a simple tool for building packages in a consistent, repeatable manner from source files and scripts in a project directory

- does not require Munki
- can be run completely standalone

https://github.com/munki/munki-pkg

● ○ ● Firefox				
Name	^	Date Modified	Size	Kind
▼ <u>build</u>		Today, 2:57 PM		Folder
build-info.plist		Today, 2:57 PM	591 bytes	Property list
▼ payload		Today, 2:57 PM		Folder
Applications		Today, 2:58 PM		Folder
Firefox		Jun 8, 2016, 10:02 AM	206 MB	Application
▼ scripts		Today, 2:57 PM		Folder

```
build-info.plist
                      <?xml version="1.0" encoding="UTF-8"?>¬
                      <!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.app]</pre>
                   3 o <pli>st version="1.0">--
                      <dict>¬
                                                    build-info.json
                            {−
                                  "postinstall_action": "none",¬
Name
                                  "suppress_bundle_relocation": true, -
                   9
    build
                  10
                                 "name": "Firefox-${version}.pkg",¬
  build-info.plist
                  11
    payload
                                 "distribution_style": false,¬
                  12
  Applications
                  13
                        6
                                 "install_location": "/",¬
        Firefox
                  14
                                 "version": "47.0",¬
    scripts
                  15
                                  "ownership": "recommended", ¬
                        8
                  16
                  17
                                  "identifier": "com.github.munki.pkg.Firefox"-
                  18
                       10
                  19
                       Line: 10 Column: 2
                                                       20
                                         Plain Text
                  21 C </dict>-
                  22 v </plist>¬
                  23

    Soft Tabs: 4    name
                                 Property List
                 Line: 12 Column: 31
```

Creating munki-pkg projects



Acceptable Use Policy

By using this equipment, you agree to the following terms and conditions:

1.0 Purpose

The computing resources at Foo Bar University support the educational, instructional, research, and administrative activities of the University and the use of these resources is a privilege that is extended to members of the Foo Bar community. As a user of these services and facilities, you have access to valuable University resources, to sensitive data, and to internal and external networks. Consequently, it is important for you to behave in a responsible, ethical, and legal manner.

In general, acceptable use means respecting the rights of other computer users, the integrity of the physical facilities and all pertinent license and contractual agreements. If an individual is found to be in violation of the Acceptable Use Policy, the University will take disciplinary action, including the restriction and possible loss of network privileges. A serious violation could result in more serious consequences, up to and including suspension or termination from the University. Individuals are also subject to federal, state and local laws governing many interactions that occur on the Internet. These policies and laws are subject to change as state and federal laws develop and change. This document establishes specific requirements for the use of all computing and network resources at Foo Bar University.

2.0 Scope

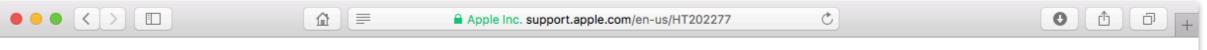
This policy applies to all users of computing resources owned or managed by Foo Bar University. Individuals covered by the policy include (but are not limited to) Foo Bar faculty and visiting faculty, staff, students, alumni, guests or agents of the administration, external individuals and organizations accessing network services via Foo Bar's computing facilities.

Computing resources include all university owned, licensed, or managed hardware and software, and use of the university network via a physical or wireless connection, regardless of the ownership of the computer or device connected to the network.

These policies apply to technology administered in individual departments, the resources administered by central administrative departments (such as the University Libraries and Computing and Information Services), personally owned computers and devices connected by wire or wireless to the campus network, and to off-campus computers that connect remotely to the University's network services.

2.1 Your Rights and Responsibilities

As a member of the University community, the university provides you with the use of scholarly and/or work-related tools, including access to the Library, to certain computer systems, servers, software and databases, to the campus telephone and voice mail systems, and to the Internet. You have a reasonable expectation of unobstructed use of these tools, of certain degrees of privacy (which may vary depending on whether you are a University employee or a matriculated student), and of protection from abuse and intrusion by others sharing these



About policy banners in OS X

Learn more about creating policy banners and fixing issues with display and position.



A policy banner is a banner that you can display at the login window that requires a user to acknowledge it before proceeding.

