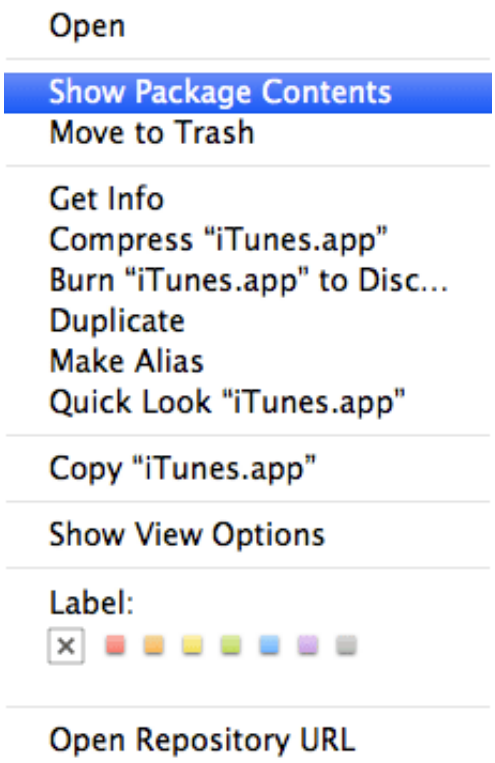
How to create simple Mac apps from shell scripts

Published 12th November 2010 · tagged with Bash, OS X

Basically, a Mac application has a `.app` extension, but it's not really a file — it's a package. You can view the application's contents by navigating to it in the Finder, right-clicking it and then choosing "Show Package Contents".



The internal folder structure may vary between apps, but you can be sure that every Mac app will have a `Contents` folder with a `MacOS` subfolder in it. Inside the `MacOS` directory, there's an extension-less file with the exact same name as the app itself. This file can be anything really, but in its simplest form it's a shell script. As it turns out, this folder/file structure is all it takes to create a functional app!

Enter appify

After this discovery, Thomas Aylott came up with a clever "appify" script that allows you to easily create Mac apps from shell scripts. The code looks like this:

```
#!/bin/bash

APPNAME=${2:-$(basename "$1" ".sh")}
DIR="$APPNAME.app/Contents/MacOS"
```

```
if [ -a "$APPNAME.app" ]; then
  echo "$PWD/$APPNAME.app already exists :("
  exit 1
fi

mkdir -p "$DIR"
cp "$1" "$DIR/$APPNAME"
chmod +x "$DIR/$APPNAME"

echo "$PWD/$APPNAME.app"
```

Installing and using appify is pretty straightforward if you're used to working with UNIX. (I'm not, so I had to figure this out.) Here's how to install it:

1. Save the script to a directory in your `PATH` and name it `appify` (no extension). I chose to put it in `/usr/local/bin`, which requires root privileges.
2. Fire up Terminal.app and enter `sudo chmod +x /usr/local/bin/appify` to make appify executable without root privileges.

After that, you can create apps based on any shell script simply by launching Terminal.app and entering something like this:

```
$ appify your-shell-script.sh "Your App Name"
```

Obviously, this would create a stand-alone application named `Your App Name.app` that executes the `your-shell-script.sh` script.

After that, you can very easily add a custom icon to the app if you want to.

Adding a custom app icon

1. Create an `.icns` file or a 512×512 PNG image with the icon you want, and copy it to the clipboard (⌘ + C). (Alternatively, copy it from an existing app as described in steps 2 and 3.)
2. Right-click the `.app` file of which you want to change the icon and select "Get Info" (or select the file and press ⌘ + I).
3. Select the app icon in the top left corner by clicking it once. It will get a subtle blue outline if you did it right.
4. Now hit ⌘ + V (paste) to overwrite the default icon with the new one.

Note that this will work for any file or folder, not just `.app` files.

Examples

Chrome/Chromium bootstrappers

I like to run Chrome/Chromium with some command-line switches or flags enabled. On Windows, you can create a shortcut and set the parameters you want in its properties; on a Mac, you'll need to launch it from the command line every time. Well, not anymore :)

```
#!
/Applications/Chromium.app/Contents/MacOS/Chromium --enable-benchmarking
--enable-extension-timeline-api&
```

The `&` at the end is not a typo; it is there to make sure Chromium is launched in a separate thread. Without the `&`, Chromium would exit as soon as you quit Terminal.app.

