Course W5: Automation and Synchronization o Disks and Images

Dave Pooser

geekboy@pooserville.com

What this course *isn't* about:

- Lab environments where machines are regularly wiped and re-imaged
- Thin-client models where Netbooting and network home directories make desktops interchangeable
- Using rsyncx or similar to automatically revert images on a nightly basis

Designing Initial Images

Plan, plan, plan!

“Organizing is what you do before you do something, so that when you do it, it is not all mixed up.”

—A. A. Milne

Questions to ask

- What are the core apps everyone uses?
  - Don’t forget UNIX commands!
    - (Why CpMac is only in Xcode I’ll never know…)
- Is localization going to be an issue in your environment?

More questions to ask

- How is your software licensed?
  - You don’t want to manage different serial numbers for every user
  - You can’t afford to risk having unlicensed software
  - Volume licenses schemes like Office v.X or Adobe Acrobat can be a huge help
  - Keyserver http://www.sassafras.com may improve your life dramatically.

The Great Debate

Should users be local administrators?

“What I do say is that no man is good enough to govern another man without that other's consent.”

—Abraham Lincoln

Should users be local admins?

- YES
  - Political reasons
  - Laptop users
  - Remote users
  - Flexibility
- NO
  - Ease of management
  - Standard configuration
  - ARD/ Timbuktu/ NetOctopus for remote support
  - One word: sudo!

Probable outcome:

- Some local admins
  - Road warriors
  - Power users
  - Management
- Most users not administrators

All this depends on organizational culture!

(and whether the IT person before you was a jerk)

Creating universal administrators

- At least one on every machine
  - More than one allows different capabilities for different individuals/departments
- Same password makes administration easier
  - Guard that password with your LIFE!
- Making them invisible improves security **marginally**, reduces login clutter
  - To make them invisible, just give them UID <500 and hide their home directory

Creating your master

“One lesson we learn early, that in spite of seeming difference, men are all of one pattern.”

—Ralph Waldo Emerson

One master, or more?

- Why would you create multiple masters?
  - Software: Fundamentally different applications
  - Hardware:
    - Portable vs Desktop
    - New models running non-standard OS
      - Think of the early G5s before Panther
      - Generally takes Apple 1-2 revs to “un-fork” software

Tune your master

Delocalization if appropriate

- Tweak preferences as needed
  - /System/Library/User Template/English.lproj
- Configure network settings
  - Master hardware should have as many interfaces as the most flexible client
    - Ethernet
    - Airport
    - Modem

Test, test, TEST!

- Install on multiple hardware types
- Test applications extensively
    - Launch every app as an administrator and a user
        - Some problems will only affect a user
        - For example: Acrobat registration database could only be read by an admin; users had to enter the serial on every launch
        - If everyone is an admin, skip the user test, and have a drink instead. You’ll need it.
    - Once you think it’s perfect, beta test it with a few power users

Creating and deploying images

Creating your image

- Three primary tools
  - Carbon Copy Cloner http://www.bombich.com
    - The first OS X GUI imaging tool; still very popular
  - Disk Utility (Panther only)
    - Seems improved in later revisions of 10.3.x
  - Network Image Utility— for NetInstall

Deploying your image

“Real artists ship.”

—Steve Jobs

Deployment techniques

- Local boot, local image
- Local boot, network image
- Network boot, network image

Local boot, local image

- CCCloner or Disk Utility
- FireWire hard drive (iPod cool, but slow)
- Doesn’t scale well
- So why do it this way?
  - Minimizes stress on slower networks
  - Politics may prevent using a Mac server
  - Offers a chance for some basic user training

Local boot, network image

- Boot from FireWire (including iPod)
- CCCloner and Disk Utility can restore from .dmg files on mounted server
- Disk utility can also restore from Web server
- Still labor- (and walking-) intensive
- So what advantages?
  - Mac server not required
  - Network image means all you need is a bootable drive
  - You may be able to expense an iPod!

