🚀 UNIVERSAL APP INSTALLER FOR ORCHESTRATION FRAMEWORK

Package Contents

This package contains a complete Universal App Installer solution for your PowerShell orchestration framework. With this template, you can install **80% of your applications** using a single script with simple configuration.

📦 What's Included

1. Universal-AppInstaller.ps1 (27 KB)

The main PowerShell script that installs applications based on parameters passed from your configuration file.

Features:

- ✅ Supports MSI, EXE, MSIX, APPX installers
- ✅ Multiple detection methods (Registry, File, AppX, Package, Custom)
- ✅ Automatic installer discovery in multiple search paths
- ✅ Version comparison and validation
- ✅ Pre/post installation script support
- ✅ Comprehensive logging and error handling
- ✅ Timeout protection
- ✅ Standard exit codes for orchestration integration

2. Universal-AppInstaller-Documentation.md (21 KB)

Comprehensive documentation covering:

- Overview and features
- When to use vs. custom scripts
- Exit code reference
- 10+ detailed configuration examples
- Detection method guide

- Common silent install switches for popular apps
- Troubleshooting section
- Best practices
- Quick reference templates

3. Phase4-Applications-Sample.ps1 (26 KB)

Ready-to-use configuration file with **20+ pre-configured applications:**

Included Applications:

- 7-Zip, WinRAR
- Notepad++, Visual Studio Code, Sublime Text
- Adobe Acrobat Reader, Foxit PDF Reader
- VLC Media Player, K-Lite Codec Pack
- PuTTY, WinSCP, FileZilla
- Git for Windows, TortoiseGit
- Paint.NET, IrfanView
- TreeSize Free, CCleaner
- Mozilla Firefox
- Zoom Client
- Java JRE
- LibreOffice
- Python
- Windows Terminal
- Greenshot

4. Universal-AppInstaller-QuickStart.md (13 KB)

Step-by-step guide to get started in 5 minutes:

- Installation instructions
- Configuration examples
- Testing procedures

- Common scenarios
- Troubleshooting quick reference
- Success checklist

🎯 Quick Start

Step 1: Copy Files

```
Your-Deployment/
├── Scripts/
│   └── Universal-AppInstaller.ps1        ← Copy here
└── Installers/
    └── Apps/
        └── [your-installers].msi/exe     ← Place installers here
```

Step 2: Add to Configuration

Open `Orchestration-Config.ps1` and add to Phase 4:

```powershell
@{
    TaskID = "APP-010"
    TaskName = "Install 7-Zip"
    ScriptPath = "Scripts\Universal-AppInstaller.ps1"
    Enabled = $true
    Timeout = 600
    RunAs = "SYSTEM"
    RequiresReboot = $false
    AllowRetry = $true
    Critical = $false
    Parameters = @{
        AppName = "7-Zip"
        InstallerFileName = "7z2408-x64.msi"
        InstallerType = "MSI"
        InstallArguments = "/quiet /norestart"
        DetectionMethod = "Registry"
        DetectionPath = "HKLM:\SOFTWARE\7-Zip"
    }
}
```

Step 3: Test

```powershell
# Test the app installer directly
.\Scripts\Universal-AppInstaller.ps1 `
    -AppName "7-Zip" `
    -InstallerFileName "7z2408-x64.msi" `
    -InstallerType "MSI" `
    -InstallArguments "/quiet /norestart" `
    -DetectionMethod "Registry" `
    -DetectionPath "HKLM:\SOFTWARE\7-Zip"

# Or test through orchestration
.\Orchestration-Master.ps1 -Phase Phase4 -DryRun
```

📖 Documentation Overview

For Beginners

Start with → **Universal-AppInstaller-QuickStart.md**

- 5-minute setup guide
- First 5 applications to deploy
- Simple troubleshooting

For Configuration

Reference → **Phase4-Applications-Sample.ps1**

- 20+ working examples
- Copy/paste ready configurations
- Organized by category

For Deep Dive

Study → **Universal-AppInstaller-Documentation.md**

- Complete feature reference
- Detection methods explained

- Silent install switches database
- Advanced troubleshooting

🎨 Use Cases

✅ Perfect For (Use Universal Installer)

Simple Applications:

- 7-Zip, WinRAR, PuTTY
- Notepad++, VS Code
- VLC, Media players
- Adobe Reader
- FileZilla, WinSCP
- Git, TortoiseGit
- Paint.NET, IrfanView
- TreeSize, utilities
- **~80% of your application portfolio**

Characteristics:

- Standard silent install switches
- Simple file or registry detection
- Minimal post-install configuration
- Standalone operation

❌ Not Recommended For (Use Custom Scripts)

Complex Applications:

- Microsoft Office (complex XML configs)
- SQL Server (multi-component)
- Adobe Creative Cloud (licensing)
- Google Chrome (enterprise policies)

- Mozilla Firefox (enterprise policies)
- BitLocker (encryption configuration)
- Custom LOB applications
- **~20% of your application portfolio**

Characteristics:

- Complex pre/post configuration
- Enterprise policy requirements
- Multi-step installation process
- Service configuration needed

🔧 Configuration Template

Quick template for adding new apps:

```powershell
```

```
@{
    TaskID = "APP-0XX"
    TaskName = "Install [App Name]"
    ScriptPath = "Scripts\Universal-AppInstaller.ps1"
    Enabled = $true
    Timeout = 600
    RunAs = "SYSTEM"
    RequiresReboot = $false
    AllowRetry = $true
    Critical = $false
    Description = "Installs [App Name]"
    Parameters = @{
        AppName = "[Application Display Name]"
        InstallerFileName = "[installer.exe/msi]"
        InstallerType = "AUTO"              # AUTO, MSI, EXE, MSIX
        InstallArguments = "[/S or /quiet]"
        DetectionMethod = "[Registry/File]"
        DetectionPath = "[path or registry key]"
        DetectionValue = ""                 # Optional
        RequiredVersion = ""                # Optional
    }
}
```

📊 Exit Codes Reference

Code	Meaning	Action
0	Success - Installed	Continue
1	General failure	Retry/Fail
2	Installer not found	Fix path
3	Detection failed	Fix detection
4	Installation failed	Check logs
5	Validation failed	Verify install
10	Already installed	Continue

🧪 Testing Workflow

1. Manual Test

```powershell
# Test installer with silent switches
.\installer.exe /S
```

2. Script Test

```powershell
# Test Universal Installer directly
.\Universal-AppInstaller.ps1 -AppName "App" -InstallerFileName "app.exe" ...
```

3. Dry Run Test

```powershell
# Test in orchestration without changes
.\Orchestration-Master.ps1 -Phase Phase4 -DryRun
```

4. Phase Test

```powershell
# Test Phase 4 only
.\Orchestration-Master.ps1 -Phase Phase4
```

5. Full Test

```powershell
# Test complete orchestration
.\Orchestration-Master.ps1
```

📁 File Structure

After deployment, your structure should look like:

```
C:\Deploy\
├── Orchestration-Master.ps1
├── Orchestration-Config.ps1
├── Scripts\
│   ├── Universal-AppInstaller.ps1        ← NEW
│   ├── Phase1-Critical\
│   ├── Phase2-Security\
│   └── ... other scripts ...
└── Installers\
    └── Apps\                             ← NEW
        ├── 7z2408-x64.msi
        ├── npp.8.6.9.Installer.x64.exe
        ├── vlc-3.0.21-win64.exe
        └── ... more installers ...

C:\ProgramData\OrchestrationLogs\
├── Orchestration_*.log                   ← Main log
└── Apps\                                 ← NEW
    ├── Install-7-Zip_*.log
    ├── Install-Notepad++_*.log
    └── ... per-app logs ...
```

🔍 Troubleshooting Quick Reference

Installer Not Found

```powershell
# Check file exists
Test-Path "C:\Deploy\Installers\Apps\installer.exe"

# Verify file name matches configuration
```

Installation Fails

```powershell
```

```
# Check app-specific log
Get-Content "C:\ProgramData\OrchestrationLogs\Apps\Install-[App]_*.log"

# Test installer manually
.\installer.exe /S
```

Detection Fails

```
powershell

# Verify detection path exists
Test-Path "C:\Program Files\App\app.exe"
Get-ItemProperty "HKLM:\SOFTWARE\App"

# Try alternative detection method
```

Already Installed Keeps Running

```
powershell

# Check detection is working
# Detection should find app and return exit code 10
# Verify DetectionPath is correct
```

💡 Best Practices

1. Test Before Deploying

Always test installers manually with silent switches first.

