



Packaging and Scripting

for mac admins

Packaging and Scripting

For Mac Admins

Adam Wickert, July 18 2023, Penn State MacAdmins 2023

Agenda

1. Scripting
2. Q&A
3. Work on ideas
4. Break
5. Packaging
6. Q&A
7. Work on ideas



Scripting

why learn scripting?

- create or modify files
- set permissions
- set preferences and settings
- create logic to automate
- extend capabilities of existing tools
- career growth and experience



what is a script?

Executable files in a scripting language

- Scripting languages are interpreted at runtime
 - bash
 - zsh
 - javascript
 - python
 - AppleScript

basic script structure

□ □ ...



⊗ 0 △ 0

Ln 1, Col 1

Tab Size: 4

UTF-8

LF

Shell Script

logic and conditionals

- check for a condition
- apply only what's needed
- allow multiple uses



\$ rosettaInstall.sh ×



\$ rosettaInstall.sh

```
1  #!/bin/bash
2
3  arch=$(/usr/bin/arch)
4
5  if [ "$arch" == "arm64" ]; then
6      /usr/sbin/softwareupdate --install-rosetta --agree-to-license
7  fi
```



⊗ 0 ⚠ 0

Ln 1, Col 1

Tab Size: 4

UTF-8

LF

Shell Script





arguments & options

- pass information to the script
- allow multiple uses
- add to the logic

zparseopts • Untitled-1 — Scripts

! # zparseopts Untitled-1

1 # zparseopts

2 #

3 # Features:

4 # - supports short and long flags (ie: -v|--verbose)

5 # - supports short and long key/value options (ie: -f <file> | --filename <file>)

6 # - does NOT support short and long key/value options with equals assignment (ie: -f=<file> | --filename=<file>)

7 # - supports short option chaining (ie: -vh)

8 # - everything after -- is positional even if it looks like an option (ie: -f)

9 # - once we hit an arg that isn't an option flag, everything after that is considered positional

10 function zparseopts_demo() {

11 local flag_help flag_verbose

12 local arg_filename=(myfile) # set a default

13 local usage=(

14 "zparseopts_demo [-h|--help]"

15 "zparseopts_demo [-v|--verbose] [-f|--filename=<file>] [<message...>]"

16)

17

18 # -D pulls parsed flags out of \$@

19 # -E allows flags/args and positionals to be mixed, which we don't want in this example

20 # -F says fail if we find a flag that wasn't defined

21 # -M allows us to map option aliases (ie: h=flag_help -help=h)

22 # -K allows us to set default values without zparseopts overwriting them

23 # Remember that the first dash is automatically handled, so long options are -opt, not --opt

24 zmodload zsh/zutil

25 zparseopts -D -F -K -- \

26 {h,-help}=flag_help \

27 {v,-verbose}=flag_verbose \

28 {f,-filename}:=arg_filename ||

29 return 1

30

31 [[-z "\$flag_help"]] || { print -l \$usage && return }

32 if ((\$#flag_verbose)); then

33 print "verbose mode"

34 fi

35

36 echo "--verbose: \$flag_verbose"

37 echo "--filename: \$arg_filename[-1]"

38 echo "positional: \$@"

39 }

Ln 10, Col 29 Spaces: 2 UTF-8 LF YAML

man man


```
adam — less ◀ man man — 92×32
MAN(1)                                General Commands Manual                                MAN(1)

NAME
    man, apropos, whatis - display online manual documentation pages

SYNOPSIS
    man [-adho] [-t | -w] [-M manpath] [-P pager] [-S mansect]
        [-m arch[:machine]] [-p [eprtv]] [mansect] page ...

    man -f [-d] [-M manpath] [-P pager] [-S mansect] keyword ...
    whatis [-d] [-s mansect] keyword ...

    man -k [-d] [-M manpath] [-P pager] [-S mansect] keyword ...
    apropos [-d] [-s mansect] keyword ...

DESCRIPTION
    The man utility finds and displays online manual documentation pages. If
    mansect is provided, man restricts the search to the specific section of
    the manual.

    The sections of the manual are:
        1. General Commands Manual
        2. System Calls Manual
        3. Library Functions Manual
        4. Kernel Interfaces Manual
        5. File Formats Manual
        6. Games Manual
        7. Miscellaneous Information Manual
        8. System Manager's Manual
        9. Kernel Developer's Manual

:
```



```
[adam@FutureMBAsOnly ~ % pkgbuild --help  
pkgbuild: unrecognized option '--help'  
Usage: pkgbuild [options] --root <root-path> [--component-plist <plist-path>] <package-output-path>  
       Build a package from an xcodebuild destination root  
  
Usage: pkgbuild --analyze --root <root-path> <plist-output-path>  
       Create template component plist from an xcodebuild destination root  
  
Usage: pkgbuild [options] {--component <component-path>} <package-output-path>  
       Build a package from one or more previously-built bundles  
  
See pkgbuild(1) for details.  
  
adam@FutureMBAsOnly ~ %
```