Launch a local web server from a directory

Say you're working on a project and you want to debug it from a web server. The following shell script will use Python to launch a local web server from a specific directory and open the index page in your default browser of choice. After appifying it, you won't even need to open the terminal for it anymore.

```
#!
cd ~/Projects/Foo/
python -m SimpleHTTPServer 8080 &> /dev/null &
open http://localhost:8080/
```

More?

Needless to say, the possibilities are endless. Just to give another example, you could very easily create an app that minifies all JavaScript and CSS files in a specific folder. Got any nice ideas? Let me know by leaving a comment!

About me

Hi there! I'm Mathias, a web standards enthusiast from Belgium. HTML, CSS, JavaScript, Unicode, performance, and security get me excited. If you managed to read this far without falling asleep, you should follow me on Twitter and GitHub.

Comments

Thomas Aylott wrote on 12th November 2010 at 18:41:

Nice examples.

I'd love to see what else people can come up with. What'd be slick is a droplet that lets you drag a folder into the app to serve that path. Maybe use AppleScript via `osascript`

to give it a simple UI or something.

Fun Times™

Mathias wrote on 12th November 2010 at 18:46:

Someone just pointed me to DropScript, which pretty much does the same thing appify does, except it's an actual OS X application.

John Lannon wrote on 12th November 2010 at 21:03:

Not sure if it's actively maintained, but Platypus accomplishes is another similar utility. Like DropScript, it's an actual OS X app.

Thomas Aylott wrote on 12th November 2010 at 22:44:

Platypus actually does something different. It sets the executable to some sort of binary and the includes the script as a resource. It's much slower to load than just a script file in 3 folders.

richtaur wrote on 13th November 2010 at 02:58:

Sweet, I might see if I can make some games with this :D

Mathias wrote on 17th November 2010 at 12:33:

If you've been looking for an easy, automated way to update to the latest Chromium nightly build, look no more!

I just found this script, saved it as `chromium-updater.sh`, then entered the following command:

```
appify chromium-updater.sh "Chromium Updater"
```

Et voilà! Now, all it takes for me to update Chromium is a single click.

Of course, this app won't be able to give any real visual feedback as to what it's doing, but if it exits almost immediately it means you already have the latest version installed. If it takes more than a few seconds, you probably don't and the app is downloading the latest version for ya. (That, or you have a *very* slow connection, or chromium.org is

down.)

If you're too lazy to build it yourself, you can just download the Chromium Updater app here, complete with a nice icon and everything.

Kevin Grant wrote on 18th November 2010 at 14:09:

There is a simpler way to make scripts openable in the Finder, which is to give them the `.command` extension (`.tool` also works, I think). This automatically makes them open in a terminal window, which is both more and less convenient, depending on whether or not the script fails. :)

Manuel Villegas wrote on 11th January 2011 at 17:48:

Hi, I was pretty seduced by the idea of making Apps out of X11 applications like `xdvi`, then associating them with `.dvi` files in Finder. But I somehow can not make it work. My simple write is:

```
/Applications/Utilities/X11.app/Contents/MacOS/X11 xdvi
```

…which works when typed in Terminal, but once the app is created, it does nothing. Can somebody help me out? I also downloaded GIMP in the X11 frame, and I don't get it to have it as a standalone app. I know, for this there is already a Mac Bundle. But for the other X11 applications?

Mathias wrote on 11th January 2011 at 19:50:

Manuel: You may wanna try DropScript or Platypus, which are both suggestions from previous comments.

shardbearer wrote on 24th January 2012 at 23:43:

I tried this, it opened, didn't do anything, and immediately stopped responding. Eventually used Automator.

Mathias wrote on 25th January 2012 at 08:00:

shardbearer: What was the script that you appified doing exactly? Share your code!

Hawk wrote on 27th January 2012 at 12:06:

I tried this, installed it as instructed. But when I did `appify test.sh` it said "`mkdir` — permission denied". So I did it with `sudo`, but it didn't create a `.app` package. I tried again, but this time it said "`test.app` already exists"… :(What's going on? Can't figure it out.

brontosaurusrex wrote on 28th January 2012 at 18:45:

Would an app like that take dropped files as arguments for embedded sh/bash script?

Would an app like that auto-launch Terminal to see what's going on or not?