Create a banner

You can set a login message that appears at the login screen of your Mac. You can also set a "policy" banner to display a longer message that you must accept before you can log in. This can be useful in situations where you need users to agree to or acknowledge terms or conditions before using the computer, such as an Acceptable Use Policy.

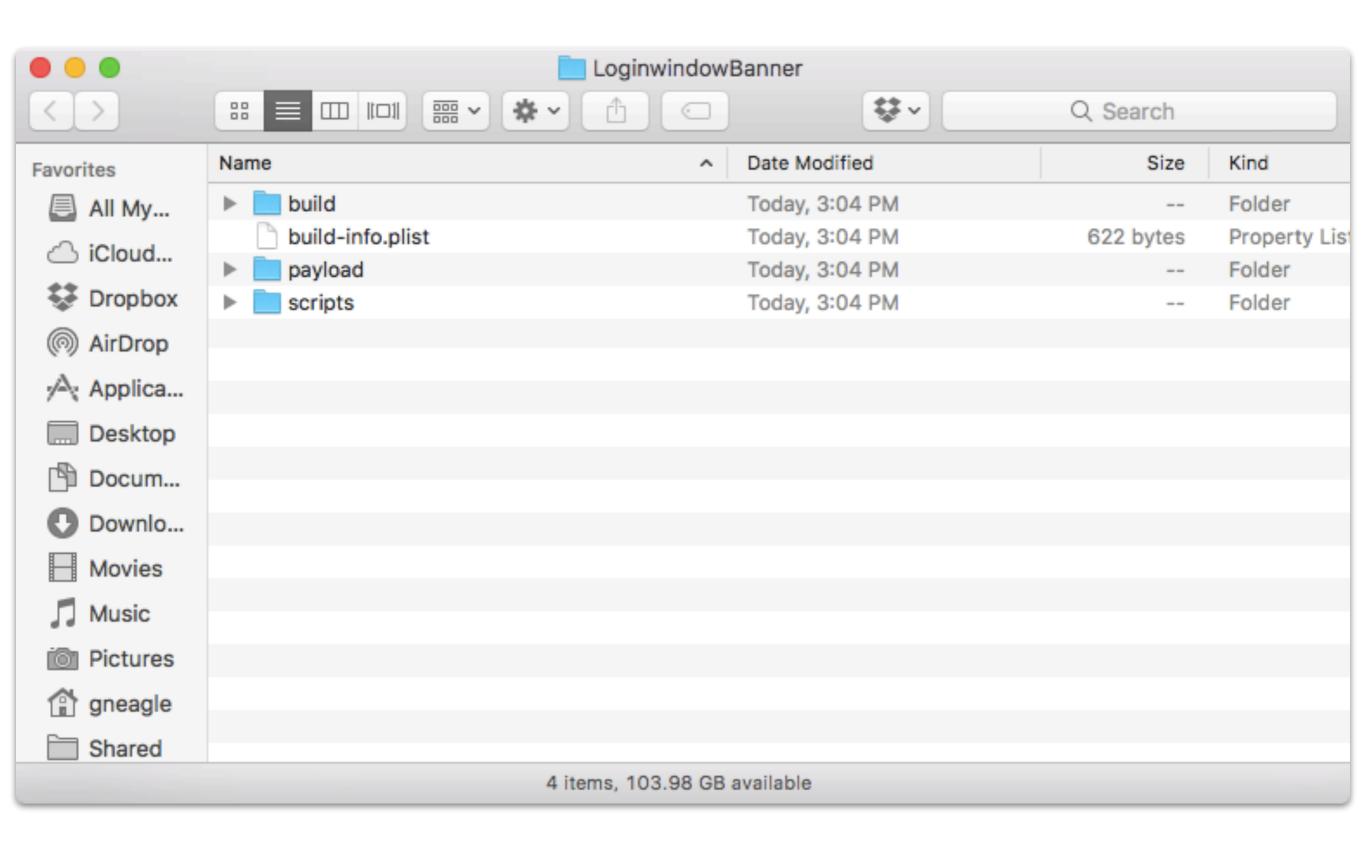
Use these steps to create a policy banner:

- 1. Create a plain text (.txt) or rich text (.rtf) document named PolicyBanner that contains your banner.
- 2. Copy the PolicyBanner file to the /Library/Security/ folder.