Network boot, network install

- Least labor-intensive
- Most network-intensive
- Requires Mac OS X server
- Two options
  - NetInstall— file-level copy
  - NetRestore— block-level copy
    - Faster than NetInstall
    - Current version also be used to create image
    - http://www.bombich.com

Using Apple Remote Desktop

Client administration using ARD

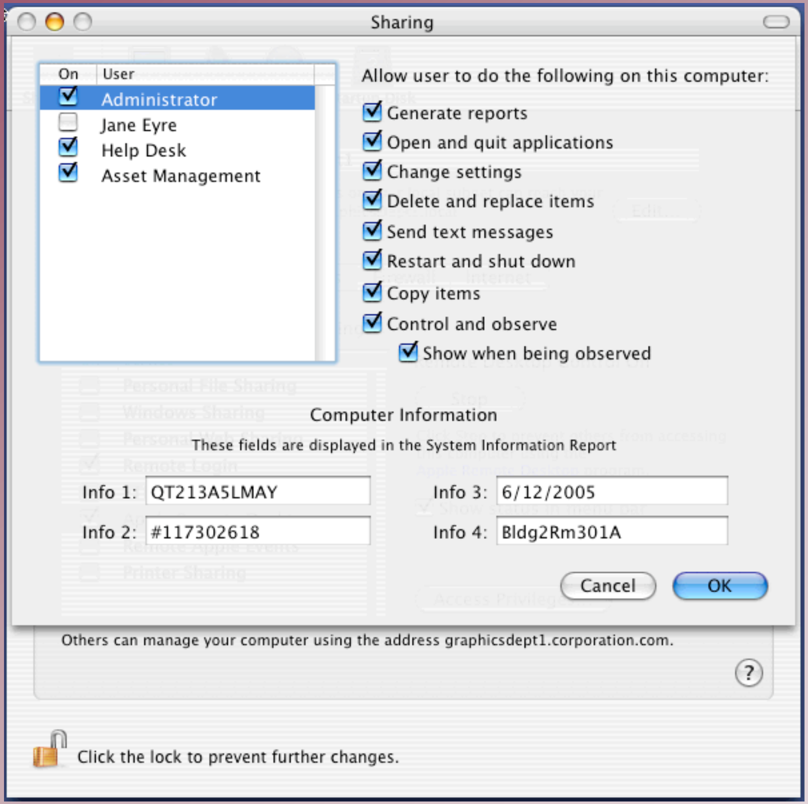
“Progress is made by lazy men looking for easier ways to do things.”

— Robert Heinlein

ARD overview

- Remember Network Assistant?
- Setup
  - Configure clients to allow ARD access
    - This should be part of your master image
    - 10.3 uses the Sharing preference pane
    - 10.2.x and 10.1.5 use Remote Desktop pane
    - Nobody cares about OS 9.x and earlier, right?
  - Create lists of clients on the admin station(s)
- UDP multicast— scales well, doesn’t stress network

Configuring clients



On	User
✓	Administrator
	Jane Eyre
✓	Help Desk
✓	Asset Management

Allow user to do the following on this computer:

- ✓ Generate reports
- ✓ Open and quit applications
- ✓ Change settings
- ✓ Delete and replace items
- ✓ Send text messages
- ✓ Restart and shut down
- ✓ Copy items
- ✓ Control and observe
  - ✓ Show when being observed

These fields are displayed in the System Information Report

Info 1: QT213A5LMAY
Info 2: #117302618
Info 3: 6/12/2005
Info 4: Bldg2Rm301A

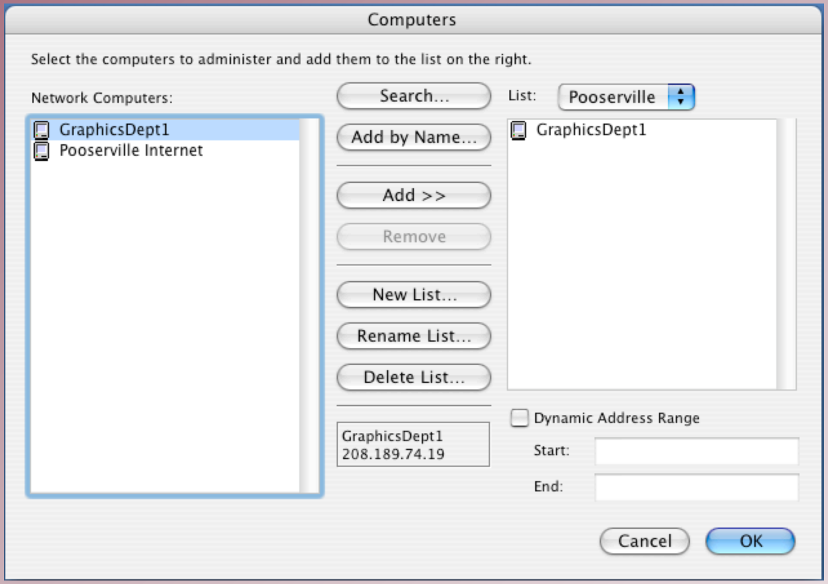
Others can manage your computer using the address graphicsdept1.corporation.com.