code editors

- Free and paid options

- BBEdit

- VScode

- CodeRunner

- Sublime Text

- Benefits of a dedicated Code Editor
 - Terminal for testing code
 - Version control
 - Context highlight color scheme
 - Autofill/suggest commands



Visual Studio interface showing the Extensions view and a code editor.

EXTENSIONS: MARK...

@sort:installs

- Python** (42.7M, 4 stars) - IntelliSense (Pylance), Lint... - Microsoft - [Install](#)
- C/C++** (23.5M, 3.5 stars) - C/C++ IntelliSense, debugg... - Microsoft - [Install](#)
- Jupyter** (22.7M, 2.5 stars) - Jupyter notebook support, ... - Microsoft - [Install](#)
- ESLint** (16.5M, 4.5 stars) - Integrates ESLint JavaScrip... - Dirk Baeumer - [Install](#)
- Prettier - ...** (15.3M, 3.5 stars) - Code formatter using prettier - Prettier - [Install](#)
- Pylance** (15.3M, 3.5 stars) - A performant, feature-rich l... - Microsoft - [Install](#)
- Live Server** (15M, 4.5 stars) - Launch a development loca... - Ritwick Dev - [Install](#)

Code Editor:

src > JS serviceWorker.js > registerValidSW > then() callback > 6

```
57 function registerValidSW(swUrl, config) {
58   navigator.serviceWorker
59     .register(swUrl)
60     .then(registration => {
61       registration.onupdatefound = () => {
62         const installingWorker = registration.installing
63         if (installingWorker == null) {
64           return;
65         }
66         installingWorker.onstatechange = () => {
67           if (installingWorker.state === 'installed'
68             if (navigator.serviceWorker.controller)
69             sendBeacon (method) Navigator.sendBeacon(url
70             serviceWorker
71             setInterval Set Inte
72             setTimeout Set Tim
73             share
74             abc state
75             mediaSession
76             storage
77             abc checkValidServiceWorker
78             abc onSuccess
79             import statement Import ext
80             requestMediaKeySystemAccess
81             } else {
```


functions

don't repeat yourself

- Don't repeat yourself
- If you need to do it twice, write a function

- Labeled image of a basic function

The background of the image consists of several overlapping, wavy, horizontal bands of dark blue and navy blue, creating a layered, mountain-like or oceanic effect. The text "fail states" is centered horizontally and vertically in a clean, white, sans-serif font.

fail states

- If data is missing, prompt
- Log an error

\$ APladdtoStaticGroup.sh ×

\$ APladdtoStaticGroup.sh

```
1  #!/bin/bash
2  # 2019-08-07 awickert
3  # Add a computer to a static group
4
5  serialNumber="$(ioreg -rd1 -c IOPlatformExpertDevice | awk -F'"' '/IOPlatformSerialNumber/{print $4}')"
6  apiUser="$4"
7  if [[ -z $apiUser ]]; then
8      read -p "Username:" apiUser
9  fi
10 apiPass="$5"
11 if [[ -z $apiPass ]]; then
12     read -sp "Password:" apiPass
13 fi
14 jssHost="$6"
15 if [[ -z $jssHost ]]; then
16     read -p "JSS Host Address:" jssHost
17 fi
18 groupId="$7"
19 if [[ -z $groupId ]]; then
20     read -p "Group ID:" groupId
21 fi
22
23 apiData="<computer_group><computer_additions><computer><serial_number>${serialNumber}</serial_number></computer></computer_additions></computer_group>"
24 curl \
25     -s \
26     -f \
27     -u ${apiUser}:${apiPass} \
28     -X PUT \
29     -H "Content-Type: text/xml" \
30     -d "<?xml version='1.0' encoding='UTF-8' standalone='no'?'>${apiData}" ${jssHost}/JSSResource/computergroups/id/${groupId}
```


