PSUMAC208:
PACKAGING
RUSTY MYERS

- Penn State University, Systems Administrator; Classroom and Lab Computing
OVERVIEW

• What Are Packages
• Why Use Packages
• How to Use Packages
• Demo Time
QUICK AUDIENCE SURVEY

• Created a Package?
• Iceberg?
PACKAGE INSTALLERS
WHAT PACKAGES DO

• User Installer.app /System/Library/CoreServices/
• Installs Files & Applications
• Run Scripts
PACKAGE SETTINGS

• Product Information
  Title, Description, Welcome, Read Me, License...

• Package Properties:
  Package Identifier, Version...

• Installation Properties
  System, Volume, & Authentication Requirements...

• Install Operations
  Pre & Post flight, install, upgrade

From:

Product information:
  Title
  Description
  Welcome file
  Read Me file
  License file
  Conclusion file

Package properties:
  Package identifier
  Package version number
  Resource fork processing

Installation properties:
  System requirements
  Volume requirements
  Authentication requirement
  Allowance for choosing an installation volume other than the boot volume
  Installation destination on the installation volume
  Rejection fee (the ability user may have to change the installation destination)
  Rejection consent
  Directory-permissions overwrite
  Postinstallation process action

Install operations:
  Preflight
  Preinstall/Preupgrade
  Postinstall/Postupgrade
  Postflight
PACKAGE INSTALLERS
What are packages

- File Directory or Flat File
- Appears as single file in Finder
- Created with PackageMaker
- `/Developer/Applications/Utilities`
- Contain Product or Component (The Payload)
- Installed based on Package Configuration


Defined by Apple:

An **installation package** (also known as a package) is a file package (a directory that appears in the Finder as a single file) created using the PackageMaker application (`/Developer/Applications/Utilities`). Packages contain a product or product component—the package’s **payload**—to be installed on a computer, and install configuration information that determines where and how the product is installed.
PACKAGE FORMATS

- .pkg
  - Component Package 10.2+
  - Flat Package 10.5+
- .mpkg
  - Metapackage 10.2+
  - Distribution Packages 10.4+

Specifies the minimum target operating system version. Defaults to 10.3. For 10.5, flat packages and metapackages will be built; for 10.4, bundle packages and distributions will be built; and for 10.3, bundle packages and metapackages will be built.

**Flat Package: XAR Archive**
10.5+
Flat packages contain a single product component. They are usually included as part of a distribution package or metapackage but can also be installed individually in computers running Mac.

**Component Packages:**
10.2+
Component packages contain a single product component. They are usually included as part of a distribution package or metapackage but can also be installed individually in computers running Mac.

**Metapackage:**
10.2+
**Multiple Component packages with product information**
Metapackages provide some of the features distribution packages provide but can be installed on computers running Mac OS X v10.2 and later.

**Distro Package:**
10.4+
**Provides product information and installation information**
Distribution packages let you define the complete install experience of your product. They also provide you with a great deal of flexibility for defining the install choices users use to customize an install. Distribution packages offer you and the users of your product the best installation solution for Mac OS X–based products. Distribution packages, however, can be installed only on computers running Mac OS X v10.4 and later.
COMPONENT PACKAGE

- Single choice
- No Custom Choices
- Some parts editable in Finder
- Folder Bundle
  - Right-Click -> Show Package Contents
Description
PackageInfo
This is a XML document that contains information about the package behavior, requirements, and identity.
Bom
This is the Bill of Materials for the files contained in the Payload archive. See mkbom (8).
Payload
This is an archive of the hierarchy of files to be installed. The hierarchy is saved as cpio archive compressed with gzip. See cpio(1), ditto(1), gzip (1).
Scripts
This is an archive of scripts and additional resources. The hierarchy is saved as cpio archive compressed with gzip. See cpio(1), ditto(1), gzip(1). This file is optional.
RunAtStartup
This is a shell script that will be supposedly invoked on the next Mac OS X startup.

http://s.sudre.free.fr/Stuff/Ivanhoe/FLAT.html
en.lproj: Language pack for product information

package_version:
major: 3
minor: 6

TokenDefinitions.plist:
Unique Identifier
Install Path
searchPlugin - Path to Search System for old versions of app

scripts are stored here as postflight & preflight (Also Preupgrade, PostUpgrade, PreInstall, Postinstall)
Each choice is a component package
distribution script runs all package installs
Packages are component packages

Resources have localizations
Resources have localizations
FLAT PACKAGE

• Single File - XAR Archive
• Edit with Flat Package Editor
• Expand Flat Packages
  • pkgutil --expand package.pkg /path/to/destination

shows as single file on other file systems and distribution systems
PackageInfo - Unique Identifier, Package Settings
BOM - Permissions and Destination for Payload files
Payload - Files to install
Scripts - Scripts to run preinstall or postinstall
WHY USE PACKAGES
WHY USE PACKAGES

