Chocolatey

Jakub Levý
jakub.levy15@gmail.com

NSWI126 2023/2024

Faculty of Mathematics and Physics
Charles University

Overview

- Package management for Windows
- FOSS (Apache License 2.0) + Enterprise (EULA)
- `choco`
- Chocolatey Community (Gallery)
  - https://community.chocolatey.org
- Chocolatey GUI
- AU
- Enterprise
  - Private CDN
  - Sync with Programs and Features
  - Package internalizer
  - Package builder
  - …

Chocolatey X Scoop

- Released in 2011
- Does not hide UAC
- Support for proprietary SW
- Packages contain boilerplate
- Treated as a money source
- Support for Autohotkey

- Released in 2013
- Hides UAC by preferring portable
- Better for niche CLI programs
- Packages are expressive
- Driven by community
- Does not support Autohotkey
- More secure by default

choco

- Command-line C# program
- `choco install`
- `choco upgrade`
- `choco search`
- `choco list`
- `choco pin`
- `choco info`

Chocolatey Community

- https://community.chocolatey.org

- NuGet 2 server
- Not FOSS anymore
- Source code not available anymore
- Single point of failure

Security

- Tested by VirusTotal
- Checked by a moderator
- Is it enough?
  - Trusted packages
- Custom private Nuget server
  - `choco source add –n NAME –s LOCATION`

Chocolatey Installation

- Chocolatey occupies `$env:ChocolateyInstall`
  - By default `C:\ProgramData\chocolatey`

Automating Windows Installations

- Very different from Linux environments
- Tedious and exhausting process
- Many different scenarios
    - Installers
    - GUI portable programs
    - CLI programs
    - Fonts
    - Drivers
    - Services

Installers (1)

- InstallShield wrapping MSI
- Pure InstallShield
- NSIS (Nullsoft Scriptable Install System)
- Inno Setup
- Windows Installer (MSI)
- Custom Ad-Hoc

- Tools for creating silent installers are generally provided by toolkits

Installers (2)

- Worst
  - Proprietary (InstallShield)
  - Installer with many options
  - Installers with non-scriptable options
  - Bugged silent installation, since nobody anticipated silent installation
  - Custom Ad-Hoc (generally no options to install silently)

- Require ✗ not require administrative privilege to install

Portable GUI Programs

- Generally easy
- Delicate problems
  - Create Start Menu icon
  - Create Desktop icon
  - …

CLI

- Generally easy
- Problems
  - Put every bin location to `PATH` ✗ symlink binary to one location in `PATH`
  - Handling versions of one program (e.g. `java`)

Symlinks

- `mklink`
- Terrible implementation in Windows
- Requires Administrative Privileges
- Solution
    - Use Chocolatey executable shimming

Creating a package

- `choco new PACKAGE_ID`
    - Creates a new nuget template with name set to `PACKAGE_ID`
- Use of Powershell 2 is required for compatibility
- Some API functions are provided, their use is preferred to get approvement from moderator
- May contain binary in addition to scripts
- Comprehensive docs
    - https://docs.chocolatey.org/en-us/create/

Package id

- Only lowercase letters
- Use nothing or hyphen as separator
    - hyphen is preferred when the package name is long
- e.g. `adobe-reader` and `adobereader` would both pass

.install and .portable packages

- Submit both installer and portable version of a program
- e.g. create three packages with respective ids
  - `deadbeef.install` is installer version of DeadBeef
  - `deadbeef.portable` is portable version of DeadBeef
  - `deadbeef` is a meta package that has `deadbeef.install` as dependency

Dependencies

- Not really good
- Limited use case
- Windows programs generally bundle everything

nuspec file

Embedded ✗ Recipe-only package

- Preferred by Chocolatey
- No dependency on download link
- Problematic with slow upload speed
- Need to contain `legal/` directory
- Not everytime possible

- Not preferred by Chocolatey
- Dependency on download link
- Packages have a few kilobytes
- Does not redistribute actual SW
- Always legal

legal/ directory

- In case of embedding, `legal/` must contain SW license full text and VERIFICATION.txt file containg checksum and direct download link where binary was obtained
- Not everytime possible, redistribution in binary form must be explicitly allowed by the software license

tools/ directory

- `chocolateybeforemodify.ps1`
    - Called before installing and uninstalling a package (also before upgrading)
- `chocolateyinstall.ps1`
    - Called as an install script
- `chocolateyuninstall.ps1`
    - Called as an uninstall script

- When embeding binary, put them in this directory
    - Preferred locations
        - `tools/x64/program.exe`
        - `tools/x86/program.exe`

chocolateybeforemodify.ps1

- Generally only used to shutdown running applications before
  - Upgrading
  - Uninstalling
  - …
- Some installers can handle running applications themselves
- Some applications should never be automatically killed (VPN)

chocolateyinstall.ps1

- Package installation recipe
- Arguments can be passed
  - Configures the installation
  - `choco install git --params “/NoGuiHereIntegration”`

chocolateyuninstall.ps1

- Package uninstallation recipe
- Arguments can also be passed
  - Configures the uninstallation, e.g. should the configuration be cleared?
  - `choco uninstall netlimiter --params “/CleanSettings”`

- Not required for
  - CLI programs
  - Portable programs
  - Standard not bugged installers which install exactly one application

Pack & Debug

- `choco pack`
- `choco install -s . PACKAGE_ID -d`
- `choco push -s SOURCE`

After push

- 30 minutes wait for CDN refresh
- Validation
- Verification
- Virus scan by VirusTotal

Tips to get your package approved easily

- Check if your package has at most 5 positives on VirusTotal
  - Otherwise no way to get approvement
  - Problematic for programs such as `netcat`
- Using approved packages code is better than reading docs
- Respect package ID guidelines
- Use CDN for package icon
- Use at most powershell 2
- Ask on discord

Deprecating a package

- Create a new version of the deprecated package
- Old package
  - Prepend `[Deprecated]` to the title of the old package
  - Add dependency to a new package
  - Remove `<iconUrl>`
  - Remove all files except for nuspec
  - Unlist previous versions except for the last final deprecated version

Requesting & handing over

- Package must always have at least one maintainer
- Requests
  - https://github.com/chocolatey-community/chocolatey-package-requests
  - RFM (request for maintainer(s))
  - RFP (request for a package)

Automatically update and push packages

- AU (Chocolatey Automatic Package Updater Module)
- https://github.com/chocolatey-community/chocolatey-au
- Powershell module
- Compatible with Powershell Core
- Runs on Linux server
- Initially developed by a mathematician
    - Code quality
    - Documentation
- Nowdays maintained by Chocolatey community

AutoHotkey

- Last resort
- Ridiculous or bugged installers
- Error-prone
- Tedious task

AutoHotkey in Practice

- Silent installer shows usually one confirmation dialog
- Silent installation is bugged
- No option for silent installation
- Watch out for language variations of the installer

Problems & inconsistencies (1)

- Desktop shortcuts
- Start menu icons
- Different package parameters for same functionality
- Missing package parameters
- Nuget 2
- Dotnet core still not supported
- Insanely long moderation time
- Unmaintained packages
- Difficult handing packages over

Problems & inconsistencies (2)

- No button to notify that a package is obsolete
- Difficult to fix a bug in foreign package
- Around 200MB limit for package size
- Community has no real impact
- No way of becoming a reviewer/moderator?

Sources

- https://docs.chocolatey.org/en-us/
- Personal experience as a maintainer of 97 packages with 230k downloads
    - My Profile