 \$ postinstallCisco.sh ×

\$ postinstallCisco.sh

```
1  #!/bin/bash
2  # 2019-11-5 awickert
3  # Postinstall script for Cisco AnyConnect
4
5  INSTALL_DIR=$(dirname $0)
6  ERROR=0
7  PKG="${INSTALL_DIR}/AnyConnect.pkg"
8  CHOICES="${INSTALL_DIR}/choicesForAnyConnect.xml"
9
10 if [[ -f $PKG ]]; then
11
12     /usr/sbin/installer -applyChoiceChangesXML "$CHOICES" -pkg "$PKG" -target $3
13
14     if [[ $? -ne 0 ]]; then
15         /usr/bin/logger -t "${0##*/}" "ERROR! Installation of package failed"
16         ERROR=1
17     fi
18 else
19     /usr/bin/logger -t "${0##*/}" "ERROR! Package $PKG not found"
20     ERROR=1
21 fi
22
23 exit $ERROR
```


exit 0

or why we need to handle errors better

- exit 0 just says its all good
- bypasses any errors that may have occurred
- doesn't help logging or troubleshooting

 \$ postinstallCisco.sh × ...

\$ postinstallCisco.sh

```
1  #!/bin/bash
2  # 2019-11-5 awickert
3  # Postinstall script for Cisco AnyConnect
4
5  INSTALL_DIR=$(dirname $0)
6  ERROR=0
7  PKG="${INSTALL_DIR}/AnyConnect.pkg"
8  CHOICES="${INSTALL_DIR}/choicesForAnyConnect.xml"
9
10 if [[ -f $PKG ]]; then
11
12     /usr/sbin/installer -applyChoiceChangesXML "$CHOICES" -pkg "$PKG" -target $3
13
14     if [[ $? -ne 0 ]]; then
15         /usr/bin/logger -t "${0##*/}" "ERROR! Installation of package failed"
16         ERROR=1
17     fi
18 else
19     /usr/bin/logger -t "${0##*/}" "ERROR! Package $PKG not found"
20     ERROR=1
21 fi
22
23 exit $ERROR
```


The background features a series of dark blue, wavy, layered shapes that resemble a stylized mountain range or ocean waves. The word "testing" is centered in a white, lowercase, sans-serif font.

testing

- Dryrun

The background features a series of overlapping, wavy, dark blue shapes that resemble a stylized landscape or water. The word "comments" is centered in a white, sans-serif font.

comments

- explain your code in your code
- variables with logical names
- comments to explain more detail

11





usage

- LaunchAgents/LaunchDaemons
- Jamf EA
- Postinstall Script



real world examples

- Open source example scripts



version control

- git
 - GitHub or GitLab
 - Self hosted
- subversion
- mercurial

chatGPT

its a tool, not the answer

- It doesn't know everything
- What it does know it might not be able to separate (different OSes, languages)
- It may apply logic from one command to another
- Its a tool that may help, but like stack overflow its not always the answer



practice

- Make a template with a shebang and authorship comment
- Assign a variable to the logged in user
- Declare an array

A photograph showing a collection of brown cardboard boxes of different sizes and orientations, stacked and scattered on a light-colored floor. The boxes are made of corrugated cardboard, with visible flaps and seams. The word "Packaging" is written in a large, white, sans-serif font across the middle-right portion of the image, overlaid on one of the boxes.

Packaging

why learn packaging?

- not available from the vendor as a package file
- add customizations
- the vendor does something weird
 - ...assume a user is logged in
 - ...assume only one user is on the system
 - ...try to write where it doesn't have permission
 - install a local web server to look for updates even after you uninstall



what is a package

- A package is a special folder displayed as a single file
- Installer packages

commandments of packaging

Commandments of Packaging

for macOS (via [afp548.com](https://www.afp548.com))

1. Do not assume that your package will be installed interactively via the GUI or on the currently booted volume.
2. Unnecessary actions are unnecessary.
3. Licensing should have the option to be managed by Systems Administrators.
4. Use pre/post-install scripts only when necessary (and heed all other rules with your scripts).
5. Be true to the Operating System.
6. Naming Conventions are Necessary and Helpful.



folder structure

- Expanded pkg file image



distribution vs flat

flat

- Contains a list of files and locations to place them (payload)
- Can have pre and post install scripts
 - Can also include files to manipulate with scripts
-

distribution

- Can contain multiple packages
- Can modify the Installer app display
- Cannot have their own scripts
- Can require applications to quit
- Can require specific OS version or applications install



package update behavior

- Files may be updated
- Permissions may be reverted
- Files in the previous version and not in the new one may be removed

inspecting packages

- Right click and expand (show package contents)
- Suspicious package

Back/Forward

munkitools-6.3.0.4...