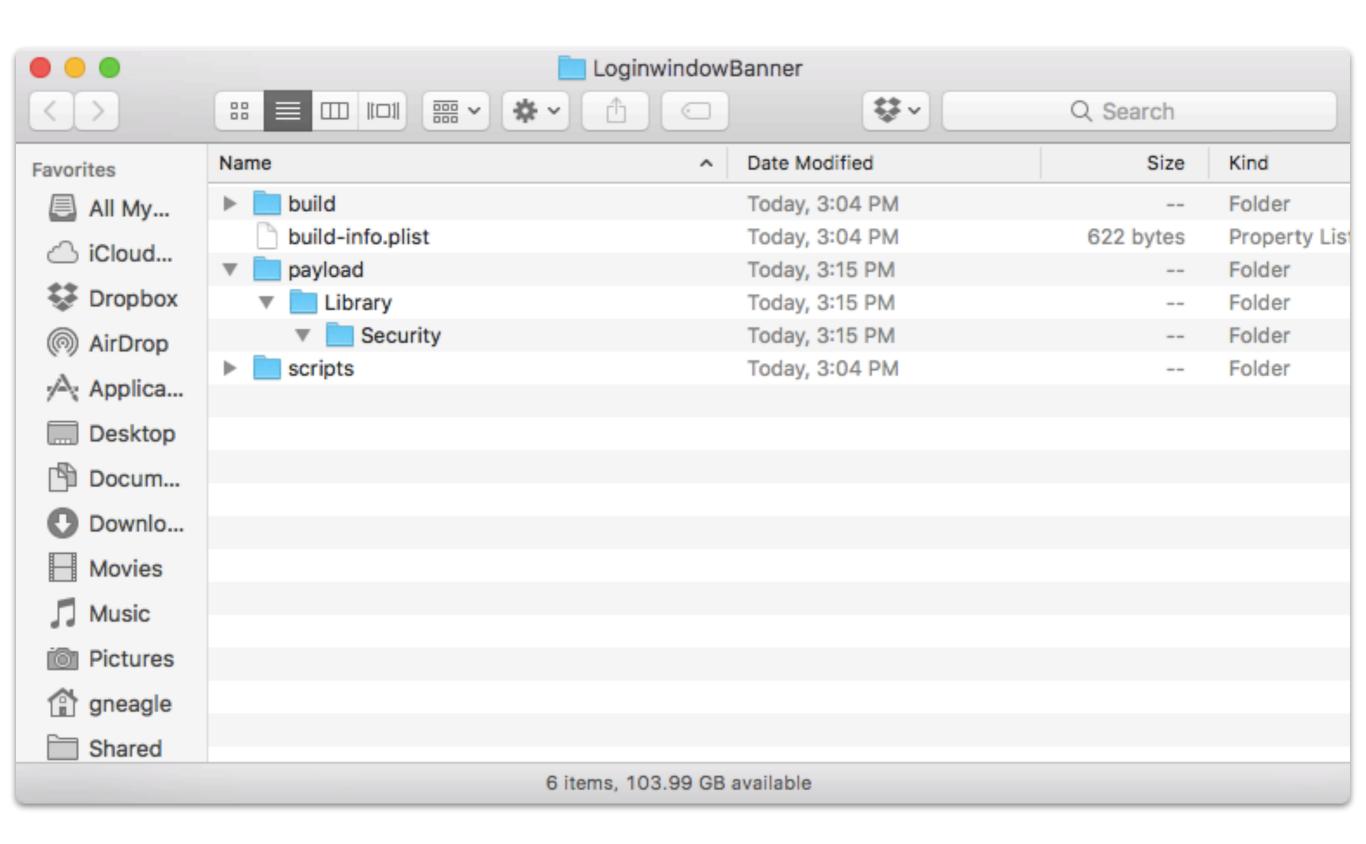
The next time you restart the computer, the banner you created appears when you would normally see the login screen.