Click the lock to prevent further changes.

- Nicely granular access privileges
- Pane will show all users >500
- Command-line nerds can also use `kickstart`
- Information fields won't autopopulate
  - But `kickstart` can be used with a shell script…

Configuring admin stations

- Automatically finds clients on same subnet
- Add remote clients by name or search IP range
- Sort clients into 20 lists; one client can be in multiple lists



Computers

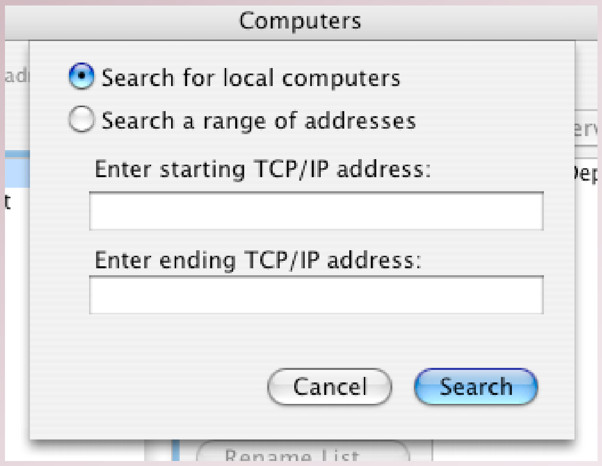
Select the computers to administer and add them to the list on the right.

Network Computers:
- GraphicsDept1
- Pooserville Internet

List: Pooserville
- GraphicsDept1

GraphicsDept1
208.189.74.19

☐ Dynamic Address Range
Start:
End:



Computers

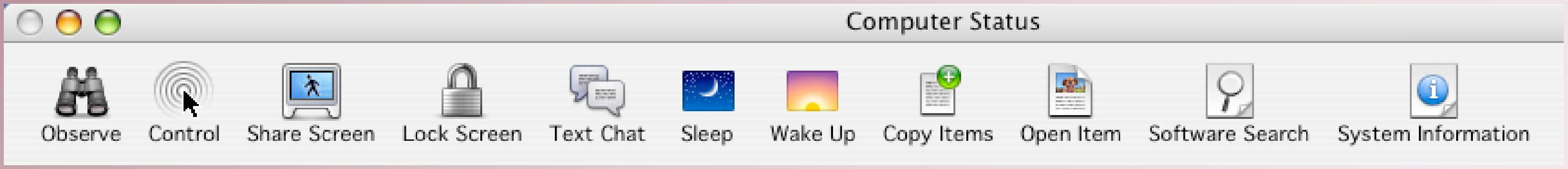
[✓ Search for local computers] Search a range of addresses

Enter starting TCP/IP address:

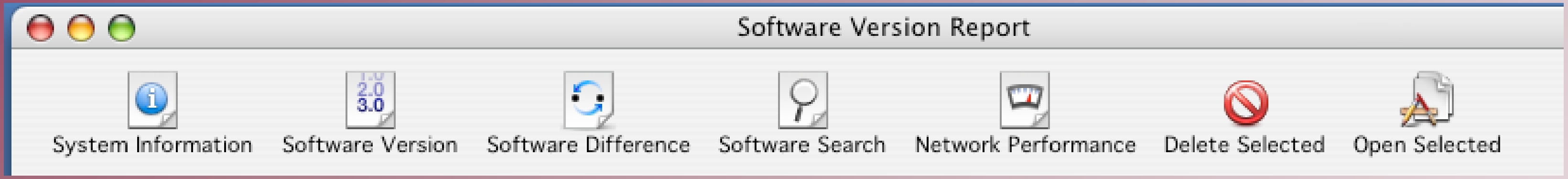
Enter ending TCP/IP address:

ARD interactive features

- Not really relevant to this session…
  - Control/Observe client machine
  - Share/lock screen
  - Text chat with user