Search

Search

Exports

Review

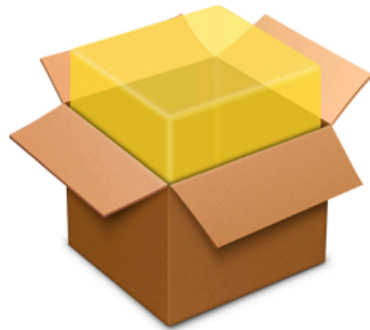
Package Info

All Files

All Scripts


Review

Receipts



munkitools-6.3.0.4574-6.3.0.4556.pkg

PRODUCT ARCHIVE ?

Previously installed on  Macintosh HD using macOS Installer — April 19, 2023 ?

→

Installs 6,543 items — 162.3 MB on disk

→

Not signed

Runs 7 install scripts

→

Executable items have mixed support for Apple Silicon and Intel

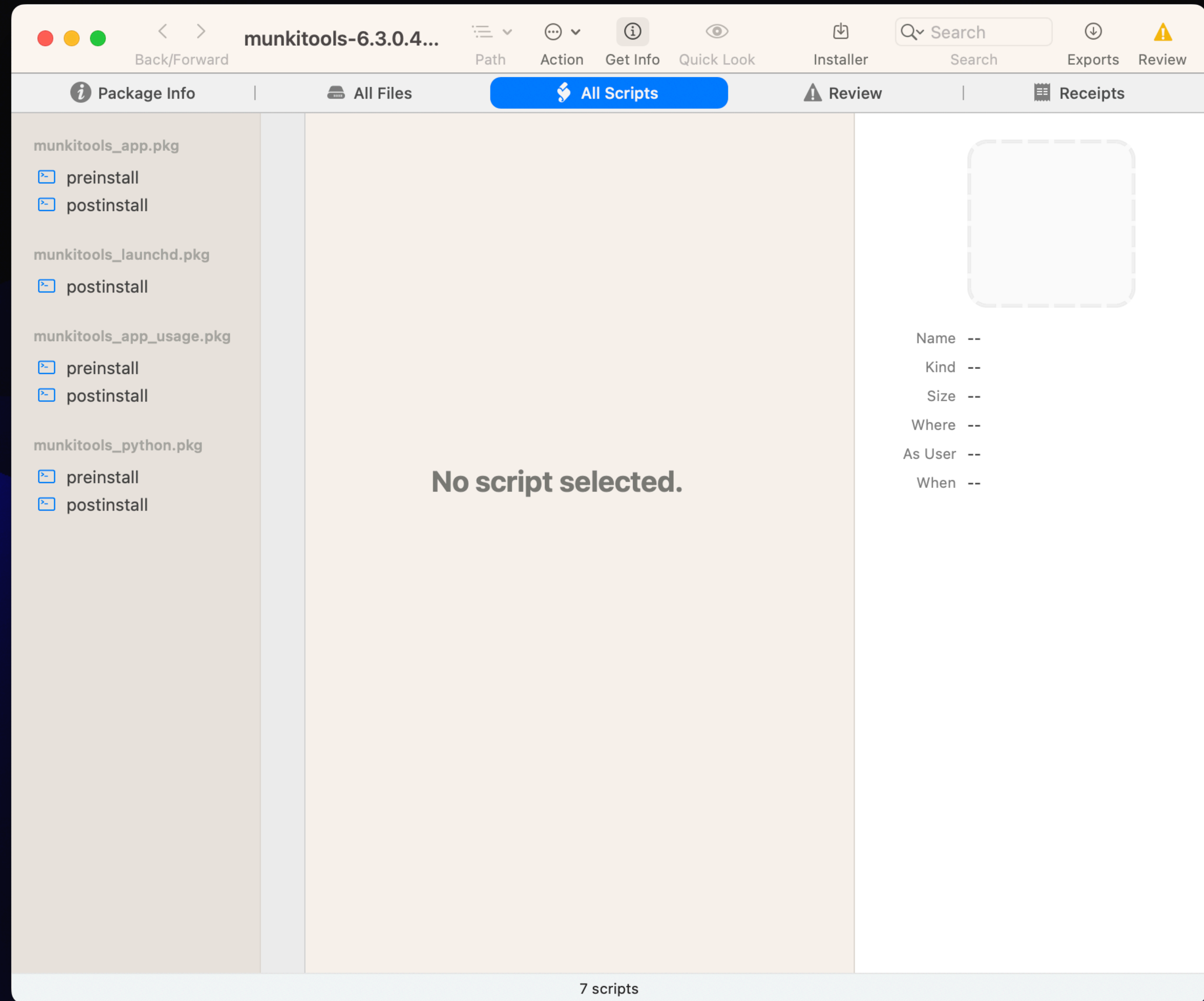
→

89 entitled executable items — 6 entitlements requested

→

Found one warning for review ?

→



Back/Forward

munkitools-6.3.0.4...