2. Use Consistent Naming

```
powershell

# Good: Clear version and architecture
InstallerFileName = "app-1.2.3-x64.msi"

# Bad: Generic name
InstallerFileName = "setup.exe"
```

3. Prefer MSI When Available

MSI installers have standardized switches and better logging.

4. Set Appropriate Timeouts

```powershell
Timeout = 300   # Small utility (5 min)
Timeout = 600   # Standard app (10 min)
Timeout = 1800  # Large app (30 min)
```

5. Enable Retry for Network-Based Installs

```powershell
AllowRetry = $true
Critical = $false  # Don't stop orchestration
```

6. Use Version Checks for Critical Apps

```powershell
RequiredVersion = "3.0.21"  # Ensures minimum version
```

7. Document Custom Switches

Add comments for non-standard switches:

```powershell
InstallArguments = "/S /NOICONS"  # /NOICONS = no desktop shortcut
```

📈 Benefits

Time Savings

- **Before:** 50 custom scripts to maintain
- **After:** 1 universal script + simple config entries
- **Savings:** ~95% reduction in code to maintain

Easier Maintenance

- Bug fix in one place fixes ALL apps
- Add new apps without writing code
- Non-technical staff can add apps

Better Reporting

- Consistent logging format
- Standard exit codes
- Individual app tracking
- Full audit trail

Reliability

- Tested detection methods
- Timeout protection
- Automatic retry logic
- Comprehensive error handling

🎓 Learning Path

Day 1: Quick Start

1. Read QuickStart.md
2. Install first 5 apps
3. Verify in orchestration

Day 2: Expand

1. Add 10 more apps from samples
2. Customize detection methods
3. Test with dry run

Day 3: Master

1. Read full documentation
2. Create custom configurations

3. Implement advanced features

Day 4: Production

1. Test on pilot machines
2. Review logs and optimize
3. Deploy to production

📞 Support Resources

Included Documentation

- **QuickStart.md -** Get started in 5 minutes
- **Documentation.md -** Complete reference guide
- **Sample.ps1 -** 20+ working examples

Finding Silent Switches

- Silent Install HQ
- WPKG Wiki
- Vendor documentation
- Search: "[App Name] silent install switches"

Logging

All logs located in:

```
C:\ProgramData\OrchestrationLogs\Apps\
```

✅ Pre-Deployment Checklist

Before deploying to production:

- [ ] Universal-AppInstaller.ps1 copied to Scripts folder
- [ ] All installer files in Installers\Apps folder
- [ ] Each app tested manually with silent switches

- [ ] Each app tested with Universal Installer script
- [ ] Detection methods verified post-install
- [ ] Phase 4 tested with dry run
- [ ] Phase 4 tested with actual installation
- [ ] Logs reviewed for errors
- [ ] Apps verified in Programs & Features
- [ ] Apps launch and work correctly
- [ ] Configuration documented
- [ ] Pilot group tested successfully

🚀 Ready to Deploy!

You now have everything needed to streamline your application deployments:

1. ✅ **Universal Installer Script -** Handles 80% of apps
2. ✅ **20+ Working Examples -** Copy and customize
3. ✅ **Complete Documentation -** Reference guide
4. ✅ **Quick Start Guide -** 5-minute setup
5. ✅ **Best Practices -** Proven patterns

Next Steps:

1. Read the QuickStart guide
2. Test your first application
3. Add more apps from samples
4. Customize for your environment
5. Deploy to production

📝 Version History

Version 1.0.0 (2024-12-08)

- Initial release
- MSI, EXE, MSIX, APPX support

- Multiple detection methods
- Comprehensive logging
- 20+ sample configurations
- Complete documentation

📄 License

This Universal App Installer is provided for use with your orchestration framework. Customize as needed for your environment.

🙏 Acknowledgments

Designed for enterprise Windows 11 deployments managing 3000+ devices. Built to integrate seamlessly with the existing orchestration framework while dramatically reducing deployment complexity.

Happy Deploying! 🚀

For questions or issues, review the comprehensive documentation or check the application-specific logs in `C:\ProgramData\OrchestrationLogs\Apps\`.