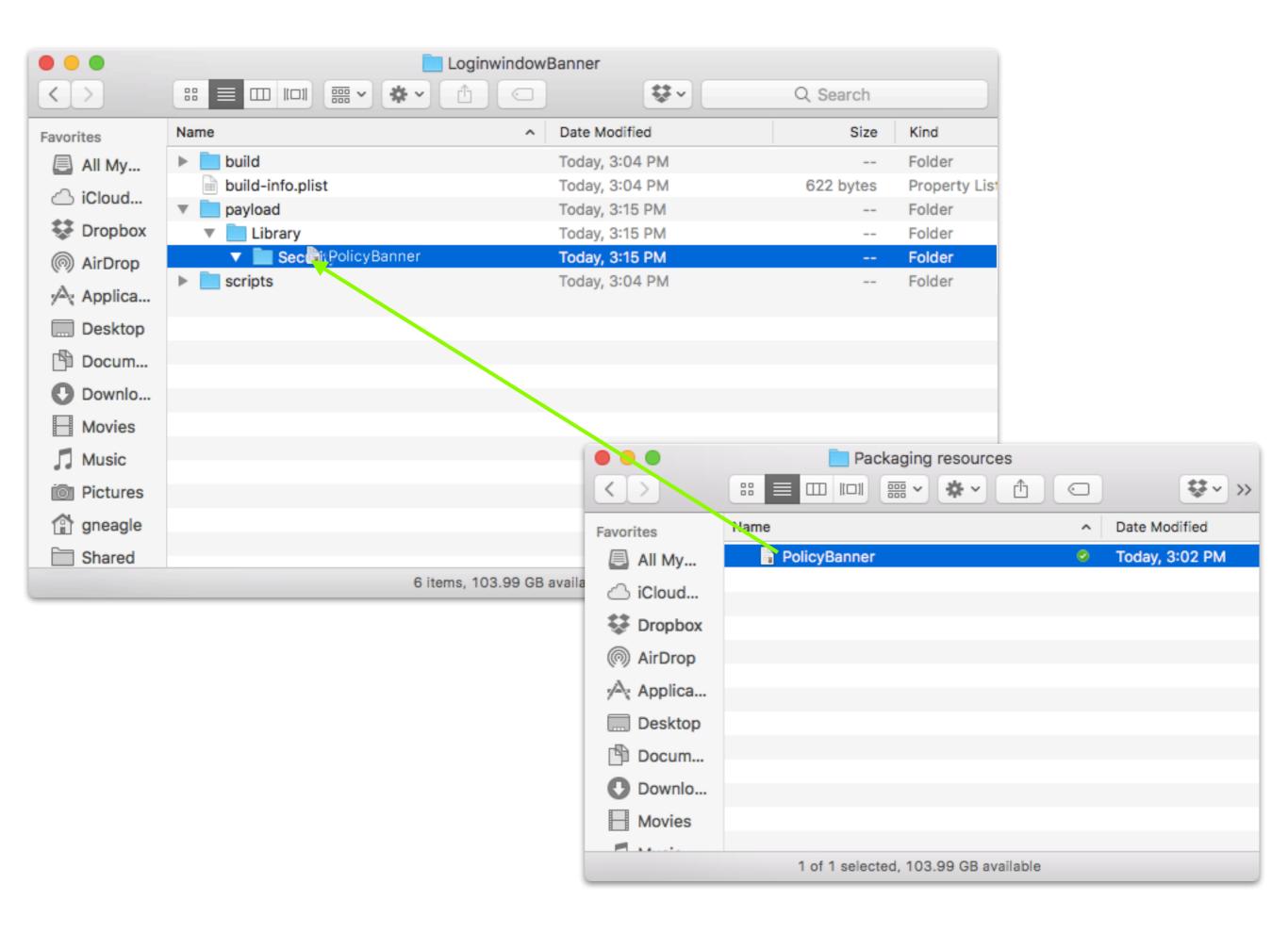
\$./munkipkg --create LoginwindowBanner
munkipkg: Created new package project at LoginwindowBanner