PathActionGet InfoQuick LookInstallerSearchSearchExportsReview

i

Package Info

|

All Files

|

All Scripts

|

Review

|

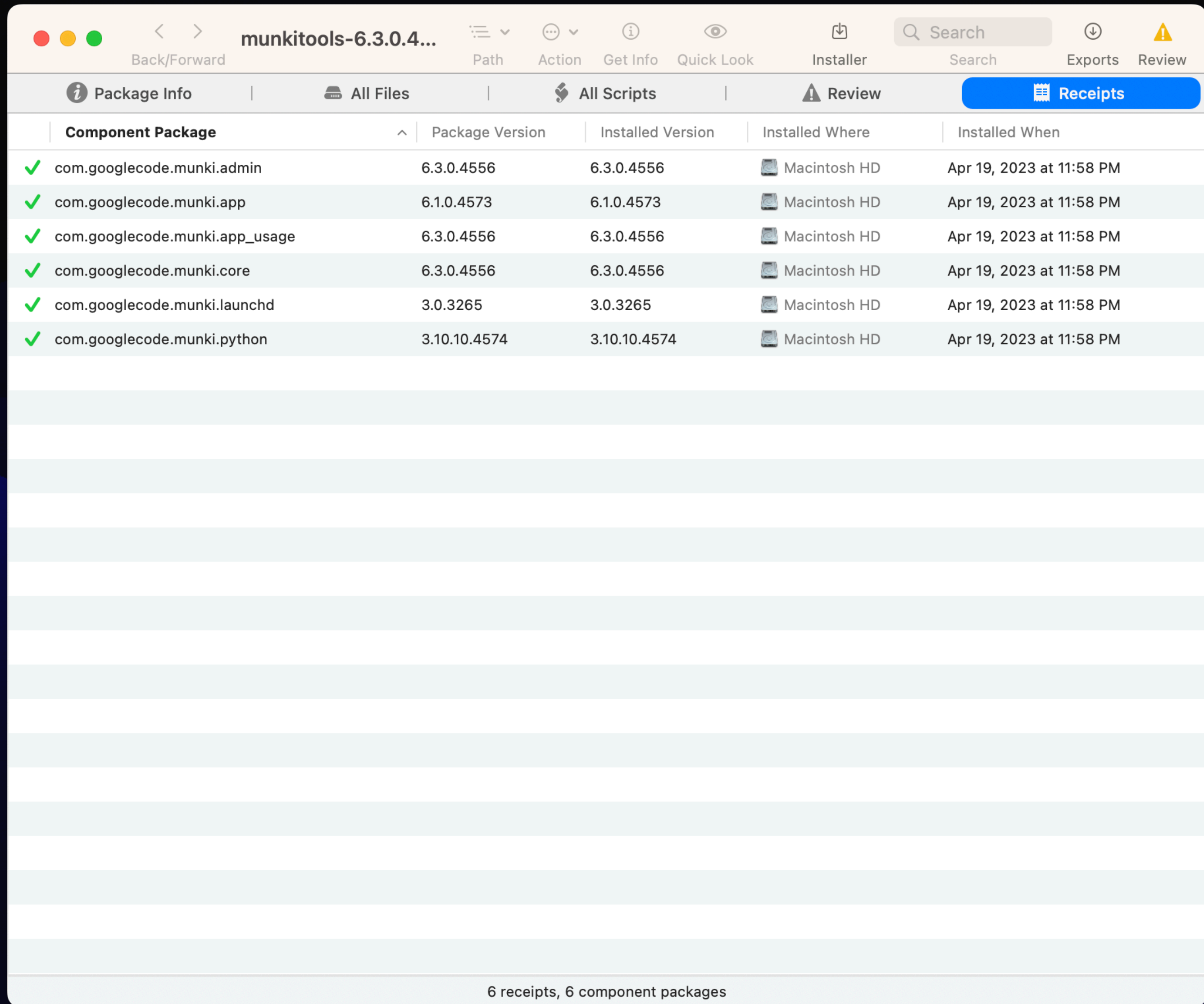
Receipts

One warning

The package isn't signed. It may not install what it claims, or it may have been tampered with since being created.

?

Show Signature Info



using script resources

- Scripts folder
- Preinstall postinstall
- Installers
- choice changes xml
- a binary to use one time



repackaging

- Run a pkg installer using choicechangesXML
- A script that needs a file (image, binary, etc)
- Configuration added
- Multiple installs at once