Ken wrote on 31st January 2012 at 22:38:

This appears to be broken. Maybe it doesn't work with OSX Lion?

In fact, it looks like none of these work anymore… Platypus, Dropscript, Appify… I think Lion broke them all. Even the `.command` trick no longer works, all that does is open the Terminal without actually running the script.

Mathias wrote on 31st January 2012 at 23:03:

Thomas created an updated version of Appify that adds OS X 10.7 Lion support by also creating an `Info.plist` file: https://github.com/subtleGradient/tilde-bin/blob/master/appify

There's a UI for Appify, too: https://github.com/subtleGradient/Appify-UI

Romeo wrote on 4th February 2012 at 21:56:

Since I use Vim for almost all my files, this kind of approach is quite nice. Is there a method that also assigns icons to the files? Different icons for different extensions would be great. I don't like it when all the files have the generic white-sheet-of-paper icon.

Mathias wrote on 5th February 2012 at 15:34:

Romeo: See the "Adding a custom app icon" section of the post. I don't know of a way to automate this, though.

Eric Larour wrote on 26th February 2012 at 07:41:

I've been trying the new Appify from Thomas to add Lion support. I'm running on Mac OS X 10.7.2.

My script is a simple:

```
#!/bin/bash
echo "hello"
```

When I appify it,

```
appify script.sh "SCRIPT"
```

I get the following:

> You can't open the application SCRIPT because PowerPC applications are no longer supported.

I get nothing from the `system.log`. Sounds like Mac did something else in the newer versions that breaks the script.

Any feedback welcome,

Eric L.

Hawk wrote on 16th March 2012 at 16:41:

I can't do this. I did `mkdir -p ~/Desktop/MyTest.app/Contents/MacOS` and saved a `MyTest` bash script into that folder using `nano`. Then `chmod +x` on both the script and the `.app` directory. The icon is that "invalid app" icon and when I try to launch it, Finder tells me:

> You can't open the application "MyTest" because it's not supported on this type of Mac.

And then, a split second later:

> You can't open the application MyTest because it is not supported on this type of Mac.

(No quotation marks, and "it is" instead of "it's".)

I'm using OS X 10.6.8.

Don wrote on 29th March 2012 at 01:22:

I appreciate the ability to copy & paste a custom icon into the selected icon in the new application's "Get Info" dialog box. But for deployment reasons, I would like to add a `.icns` icon to the package, and know that it's there by seeing it in the application's Resources folder and see an entry for it in the application's `Contents/Info.plist` file. I've tried adding lines like:

```
<key>CFBundleIconFile</key>
<string>iconfile</string>
```

…after the last CFBundle statement in `Info.plist` file, where `iconfile` is the name of the `.icns` icon residing in the Resources folder that I created after the `appify.bash` run, and into which I copied the `.icns` icon file. Yet I just don't seem to be able to make this icon show up in the folder-browser displaying the new application that the newest rev of `applify.bash` had created. The app runs correctly, but the icon is wrong. I've tried copying the application to a new folder, and in this new folder, the displayed icon is still wrong in the folder browser. The icon itself seems to be in good shape; it has all lower-case letters in its name, and the reference for it in `Info.plist` is thought to be correct. I've also tested specifying the filename with or without the icon's `.icns` extension.

At this point, I'm out of ideas. I'm probably missing something obvious, but I don't know what. How does one add a nice `.icns` icon to a package that the newest rev of `applify.bash` has just created? Is there some place to look for error messages regarding problems in reading the `Info.plist` file?

Thanks in advance for your help!

Daniel wrote on 7th May 2012 at 21:19:

It appears that the script needs to end in `.command` to work in Lion. This removes the "You can't open… type of Mac" errors.

Anonymous wrote on 26th August 2012 at 16:35:

After following this post, I was unable to run the .app after using the command `appify your-shell-script.sh "Your App Name"` in Terminal, using:

```
#!
cd ~/Projects/Foo/
python -m SimpleHTTPServer 8080 &> /dev/null
```

```
open http://localhost:8080/
```

I receive a prompt saying it was damaged or incomplete.