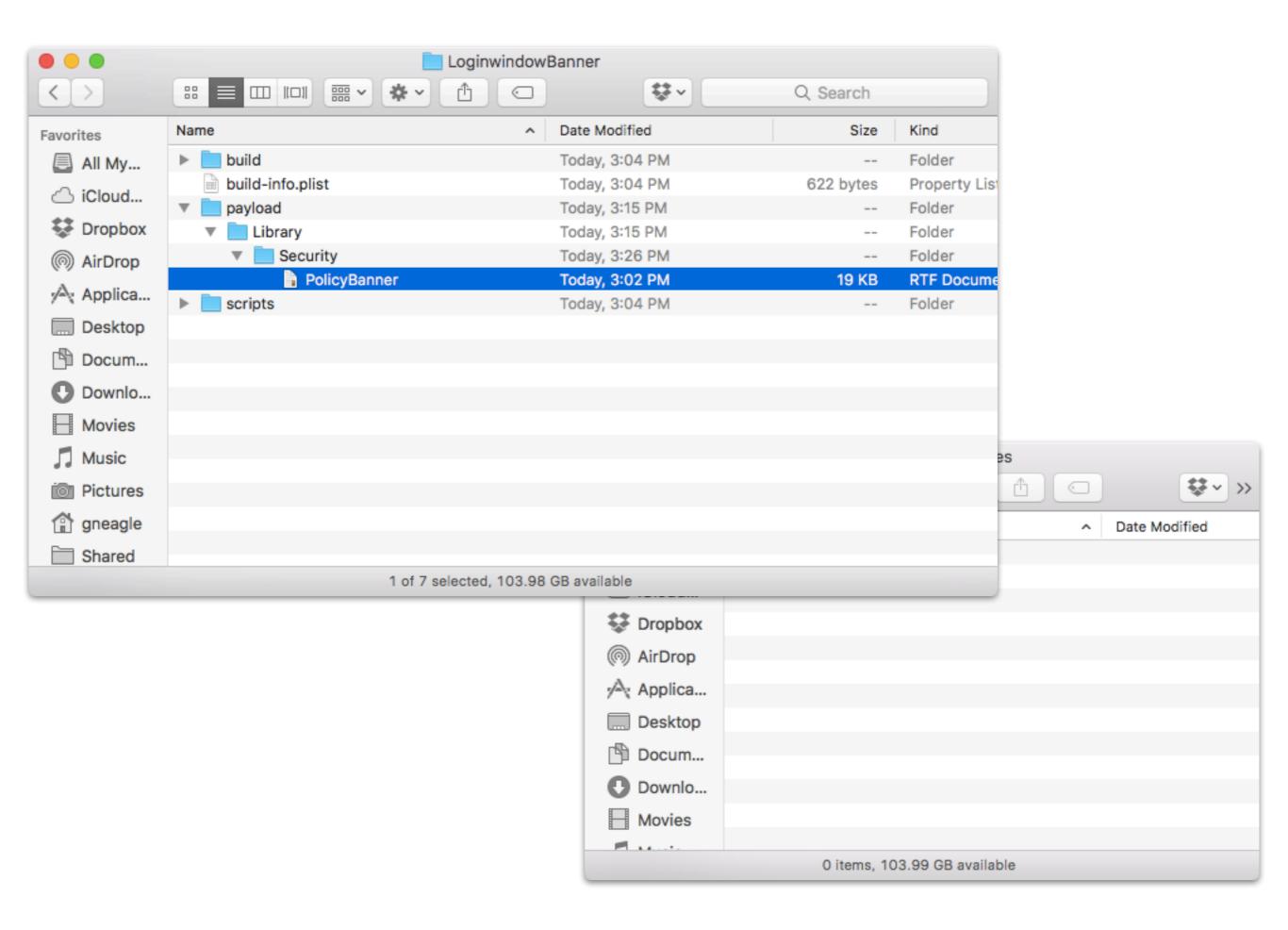
```
$ ls LoginwindowBanner/
build build-info.plist payload scripts
$ open LoginwindowBanner/
```