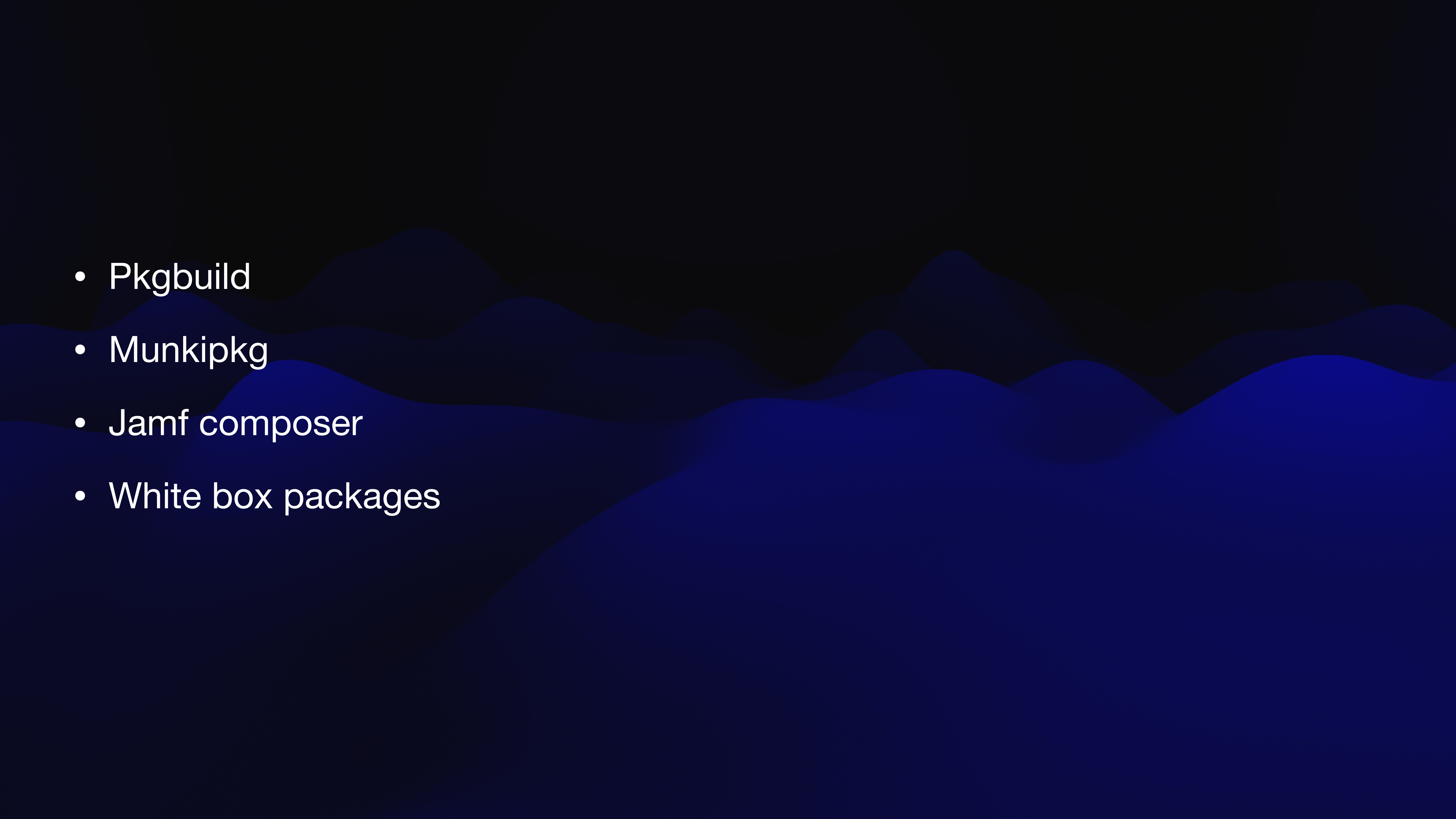


scripts and defaults

- \$0 script path
- \$1 package path
- \$2 target location
- \$3 target volume



tools

- 
- Pkgbuild
 - Munkipkg
 - Jamf composer
 - White box packages

test

ProjectSettingsPayloadScriptsComments

Build

Name:

test

Path:

R

build

⋮

Reference Folder:

Project Folder

Exclusions:

•

Pattern

Kind

RegEx

✓

Remove .DS_Store files

✓

Remove .pbdevelopment files

✓

Remove SCM metadata

✓

Optimize nib files

✓

Remove Resources Disabled folders

+

−

☐

Exclude items only from the payload

test

Project

Settings

Payload

Scripts

Comments

Tag

Identifier:

com.mygreatcompany.pkg.test

Version:

1.0

Post-installation Behavior

On Success:

Do Nothing

Options

☒ Require admin password for installation

☐ Relocatable

☐ Overwrite directory permissions

☐ Follow symbolic links

test

Project

Settings

Payload

Scripts

Comments

Settings

Default Destination: /

Set

Contents

Type: Internal

Name	Owner	Group	Permissions
▼ /	root	wheel	drwxr-xr-x
> Applications	root	admin	drwxrwxr-x
▼ Library	root	wheel	drwxr-xr-x
> Application Support	root	admin	drwxr-xr-x
> Automator	root	wheel	drwxr-xr-x
> Documentation	root	wheel	drwxr-xr-x
> Extensions	root	wheel	drwxr-xr-x
> Filesystems	root	wheel	drwxr-xr-x
> Frameworks	root	wheel	drwxr-xr-x
> Input Methods	root	wheel	drwxr-xr-x