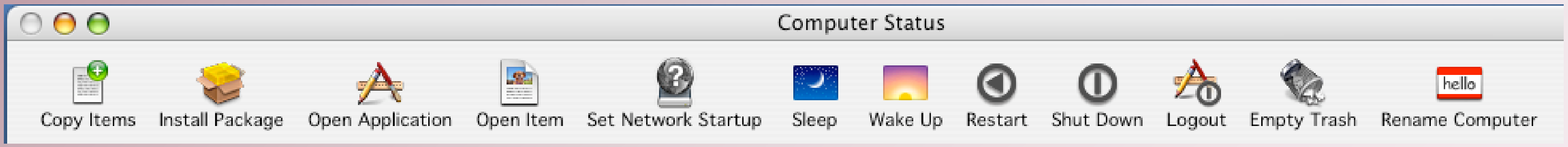
ARD Reporting features



- Reports
  - System Information for hardware and OS settings
  - Software Versions for installed software
  - Software Difference compares client(s) to admin station
  - Software search searches for any file on the client(s)
  - Delete and Open Selected for files found using other reports
  - Network Performance— test net between you & client

ARD Management features

- All sorts of cool capabilities, especially
  - Install Package— this is the essence of this session
  - Copy items— for pushing single files/folders
  - Open item— great for running AppleScripts on multiple remote machines simultaneously
- Management tasks happen behind the scenes
- Target multiple machines simultaneously
  - You should hear angels singing at this point!



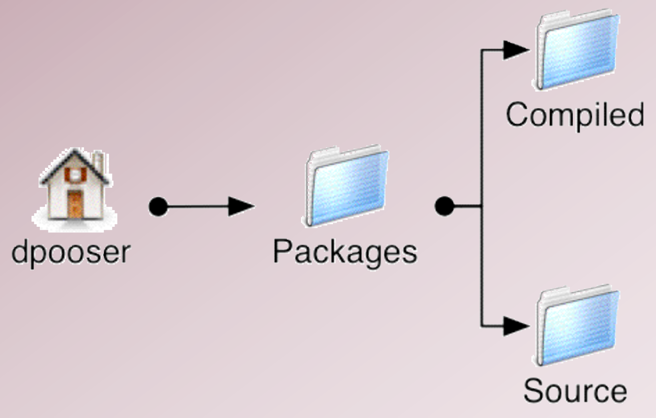
PackageMaker essentials

Why use PackageMaker?

- Install multiple items to multiple locations
- Install items that require special privileges
- The ability to do pre-install or post-install tuning via scripts
- Create metapackages to simplify updating

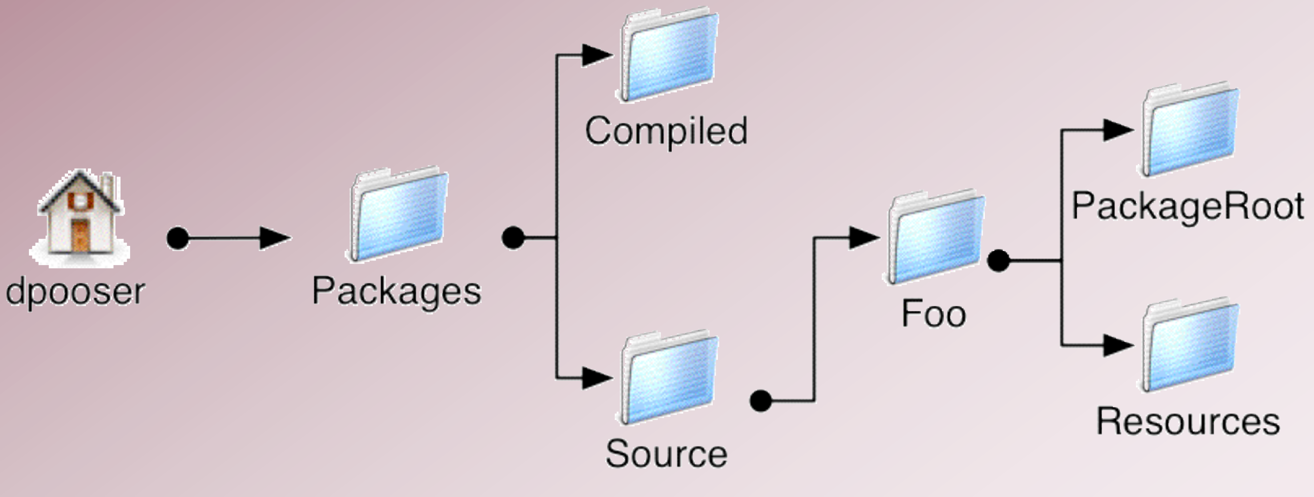
PackageMaker preparation

- If you don't have Xcode installed (why on earth not?) install it
- Create a standard folder structure to hold source and completed projects