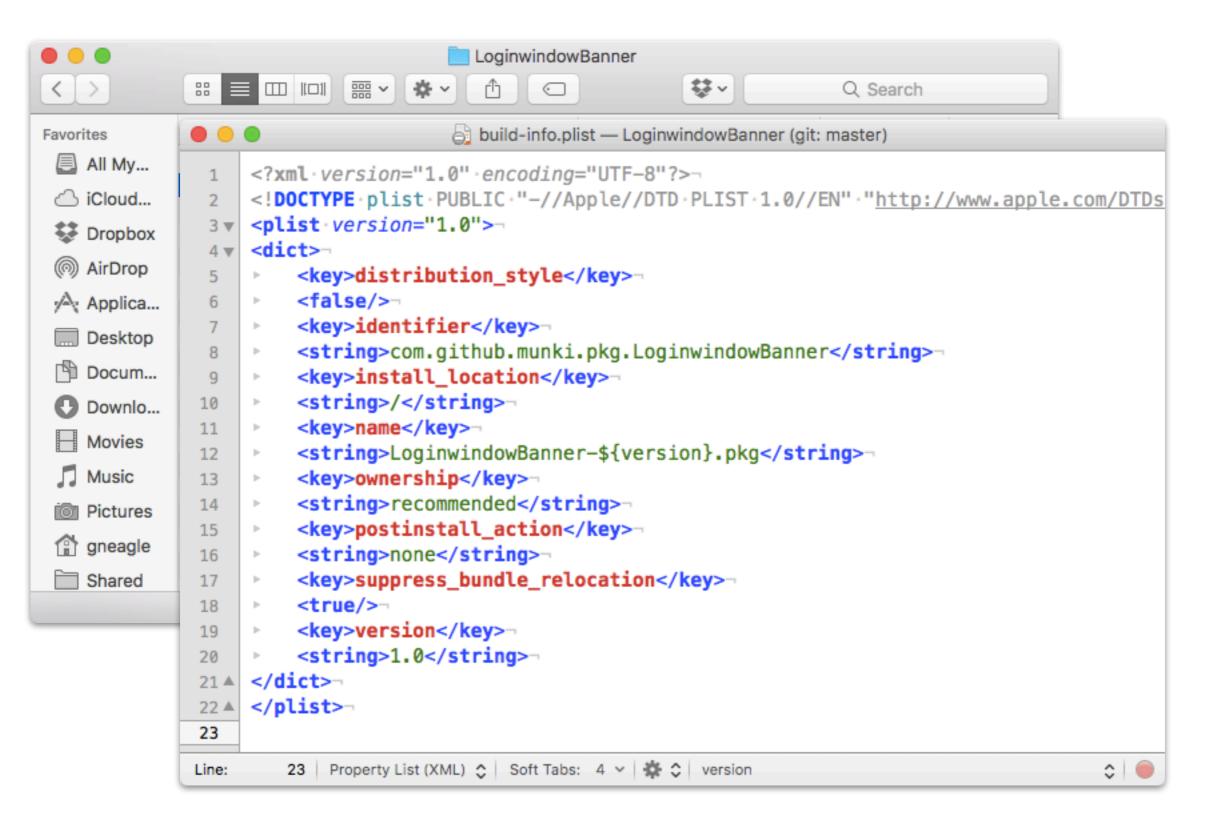


\$ mkdir -p LoginwindowBanner/payload/Library/Security

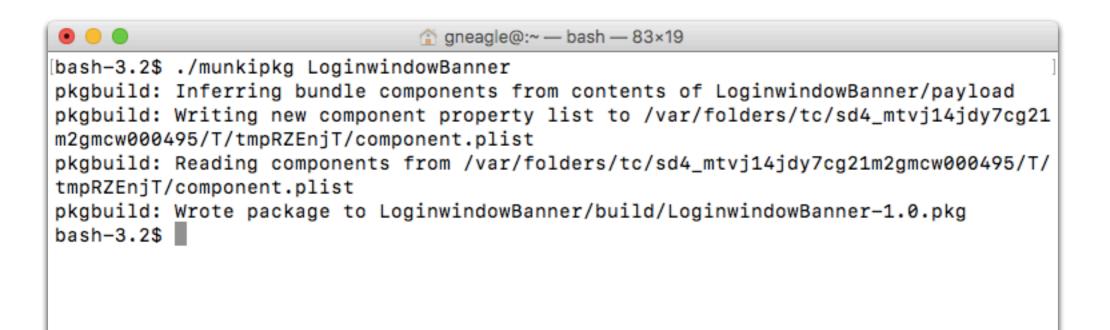


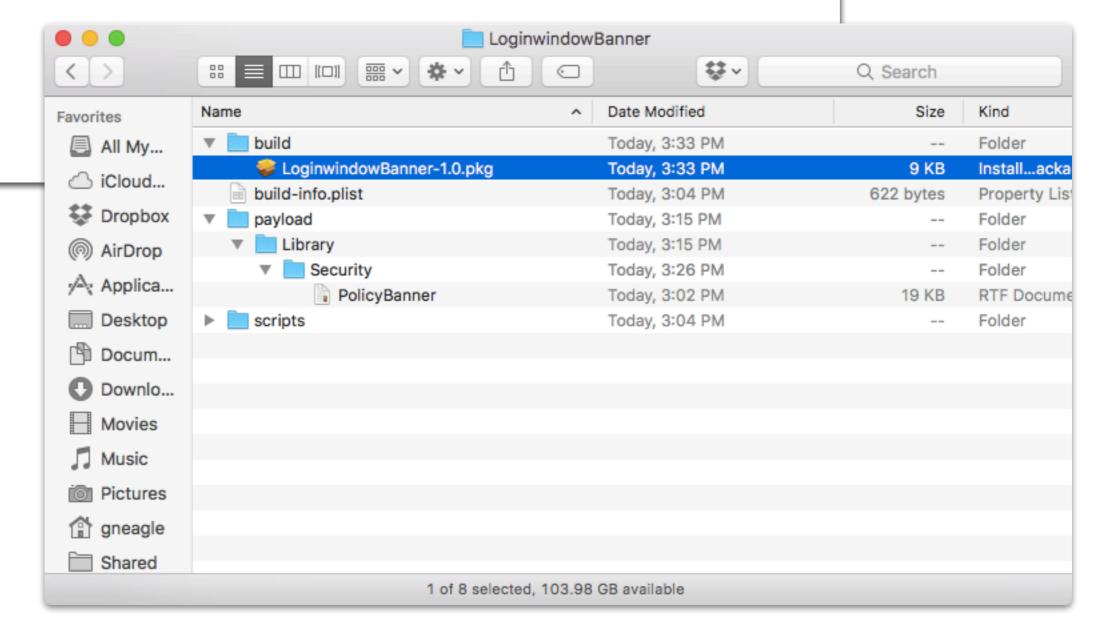






Building our package





build-info

build-info.plist

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/</pre>
PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>distribution_style</key>
    <false/>
    <key>identifier</key>
    <string>com.github.munki.pkg.Foo</string>
    <key>install_location</key>
    <string>/</string>
    <key>name</key>
    <string>Foo-${version}.pkg</string>
    <key>ownership</key>
    <string>recommended</string>
    <key>postinstall_action
    <string>none</string>
    <key>suppress_bundle_relocation</key>
    <true/>
    <key>version</key>
    <string>1.0</string>
</dict>
</plist>
```

build-info.json

```
"postinstall_action": "none",
"suppress_bundle_relocation": true,
"name": "Foo-${version}.pkg",
"distribution_style": false,
"install location": "/",
"version": "1.0",
"ownership": "recommended",
"identifier": "com.github.munki.pkg.Foo"
```

build-info.yaml