> Internet Plug-Ins	root	wheel	drwxr-xr-x
> Keyboard Layouts	root	wheel	drwxr-xr-x
> LaunchAgents	root	wheel	drwxr-xr-x
> LaunchDaemons	root	wheel	drwxr-xr-x
> PreferencePanels	root	wheel	drwxr-xr-x
> Preferences	root	wheel	drwxr-xr-x
> Printers	root	admin	drwxr-xr-x

+

-

Empty Selection

test

Project

Settings

Payload

Scripts

Comments

Pre-installation

No Scripts Set
Modified: -
A ▴ Choose...

Post-installation

No Scripts Set
Modified: -
A ▴ Choose...

Additional Resources

Name

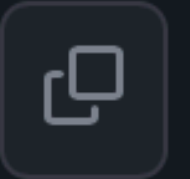
+ -

These resources can be used by the pre and post-installation scripts.

Empty Selection

Package project directory layout

```
project_dir/  
  build-info.plist  
  payload/  
  scripts/
```



Resources

Books

- [macOS Terminal and Shell](#)
- [Moving to zsh](#)
- [Packaging for Apple Administrators](#)
- [Apple Device Management: A Unified Theory of Managing Macs, iPads, iPhones, and Apple TVs](#)

Websites

- afp548.com
- devhints.io
- scriptingosx.com
- derflounder.wordpress.com
- modtitan.com
- And many more!

Conference sessions

- PowerShell on macOS - Wednesday 1:30 pm
- Scripting the unscriptable - GUI Scripts in macOS - Thursday 1:30 pm
- The Journey from No Code, to Low-Code, to Code - Friday 10:45 am

Videos

- [JNUC YouTube/Apple TV App](#)
- [PSU MacAdmins YouTube](#)

<https://bit.ly/psumac2023-106A>

<https://bit.ly/psumac2023-107>