Creating a simple package

- Create a package-specific folder structure
  - ~/Packages/Source/Foo/
  - ~/Packages/Source/Foo/PackageRoot/
  - ~/Packages/Source/Foo/Resources/



Add files to Resources

- Installer options
  - Welcome.txt (or .rtf, or .html)
  - License.txt
  - ReadMe.txt
  - background.jpg (optional)
    - If you want a cool background to the installer
    - Can also be .tiff, .gif, .pict, .eps, or .pdf file
- Scripts— may include any/all of:
  - `InstallationCheck`
  - `VolumeCheck`
  - `preflight`
  - `preinstall/preupgrade`
  - `postinstall/postupgrade`
  - `postflight`

Add structure to PackageRoot

- PackageRoot should simulate “lowest common directory” — that is, lowest in the directory tree that contains everything being installed
  - Example: if all files go into /Applications then PackageRoot is simulating /Applications
  - Example2: if files go into /Applications and /Library then PackageRoot is simulating / and must contain Library and Applications subdirectories
- WARNING: The installer will overwrite symlinks with directories. If PackageRoot is simulating / and you need to put files in /etc, remember it is really /private/etc — overwriting the `/etc -> /private/etc` symlink will ruin your day!

Prepare the files

- Put files in PackageRoot and subdirectories
  - Be aware of resource forks
  - Copy files using the Finder or use `ditto -rsrc` or `CpMac` to preserve resource forks if necessary
- Run `SplitForks` from /Developer/Tools on the PackageRoot directory
- Delete any .DS_Store files (Finder droppings) using Terminal
- Double-check permissions; make sure scripts are executable

Running PackageMaker

- Found in /Developer/Applications/Utilities
- Description tab— pretty straightforward. The “Delete Warning” is not currently implemented
- Files tab— Root is full path to PackageRoot. Compress Archive saves space, takes more time to create and install
- Resources tab— wants path to Resources folder

[continued]

Running PackageMaker [continued]

- Info tab — where most of the settings go
  - Default location: directory simulated by PackageRoot
    - Probably / or /Applications
  - Restart Action— No Restart Required, Required Restart, Shutdown Required
    - Also Recommended Restart, but that doesn’t work
    - Why Shutdown? Think hardware drivers…
    - You shouldn’t have to restart under most circumstances
  - Authorization Action— determines install permissions, ownership of files
    - Admin for installing to /Applications, /Library
    - Root for installing to /System, /private/var, and so on

Running PackageMaker [continued]

- Info tab— Flags
  - Most of these should be left unchecked
  - Relocatable— only for self-contained apps
    - Even then, do you want to wonder where's Waldo.app every time you update a machine?
  - Required— BAD idea in a single package
  - Root Volume Only— only relevant if Relocatable checked
  - Update Installed Languages Only— handy if you run some multilingual systems
  - Overwrite permissions— normally not needed

Running PackageMaker [continued]

- Version tab
  - Display name is what will show in the Finder
  - Other Display Information contents should match the Info.plist settings in the application package’s Contents subdirectory
    - “Get Info string” = CFBundleGetInfoString
    - “Identifier” = CFBundleIdentifier
    - “Short version” = CFBundleShortVersionString
  - Version— Major is portion left of the first decimal point
    - Example: Version 6.5.9 is Major 6, Minor 59

Create the package!

- Select Create Package under File menu
  - Save package into ~/Packages/Compiled
  - If PackageMaker offers to split forks, choose "Don't Split"
    - Some users have found the command-line `SplitForks` more reliable
  - Save PackageMaker document for future updates

Metapackage creation

What is a metapackage?

- Just a list of packages (possibly including other metapackages) and the info needed to install them
- Packages, like batteries, are not included

Why make a metapackage?

