\Deploy\
# - E:\Deploy\ (USB)
SourcePath = "Installers\Apps\Chrome"
```

How it works:

```
Script location:  Deploy\Scripts\Phase4-Applications\Universal-AppInstaller.ps1
                  ↓
Goes up 2 levels:  Deploy\
                  ↓
Appends path:      Installers\Apps\Chrome\
                  ↓
Final result:      C:\Deploy\Installers\Apps\Chrome\ (or wherever Deploy is)
```

Absolute Paths:

Use when: Installers are in a different location

```powershell
# Local absolute
SourcePath = "C:\Software\Installers\Chrome"

# Network absolute
SourcePath = "\\FileServer\Software\Chrome"
```

🎯 Complete App Examples

1. Adobe Acrobat Reader DC

```powershell
@{
    TaskID = "APP-002"
    TaskName = "Install Adobe Acrobat Reader"
    ScriptPath = "Phase4-Applications\Universal-AppInstaller.ps1"
    Enabled = $true
    Timeout = 900
    Parameters = @{
        AppName = "Adobe Acrobat Reader DC"
        SourcePath = "Installers\Apps\AdobeReader"
        InstallerFileName = "AcroRdrDC.exe"
        InstallerType = "EXE"
        InstallArguments = "/sAll /rs /msi EULA_ACCEPT=YES"
        DetectionMethod = "File"
        DetectionPath = "C:\Program Files (x86)\Adobe\Acrobat Reader DC\Reader\AcroRd32.exe"
    }
}
```

2. Google Chrome

```powershell
@{
    TaskID = "APP-003"
    TaskName = "Install Google Chrome"
    ScriptPath = "Phase4-Applications\Universal-AppInstaller.ps1"
    Enabled = $true
    Timeout = 300
    Parameters = @{
        AppName = "Google Chrome"
        SourcePath = "Installers\Apps\Chrome"
        InstallerFileName = "ChromeSetup.exe"
        InstallerType = "EXE"
        InstallArguments = "/silent /install"
        DetectionMethod = "File"
        DetectionPath = "C:\Program Files\Google\Chrome\Application\chrome.exe"
    }
}
```

3. 7-Zip

```powershell
@{
    TaskID = "APP-007"
    TaskName = "Install 7-Zip"
    ScriptPath = "Phase4-Applications\Universal-AppInstaller.ps1"
    Enabled = $true
    Timeout = 180
    Parameters = @{
        AppName = "7-Zip"
        SourcePath = "Installers\Apps\7Zip"
        InstallerFileName = "7z2301-x64.exe"
        InstallerType = "EXE"
        InstallArguments = "/S"
        DetectionMethod = "File"
        DetectionPath = "C:\Program Files\7-Zip\7z.exe"
    }
}
```

4. Mozilla Firefox

```powershell
@{
    TaskID = "APP-004"
    TaskName = "Install Firefox"
    ScriptPath = "Phase4-Applications\Universal-AppInstaller.ps1"
    Enabled = $true
    Timeout = 600
    Parameters = @{
        AppName = "Mozilla Firefox"
        SourcePath = "Installers\Apps\Firefox"
        InstallerFileName = "Firefox Setup.exe"
        InstallerType = "EXE"
        InstallArguments = "/S"
        DetectionMethod = "File"
        DetectionPath = "C:\Program Files\Mozilla Firefox\firefox.exe"
    }
}
```

5. Notepad++

```powershell
@{
    TaskID = "APP-011"
    TaskName = "Install Notepad++"
    ScriptPath = "Phase4-Applications\Universal-AppInstaller.ps1"
    Enabled = $true
    Timeout = 300
    Parameters = @{
        AppName = "Notepad++"
        SourcePath = "Installers\Apps\Notepad++"
        InstallerFileName = "npp.8.6.9.Installer.x64.exe"
        InstallerType = "EXE"
        InstallArguments = "/S"
        DetectionMethod = "File"
        DetectionPath = "C:\Program Files\Notepad++\notepad++.exe"
    }
}
```

6. VLC Media Player

```powershell
@{
    TaskID = "APP-012"
    TaskName = "Install VLC"
    ScriptPath = "Phase4-Applications\Universal-AppInstaller.ps1"
    Enabled = $true
    Timeout = 300
    Parameters = @{
        AppName = "VLC Media Player"
        SourcePath = "Installers\Apps\VLC"
        InstallerFileName = "vlc-3.0.20-win64.exe"
        InstallerType = "EXE"
        InstallArguments = "/L=1033 /S"
        DetectionMethod = "File"
        DetectionPath = "C:\Program Files\VideoLAN\VLC\vlc.exe"
    }
}
```

7. Microsoft Teams (Wildcard Detection)

```powershell
@{
    TaskID = "APP-006"
    TaskName = "Install Microsoft Teams"
    ScriptPath = "Phase4-Applications\Universal-AppInstaller.ps1"
    Enabled = $true
    Timeout = 900
    Parameters = @{
        AppName = "Microsoft Teams"
        SourcePath = "Installers\Apps\Teams"
        InstallerFileName = "Teams_windows_x64.exe"
        InstallerType = "EXE"
        InstallArguments = "-s"
        DetectionMethod = "File"
        DetectionPath = "C:\Program Files\WindowsApps\MSTeams_*\msteams.exe"  # ← Wildcard!
    }
}
```

8. Microsoft 365 (Registry Detection)