Mathias wrote on 27th August 2012 at 09:39:

Anonymous: See comment 15.

asdf wrote on 15th February 2013 at 12:54:

How do you stop it from bouncing? I have a command that just starts an SSH tunnel and keeps it up. When I start it, it keeps bouncing for like 30 seconds, then when I Ctrl-click it says “Program not responding” and lets me “force quit”. I’d like to tell OS X that it *has* started up fine, and it can kill it in the normal way if it wants to.

David wrote on 3rd August 2013 at 03:10:

Thank you! This is exactly what I was looking for! Now I can write scripts for my users to do complicated and/or repetitive tasks, and they can use as a regular app.

Gino wrote on 18th October 2013 at 13:59:

This “appify” is quite useless… It’s far easier (and gives fewer problems with scripts) to follow this procedure:

1. Create your shell script (ex `command.sh`)
2. Give it execution permissions (`chmod 755 ./command.sh`)
3. Get Info → Open with: → (associate the Terminal app)
4. Get Info → (put the icon you want on it)

Kind Regards,
Gino

Kwyll wrote on 21st December 2013 at 13:27:

Wow, neatest little trick I found in years! Solved my problems with Eclipse on OS X 10.9 in a breeze…

David wrote on 1st March 2014 at 18:26:

The problem with Gino's solution is that you won't be able to add your script to the Dock. This is why it won't do for me as I want it to be available from there and not my cluttered desktop. Though, it might work for others.

David wrote on 3rd March 2014 at 04:22:

Just found a problem with this app – it doesn't run `rsync` properly via double click whereas it works fine from a Terminal window. Any idea why?

Joran wrote on 24th April 2014 at 14:51:

David: David, found the same problem as you, any solution?

D3@TH wrote on 16th June 2014 at 06:59:

I need help with this. I'm trying to make an app using the `.sh` extension to run a command like `git clone; git submodule update --init; ./make.sh; cp -f` but it's not working.

mahboud wrote on 16th June 2014 at 12:00:

Use Automator to turn a shell script into an app. That way, the app can also act as a droplet: drag files onto it and the files will appear to the script as arguments, and are accessible via `$1`, `$2`, `$3` (or loop through them with `for f in "$@"`).

Don't waste your time with Appify or Platypus — they are either obsolete, bad-tempered or just not as intuitive and powerful as Automator.

Jason wrote on 10th September 2014 at 23:22:

Don: It might be because you're only subbing in one file size that is accepted, and a part of the `icns` package. To create a proper `icns` file, you can use the `iconutl -c icns <iconset package>` command, where the `<iconset package>` is the path to the folder holding the different accepted dimensions of the icon. You can also build in Xcode and have it auto-created. Not sure that's the problem here, but off the top of my head it's worth a look.

Patrick wrote on 18th September 2014 at 18:44:

I created an app as described and although I can place it in the Dock it won't show up in Launchpad (and I can not add it, I tried many ways). Any ideas why?

Alexandru Barbat wrote on 31st October 2014 at 17:19:

Did you copy the bundle to `/Applications`?

Dan wrote on 18th January 2015 at 06:18:

This works wonderfully for me on OS X Yosemite. I copied it to a script, put it into `/usr/local/bin` (where other stuff is, in my path) and ran it against a script I run for a Wine application. It created a directory that emulates a normal OS X app, so afterwards you can drag it to wherever you want. I put it into my `/Applications` folder and now I can run my Wine app like every other application.

Kevin VanStrien wrote on 10th February 2015 at 17:41:

Eric Larour:

Did you ever find a solution for this? I get the PowerPC issue as well. I am trying to make a Java program that will dynamically create a shell scripts and write it out into this structure to make am easily launch-able app icon for users. I can't use a tool like Automator, Platypus, etc either since it needs to be written out with simple Java code. It is essentially a shell script that will call javaws to launch our application for many different deployments out there (different URLS, etc) - just want an easily double clickable app for the users.

I also don't like the solution of making a *.command file...while that works - hard in Java code to add an icon (Mac OS X specific resource bundle stuff), and hide the extension on the file name. Also it leaves the Terminal running, which would cause confusion for my users.

If anyone has an idea...please let me know! I know it's kind of a crazy "requirement". :)

Kevin VanStrien wrote on 10th February 2015 at 21:15:

Kevin VanStrien: Nevermind - got it to work! I think I now needed a valid Info.plist file, and make the script executable underneath the MacOS folder.

Leave a comment

Name *

Email *

Website

Your input will be parsed as Markdown.

Add your comment