- For interactive installs
  - Allows more modular software installation
    - Enables “Customize” button in installer
    - Some components may be required and others optional, such as fonts or clip art
  - Avoids multiple restarts
    - Install a Security Update and a new version of QuickTime without rebooting between them
- For remote installs
  - Avoids multiple restarts
  - Easier to update metapackage when one component changes

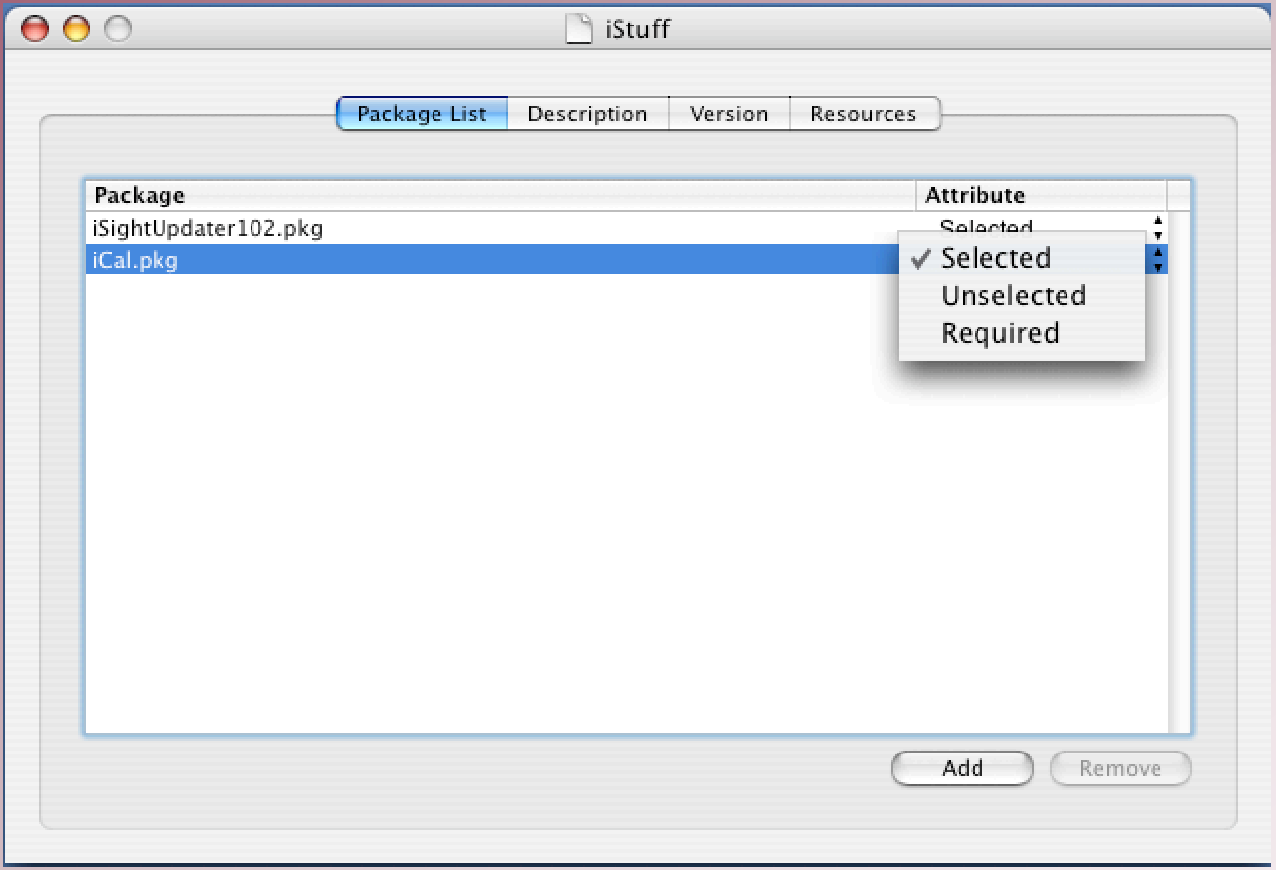
Preparing a metapackage

- Create package-specific folder structure
  - ~/Packages/Source/MetaFoo/
  - ~/Packages/Source/MetaFoo/Resources/
- Add files to resources
  - Installer will NOT show Welcome, License or ReadMe files from component packages

Assembling a metapackage

- Launch PackageMaker and go to File:New —> Metapackage
- PackageMaker opens a new window with an empty package list
- Drag one or more packages to the list
  - Installation order is determined by position in the list (top goes first)
  - Each package can be set to Required, Selected (by default) or Unselected (by default)

Metapackage package list



Package	Attribute
iSightUpdater102.pkg	Selected
iCal.pkg	✓ Selected / Unselected / Required

