How Do I Give My Program an Icon?

This month I'll try to answer the question: "How do I give my program an icon that will show up on the desktop?" Unfortunately, this is one of the more complicated tricks in Mac programming. I hope I can make it fairly clear with the aid of several screen dumps.

What's Going On?

Every Macintosh file has two four character codes called the file's 'type' and 'creator'. Some people refer to the creator as the file's 'signature', and the verbose may say "creator signiture" which sounds slightly theological. Programs use the type to determine whether they can open the file. The creator links applications with the documents that they create. For example, all word processors, most communication programs, and several spreadsheets can create TEXT type files. If a program can read TEXT files at all, then every TEXT file will show up in the minifinder window when the user chooses 'Open' from the File menu, but TEXT files may have different icons on the desktop and launch different programs if they are double-clicked. How a file is displayed and what program it launches are determined by its creator.

Desktop icons are complicated because of the complex interaction between the Finder, applications and documents. Most Mac applications contain information for the Finder on their own icon and icons for the documents that they create. It would take to long for the Finder to scan all mounted volumes to find the right icon for each file every time it draws the desktop. After all, a volume can be an 80 meg hard disk. So Finder maintains an invisible file, called DeskTop, in the root directory of every volume with information on how to show the files on the volume. The DeskTop file contains a special resource, the autograph, for every creator represented on the volume. The autograph stands for the application in the DeskTop file.

We want our program to have its own icon, so we must provide an autograph resource in its program file. In fact, if our program created any documents, we would need to make the autograph anyway. If we didn't, users would get an "An application can't be found…" message when they tried to open our program's documents. The resource type of the autograph resource must be the same as the creator of our application. Finder only coppies the autograph resource from the program file to the DeskTop file; it never opens it. The resource data can be anything at all, but it is conventional to make it a string such as "MyNewApplication version 1.0 Oct. 18, 1987." For this reason, the autograph resource is sometimes called the 'version resource.'

The next resource type we need is the icon for the application itself as well as one for any document type that it creates. First lets clear up a vocabulary issue. What is loosely called an icon on the desktop is actually an ICN# (pronounced "icon list"). An ICON is a simpler thing that doesn't have a selected and an unselected state. An ICN# can contain any number of of ICONs but the ones the Finder uses have only two. The first ICON is called the data ICON and the second is the mask. The way that the icon is displayed on the desktop depends on the background pattern underlying the icon as well as whether or not it is selected. When the icon is in its standard, unselected, state each bit is on or black if it satisifies Data OR Background AND NOT Mask. When the icon is selected, the formula is (Data XOR Mask) OR Background AND NOT Mask. Don't worry if this sounds confusing; ResEdit makes it very visually intuative.

Since an application may specify its own icon as well as icons for each type of file that it creates, the Finder must have a way of matching icons to file types. An FREF, pronounced 'File Reference' is a little two entry resource that associates the local resource ID of an ICN# with a file type. We need one FREF for each ICN#.

Notice that the preceeding paragraph contained the phrase "local resource ID." When you copy a program from on volume to another the Finder coppies not only the autograph resource, but, assuming the bundle bit is set, all of the other finder related resources as well. The problem is the potential for resource ID conflicts in the DeskTop file. You may have given your program's own ICN# a resource ID of 200, but Finder discovers that there is already an ICN# with ID 200 in the file. The solution is the bundle resource, type BNDL, that bundles all the Finder resources in your program together. Finder patches the copy of this resource in the DeskTop file to avoid resource ID conflicts. Lets say, for example, that your program has two ICN#s one for the program itself and one for TEXT files that it creates. Assume that the resource IDs for these programs are 200 and 201. You would also need two FREFs and, for convenience, you might as well also give these ID numbers 200 and 201. Inside the FREFs, however, you could say that the ICN# for file type APPL is 0 and the one for TEXT files is 1. Here the 1 and 0 are local resource IDs. The BNDL ties the resources together. It says, in effect, this application's creator is MyAp: it has two ICN#s whose local IDs are 0 and 1 and whose actual IDs are 200 and 201; furthermore, it has two FREFs whose local IDs are 0 and 1 and whose actual IDs are 200 and 201. When Finder coppies these resources to the DeskTop file it changes the actual IDs and patches the BNDL.