• Silent
• Free
• Easy Deployment
• Capture Licensing Info
• Set Computer Requirements
• Log of Install Files and Permissions (Bill of Materials)

Can be installed silently via command line/Apple Remote Desktop
Sexy!
DEPLOYING PACKAGES

• Disk Image Container
• Apple Remote Desktop
• Command Line
  • `sudo installer -pkg /path/to/pkg -target /
• 3rd Party Tools
HOW TO MAKE PACKAGES
HOW TO MAKE PACKAGES

- Manually Define Payload
- GUI
- Command Line
- File System Events
- Before & After Snapshots
PACKAGING TOOLS

• Manual
• GUI
  • PackageMaker
  • Iceberg
  • Composer
  • Packages
• InstallEase
• Command Line
  • PackageMaker
  • luggage
  • Packages
PACKAGING TOOLS

- File System Events
- GUI
  - PackageMaker
  - Composer
  - Packages
  - FSEventer
- Command Line
  - PackageMaker
  - fslogger
PACKAGING TOOLS

- Snapshots
- GUI
  - Composer
  - Packages
  - InstallEase
- Command Line
  - loggen/pkggen

[Images: GUI, Composer, Command Line]
BASIC PACKAGING

- Manually choose Install files
- Manually set Installation Destination
- Can be complicated
  - Experience
  - When to Include & Exclude Files and Components
Organization is:
Bundle Identifier
This is a unique identifier string for the bundle. This identifier should be in the form of a Java-style package name, for example com.mycompany.myapp. The bundle identifier can be used to locate the bundle at runtime. The preferences system also uses this string to uniquely identify applications.
http://www.advancedinstaller.com/user-guide/mac-os-tab.html
The bundle identifier string identifies your application to the system. This string must be a uniform type identifier (UTI) that contains only alphanumeric (A-Z,a-z,0-9), hyphen (-), and period (.) characters. The string should also be in reverse-DNS format. For example, if your company’s domain is Ajax.com and you create an application named Hello, you could assign the string com.Ajax.Hello as your application’s bundle identifier.
The bundle identifier is used in validating the application signature.
MINIMUM TARGET

- Mac OS X 10.3
  - component packages & metapackages
- Mac OS X 10.4
  - component & distribution packages
- Mac OS X 10.5
  - flat packages & metapackages

Specifies the minimum target operating system version. Defaults to 10.3. For 10.5, flat packages and metapackages will be built; for 10.4, bundle packages and distributions will be built; and for 10.3, bundle packages and metapackages will be built.
BASIC PACKAGING

• Set the
• Title
• Install Destination
• Don’t be cruel
• Description

Allow the users to choose the install destination if possible
Use descriptions for your notes
BASIC PACKAGING

- Set the Destination
- Package Version
- Make sure Package Identifier is unique

Increment Package version to upgrade the previous payload - Identifier must be the same
BASIC PACKAGING

• Do you need to...
  • Restart?
    • Only if Needed
  • Require Admin authentication?
    • Installing in /Users/ or /tmp
  or

If you don’t need to restart, don’t. extra hassle for users
Admin Auth: If your not sure, drag the app/component to install location. Does it ask for auth?
BASIC PACKAGING

- Unchecked items are not installed.
- Set
  - Permissions
  - Owner & Group
BASIC PACKAGING

- Add Script if used
- Preflight Runs Before Script
- Postflight Runs After Script
- Add script resources to "Scripts directory"

Scripts Directory resources are installed into the Contents/Resources of the package.
EVENTS PACKAGING

- Finds New and Modified Files for You in Real Time
- Does not survive Restarts
- Can capture unwanted files
EVENTS PACKAGING

- Choose Project Menu & Add Snapshot Package
- Install Software
- Stop Capture
EVENTS PACKAGING

• Choose Project Menu & Add Snapshot Package

• Install Software

• Stop Capture

• Uncheck boxes to remove files
EVENTS PACKAGING

- Same settings as Basic Package
- Double Check Payload
SNAPSHOT PACKAGING

- Find Filesystem Differences
- Can capture unwanted files
- Can handle restarts
COMPOSER SNAPSHOTs

- Choose Snapshot Method
COMPOSER SNAPSHOTs

• Choose Snapshot Method

• Set Package Name
COMPOSER SNAPSHOTS

- Choose Snapshot Method
- Set Package Name
- First Snapshot is taken
COMPOSER SNAPSHOTS

- Choose Snapshot Method
- Set Package Name
- First Snapshot is taken
- Install Software
COMPOSER SNAPHOTS

- Second Snapshot is taken
- Remove unwanted files
- Package is built
COMMAND LINE PACKAGING

- Scriptable
- Advanced
- Easily Repeatable
DEMO TIME
ICEBERG
ADVANCED PACKAGES...
Also on choice requirements
ACTIONS

• Pre and Post Install
• Workflow style
Payload can sit in multiple places.