```powershell
@{
    TaskID = "APP-001"
    TaskName = "Install Microsoft 365"
    ScriptPath = "Phase4-Applications\Universal-AppInstaller.ps1"
    Enabled = $true
    Timeout = 1800
    Parameters = @{
        AppName = "Microsoft 365 Apps"
        SourcePath = "Installers\Apps\Office"
        InstallerFileName = "setup.exe"
        InstallerType = "EXE"
        InstallArguments = "/configure configuration.xml"
        DetectionMethod = "Registry"
        DetectionRegistry = "HKLM:\SOFTWARE\Microsoft\Office\ClickToRun\Configuration"
        DetectionRegistryValue = "VersionToReport"
    }
}
```

🔍 How to Find Detection Info

For File Detection:

```
powershell

# After manually installing the app, find the executable:
Get-ChildItem "C:\Program Files" -Recurse -Filter "*.exe" |
    Where-Object { $_.Name -like "*AppName*" } |
    Select-Object FullName
```

For Registry Detection:

```
powershell

# Find app in Add/Remove Programs:
Get-ItemProperty "HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\*" |
    Where-Object { $_.DisplayName -like "*AppName*" } |
    Select-Object DisplayName, PSPath, DisplayVersion

# Also check 32-bit registry:
Get-ItemProperty "HKLM:\SOFTWARE\WOW6432Node\Microsoft\Windows\CurrentVersion\Uninstall\*" |
    Where-Object { $_.DisplayName -like "*AppName*" } |
    Select-Object DisplayName, PSPath, DisplayVersion
```

🧪 Testing

Test Individual App:

```
powershell
```

```powershell
# Test Adobe Reader installation
cd C:\Deploy\Scripts\Phase4-Applications

.\Universal-AppInstaller.ps1 `
    -AppName "Adobe Acrobat Reader DC" `
    -SourcePath "Installers\Apps\AdobeReader" `
    -InstallerFileName "AcroRdrDC.exe" `
    -InstallerType "EXE" `
    -InstallArguments "/sAll /rs /msi EULA_ACCEPT=YES" `
    -DetectionMethod "File" `
    -DetectionPath "C:\Program Files (x86)\Adobe\Acrobat Reader DC\Reader\AcroRd32.exe"
```

Test via Orchestration:

```
powershell

# Run Phase 4 only
cd C:\Deploy\Scripts
.\Orchestration-Master.ps1 -Phase Phase4
```

📊 Exit Codes

Code	Meaning	Action
0	Success	App installed successfully
10	Already installed	App detected, skipped installation
1	General failure	Check logs for details
2	Installer not found	Verify SourcePath and InstallerFileName
3	Detection failed	Check detection parameters
4	Installation failed	Check InstallArguments and installer
5	Validation failed	App didn't install correctly

✅ Checklist for Adding New Apps

- [ ] Download installer to: `C:\Deploy\Installers\Apps\AppName\`
- [ ] Test manual installation to find detection path
- [ ] Add configuration to Orchestration-Config.ps1 Phase 4
- [ ] Set correct `SourcePath` (relative path)
- [ ] Set correct `InstallerFileName`
- [ ] Set correct `InstallArguments` (silent install)
- [ ] Set correct `DetectionMethod` and `DetectionPath`
- [ ] Set appropriate `Timeout` value
- [ ] Test installation: `.\Orchestration-Master.ps1 -Phase Phase4`
- [ ] Verify app installs successfully
- [ ] Verify app is detected (doesn't reinstall on second run)

🎉 Benefits of Universal Template

Before (Custom Scripts):

```
Phase4-Applications\
├── Install-Chrome.ps1          ← 200 lines
├── Install-AdobeReader.ps1     ← 250 lines
├── Install-7Zip.ps1            ← 180 lines
├── Install-Teams.ps1           ← 300 lines
├── Install-Office.ps1          ← 400 lines
└── ... (20 more scripts!)      ← Thousands of lines!
```

After (Universal Template):

```
Phase4-Applications\
└── Universal-AppInstaller.ps1  ← ONE file (600 lines)
                                ← Handles ALL apps!
```

Advantages:

1. ✅ **One file to maintain** instead of 20+
2. ✅ **Standardized behavior** across all apps
3. ✅ **Built-in detection -** no reinstalls

4. ✅ **Portable -** works on any drive/network
5. ✅ **Consistent logging** for all apps
6. ✅ **Easy to add apps -** just add config!
7. ✅ **Less code -** thousands of lines reduced

🚀 Migration from Custom Scripts

Step 1: Replace Custom Scripts

Delete all custom Install-*.ps1 files:

```powershell
Remove-Item "C:\Deploy\Scripts\Phase4-Applications\Install-*.ps1"
```

Step 2: Place Universal Template

Copy Universal-AppInstaller.ps1 to:

```
C:\Deploy\Scripts\Phase4-Applications\Universal-AppInstaller.ps1
```

Step 3: Update Config

Change all ScriptPath entries:

```powershell
# OLD:
ScriptPath = "Phase4-Applications\Install-Chrome.ps1"

# NEW:
ScriptPath = "Phase4-Applications\Universal-AppInstaller.ps1"
```

Step 4: Add Detection Parameters

Add detection to each app:

```powershell
```

```
Parameters = @{
    # ... existing parameters ...

    # ADD THESE:
    DetectionMethod = "File"
    DetectionPath = "C:\Program Files\AppName\app.exe"
}
```

📝 Version History

v2.0.0 (2024-12-11)

- Added SourcePath parameter with relative path support
- Added Resolve-DeployPath function for portability
- Added DetectionRegistry parameters for better registry detection
- Fixed empty string logging support
- Improved error handling and validation
- Enhanced detection methods with wildcard support

v1.0.0 (2024-12-08)

- Initial release
- Basic MSI/EXE support
- File and Registry detection
- Standard logging

Document Version: 2.0.0
Date: 2024-12-11
Author: IT Infrastructure Team
Status: Production Ready ✅