```
distribution_style: false
identifier: com.github.munki.pkg.Foo
install_location: /
name: Foo-${version}.pkg
ownership: recommended
postinstall_action: none
suppress_bundle_relocation: true
version: '1.0'
```

build-info keys

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://</pre>
www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>distribution_style</key>
    <false/>
    <key>identifier</key>
    <string>com.github.munki.pkg.Foo</string>
    <key>install_location</key>
    <string>/</string>
    <key>name</key>
    <string>Foo-${version}.pkg</string>
    <key>ownership</key>
    <string>recommended</string>
    <key>postinstall_action</key>
    <string>none</string>
    <key>suppress_bundle_relocation</key>
    <true/>
    <key>version</key>
    <string>1.0</string>
</dict>
</plist>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://</pre>
www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>distribution_style</key>
    <false/>
    <key>identifier</key>
    <string>com aithuh munki nka Foo</string>
    <ke distribution_style:</pre>
   If true, build as a "distribution-style" package using
   <st productbuild.
    Otherwise, build as a component package using pkgbuil
    <key>postinstall_action</key>
    <string>none</string>
    <key>suppress_bundle_relocation</key>
    <true/>
    <key>version</key>
    <string>1.0</string>
</dict>
</plist>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://</pre>
www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>distribution_style</key>
    <false/>
    <key>identifier</key>
    <string>com.github.munki.pkg.Foo</string>
    <key>install_location</key>
    <st
    <ke identifier:
    <st String containing the package identifier.
    <ke If this is missing, one is constructed using the name of
    the package project directory.
    <string>none</string>
    <key>suppress_bundle_relocation</key>
    <true/>
    <key>version</key>
    <string>1.0</string>
</dict>
</plist>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://</pre>
www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>distribution_style</key>
    <false/>
    <key>identifier</key>
    <string>com.github.munki.pkg.Foo</string>
    <key>install_location</key>
    <string>/</string>
    <key>name</key>
    <string For & Suggian Nac/ctains
    <ke install location:</pre>
    String. Path to the intended install location of the
    payload on the target disk. Defaults to "/".
    <key>suppress_bundle_relocation</key>
    <true/>
    <key>version</key>
    <string>1.0</string>
</dict>
</plist>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://</pre>
www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>distribution_style</key>
    <false/>
    <key>identifier</key>
    <string>com.github.munki.pkg.Foo</string>
    <key>install_location</key>
    <string>/</string>
    <key>name</key>
    <string>Foo-${version}.pkg</string>
    <ke_
    <st name:
    String containing the package name. If this is missing,
   one is constructed using the name of the package
    project directory.
    <key>version</key>
    <string>1.0</string>
</dict>
</plist>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://</pre>
www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <ke ownership:</pre>
    <fq String. One of "recommended", "preserve", or
    "preserve-other". Defaults to "recommended". See man
   <ke pkgbuild for more info.</pre>
    <struing>/</struing>
    <key>name</key>
    <string>Foo-${version}.pkg</string>
    <key>ownership</key>
    <string>recommended</string>
    <key>postinstall_action
    <string>none</string>
    <key>suppress_bundle_relocation</key>
    <true/>
    <key>version</key>
    <string>1.0</string>
</dict>
</plist>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://</pre>
www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <ke
    <fa postinstall_action:</pre>
    <ke String. One of "none", "logout", or "restart". Defaults to
    <st "none".
    <string>/</string>
    <key>name</key>
    <string>Foo-${version}.pkg</string>
    <key>ownership</key>
    <string>recommended</string>
    <key>postinstall_action</key>
    <string>none</string>
    <key>suppress_bundle_relocation</key>
    <true/>
    <key>version</key>
    <string>1.0</string>
</dict>
</plist>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://</pre>
www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
                <ke
                suppress_bundle_relocation:
                Object to the contract of t
                <st false, bundle relocation will be allowed, which causes
               the Installer to update bundles found in locations other
                than their default location.
                <string>Foo-${version}.pkg</string>
                <key>ownership</key>
                <string>recommended</string>
                <key>postinstall_action</key>
                <string>none</string>
                <key>suppress_bundle_relocation</key>
                <true/>
                <key>version</key>
                <string>1.0</string>
</dict>
</plist>
```

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://</pre>
www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <ke
    <fa version:
    <ke A string representation of the version number. Defaults
    <st to "1.0".
    <string>/</string>
    <key>name</key>
    <string>Foo-${version}.pkg</string>
    <key>ownership</key>
    <string>recommended</string>
    <key>postinstall_action</key>
    <string>none</string>
    <key>suppress_bundle_relocation</key>
    <true/>
    <key>version</key>
    <string>1.0</string>
</dict>
</plist>
```

Package signing

signing_info:

</dict>

The only required key/value in the signing_info dictionary is 'identity'.

See the SIGNED PACKAGES section of the man page for **pkgbuild** or the SIGNED PRODUCT ARCHIVES section of the man page for **productbuild** for more information on the signing options.

signing_info:

The only required key/value in the signing_info dictionary is 'identity'.

Importing existing packages

munkipkg --import /path/to/foo.pkg Foo

- Creates a new package project directory named "Foo"
- Payload, scripts and build-info are extracted from foo.pkg.
- •Complex or non-standard packages may not be extracted with 100% fidelity, and not all package formats are supported.

Demo (time permitting)