Creating a metapackage

- Fill out Description, Version and Resources tabs as usual
- Select Create package under File menu
- Save package into ~/Packages/Compiled
- Move component packages into ~/Packages/Compiled
  - Or edit IFPkgFlagComponentDirectory in the metapackage Info.plist (found in Contents)
    - (Info.plist will need to be made writeable for this to work)
  - By default, the metapackage expects its components in its own directory

Converting other installers

Converting installers— overview

- Start with a clean system— OS + Xcode
- Get a baseline list of all files on the system
- Install the software, launch, and enter license information
- Get an update list of all files on the system
- Generate a list of changed files
- Create a package from the list of changes

Converting installers— detail

- Start with a clean system
    - Install OS with all the bells and whistles and all software updates
        - This should include every language you support
    - Add Xcode and updates as required
    - Clone this system— you should plan on erasing it after every installation

Get a baseline list of every file

- Download and install `logGen` http://www.lsa.umich.edu/lsait/AdminTools/osx/software/
  - Requires 10.3 (or 10.2 + Digest and File::Spec Perl modules)
- At the command line type `sudo /usr/local/sbin/logGen basefiles.log`

Install the software

- Install the software with every option any user might need
  - Remember it’s easier to remove than add components at the packaging stage!
- Launch the software and enter generic licensing information
  - Quit and relaunch the software to make sure the licensing information took
  - Log in as a user and make sure everything still works as expected

Generate a list of changed files

- `logGen` to the rescue again!
  - `sudo ./logGen current.log basefiles.log > Acrobat6.log`
- Acrobat6.log (in this example) is now a blueprint for creating a package
- `logGen` will list every file that has been added, changed or deleted, or the parent directory if every file has been affected

Create a package from the list

- Manually copy each file or directory to its appropriate location under PackageRoot
- Or, use a shell script to parse the `logGen` output and automatically copy files to their proper locations
  - Remember to use `CpMac` or `ditto -rsrc` to preserve resource forks
  - Some paths may have spaces and will need to be quoted
- Consider creating a metapackage, with separate packages for optional installs

Distributing configuration files

Two main OS X config methods

- Traditional Unix— text configuration files
  - Lives on in .plists, which are just XML files
  - Editable directly with shell and Perl scripts
- Traditional Mac— binary preference files
  - Still common in many Carbon apps
  - Editable only through creating application, which may or may not be AppleScriptable

The good news— we can handle both!

- Shell scripts can execute AppleScripts
- AppleScripts can call shell scripts
- Packages can replace configuration files
- As administrators, we get our duct tape and our power tools too!

Why not use ARD Copy item?

- Copy Item is simpler for single files
- Packages give you more flexibility
  - Pre-flight script can back up existing configuration
  - Packages can place several items in different locations
  - Post-flight script can force a daemon to reload its .conf file to apply changes without a reboot

Example: Pushing out a new printer

- Preflight script backs up `/private/etc/cups/printers.conf`
- Package places new PPD in `/private/etc/cups/ppd/` and new partial .conf file in `/private/tmp/newprinter.conf`
- Postflight script adds the partial .conf file to `/private/etc/cups/printers.conf` and deletes `/private/tmp/newprinter.conf` before sending a `kill -1` to cupsd

More fun with scripts

Six kinds of scripts

- `InstallationCheck` checks for system characteristics and can permit or deny installation
- `VolumeCheck` checks for volume characteristics and can permit or deny installation on a specific disk
- `preflight` runs before any files are copied
- `preinstall/preupgrade` runs before any files are copied, depending on whether the package is an upgrade
- `postinstall/postupgrade` runs after all files are copied, depending on whether the package is an upgrade
- `postflight` runs last of all

When are they used?

- `InstallationCheck, VolumeCheck`
  - These scripts are most often used as a failsafe, to make sure you don’t try an install while missing prerequisite hardware or software
- `preflight`
  - Used for any operation that needs to take place prior to both installs or upgrades— for instance, stopping any running SMTP servers before installing a new one

When are they used? [continued]

- `preupgrade, postupgrade`
  - Handy for backing up files that might be overwritten, and then for moving them back once the upgrade is complete
- `preinstall, postinstall`
  - Useful for configuring default settings for a new application
- `postflight`
  - All sorts of things from cleaning up temp files to launching a newly-installed app

Running shell scripts via ARD

Creating a script-only .pkg

- Only requirement is a postflight script in Resources and an empty PackageRoor
- Set authentication in PackageMaker to desired level, either admin or root
- Like any package, the script can be run on multiple clients simultaneously
- SSH access doesn’t have to be turned on