- Internal (Default)
- Same Level
- Custom Path
- HTTP URL
- Removable Media
Most common languages are Perl and Shell

Scripts have variables to help

You can call other included files/scripts/apps - They are located in the .pkg/Contents/Resources/ folder

Important: In Mac OS X v10.5 clients, the only install operations available are preinstall and postinstall.

Note: Consider defining preinstall and postinstall actions on the product package (see “Product Package Actions Pane”) instead of preinstall and postinstall operation in component packages. The latter are inherently less secure and, therefore, causes the warning described earlier.
SHELL VARIABLES

• $0 = Path to this Script
• $1 = Destination path of where package is being installed
• $2 = Path to the Target Location
• $3 = Mountpoint of the destination volume
PERL VARIABLES

• $0 = Path to this script
• $ARGV[0] = Path to the Package being installed
• $ARGV[1] = Destination path of where package is being installed
• $ARGV[2] = Mountpoint of the destination volume
• $ARGV[3] = Path to the directory containing the System
PAYLOAD FREE

• Run a script
• Doesn't install anything
• Easier for Users to Run
• Scripts receives target volume path ($3)
PAYLOAD FREE

- % touch emptyfile
- Add to PackageMaker
- Uncheck Contents
- Add Post Install script
PAYLOAD FREE

- % touch emptyfile
- Add to PackageMaker
- Uncheck Contents
- Add Post Install script
PAYLOAD FREE

- Iceberg is Easier
POST-PACKAGING
INSTALLED PACKAGES

- List installed packages by Unique Identifier 10.6 Only
  - pkgutil --pkgs
- History of Installed Packages 10.6+
  - /Library/Receipts/InstallHistory.plist
VERIFY INSTALLATION

• Package Receipt Locations
  • 10.6 ≥ /var/db/receipts/
  • 10.5 ≤ /Library/Receipts/
LIST BILL OF MATERIALS (BOM)

• 10.5
  lsbom -p UGMsF /Library/Receipts/boms/com.org.app.bom

• 10.6
  lsbom -p UGMsF /var/db/receipts/com.org.app.bom

• Files may be changed with postflight scripts
FORCE REINSTALL

- Remove Old Files
- Remove Receipt
  - sudo pkgutil --forget com.org.pkg --volume /
MAINTAINING PACKAGES

• Use Version Numbers
• Use Component Package
  • Easier to edit Script
• Save PackageMaker (Or Other) Project File (.pmdoc)
  • Reusable settings (Permissions, Destination, Scripts)
PACKAGE NAMES & VERSIONS

```bash
$ find /path/to/some/folder -name ".pkg" | while IFS= read -r x; do
  echo -n "${x##/*/} "; grep -A1 CFBundleShortVersionString "$x/Contents/Info.plist" | 
  sed '/string/!d;s@^.*<string>\([^<]*\)</string>.*$@\1@g'; done

$ find /Volumes/iTunes 8.1.1 -name ".pkg" | while IFS= read -r x; do
  echo -n "${x##/*/} "; grep -A1 CFBundleShortVersionString "$x/Contents/Info.plist" | 
  sed '/string/!d;s@^.*<string>\([^<]*\)</string>.*$@\1@g'; done

iTunes.mpkg 8.1.1
AppleMobileDeviceSupport.pkg 2.3
CoreFP.pkg 1.3
iTunesAccess.pkg 8.1
iTunesX.pkg 8.1.1

script fails when package doesn't have a *.pkg/Contents/Info.plist file. Instead they have a *pkg/some.pkg/PackageInfo file.

```
TOOLS TO EXAMINE OR EDIT PACKAGES

Bundle

Finder

Terminal

Flat

Flat Package Editor

Pacifist

XAR Tool

http://charlessoft.com/
Flat Package Editor in PackageMaker Resources in Developer Tools
THE LUGGAGE

• Makefiles with Payloads/Scripts Referenced
• Easy to Review Packages
• Easy to Rollback Changes (SVN)

http://luggage.apesseekingknowledge.net/
TOOLS

IceBerg, Packages, Pacifist, InstallEase
http://s.sudre.free.fr/Packaging.html

http://blog.macadmincorner.com/mac-software-packaging-utilities-list/
REFERENCES

Apple Software Delivery Guide
http://tinyurl.com/SoftwareDeliveryGuide

PackageMaker User Guide
http://tinyurl.com/PackageMakerUG

MacEnterprise: Packaging for Sys Admins
http://tinyurl.com/MacTechPackaging
PACKAGING GUIDES

Sudre PackageMaker How To
http://s.sudre.free.fr/Stuff/PackageMaker_Howto.html

Sudre Flat Package Missing Documentation
http://s.sudre.free.fr/Stuff/Ivanhoe/FLAT.html
Q & A

Rusty Myers  rzm102@psu.edu
@thespider