May I Have Your Autograph?

First, let's make an autograph resource. For this job, as well as for any programming task having to do with resources, use ResEdit. I was surprised to learn that some people are trying to make do with RMaker simply because it came with their developement system and was documented in their manual. Some time ago I asked Palo Alto Shipping Company, the folks who make Mach2, why they ship RMaker, rather than ResEdit, with their product. They said that they were influenced somewhat by the fact that ResEdit is so big that they would have had to include another disk. Another factor in their decision was that RMaker is pretty much frozen. When they made the decision, Apple was releasing a new version of ResEdit every few months and they didn't want their manual to be out of date when it came back from the printer. The most important consideration was that RMaker is text based, which makes it easier to discuss in the manual and much easier to support on their SIG. Programmers can include a portion of their RMaker input file in their message if they are having trouble. When Palo Alto Shipping company programs, they use ResEdit.

The conclusion seems pretty inescapible: RMaker is for **talking** about programming. ResEdit is for **programming**.
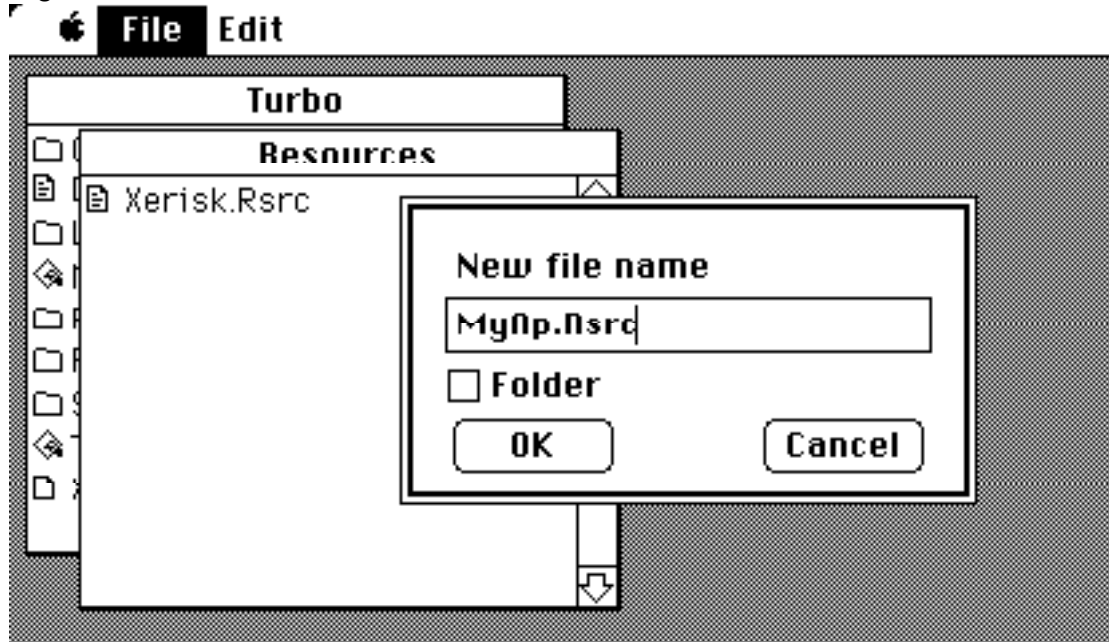
First, pick a four character sequence to be your program's signature. You can use any four characters you want so long as they aren't ????, the conventional signature for all applications that don't have icons, and as long as they don't already belongs to another program or to one of the standard resource types. Upper and lower case letters are unique. When you go commercial you should register your program's signiture by writing to:

> Macintosh Technical Support
> Mail Stop 3-T
> Apple Computer, Inc.
> 20525 Mariani Avenue
> Cupertino, CA 95014

In the mean time, stay away from things like MPNT, WORD and MWRT. Does this requirement for a unique signature mean that there is a limit to the number of possible Macintosh applications? I'm afraid that it does. I wouldn't worry yet, but once we have about 4.3 billion applications we're going to be in serious trouble.

Start up ResEdit, click on windows and the mini folder icons until the folder where you would like the file to be placed is in the frontmost window and choose 'New' from the File menu. Figure 1 shows how you create a new file in ResEdit.
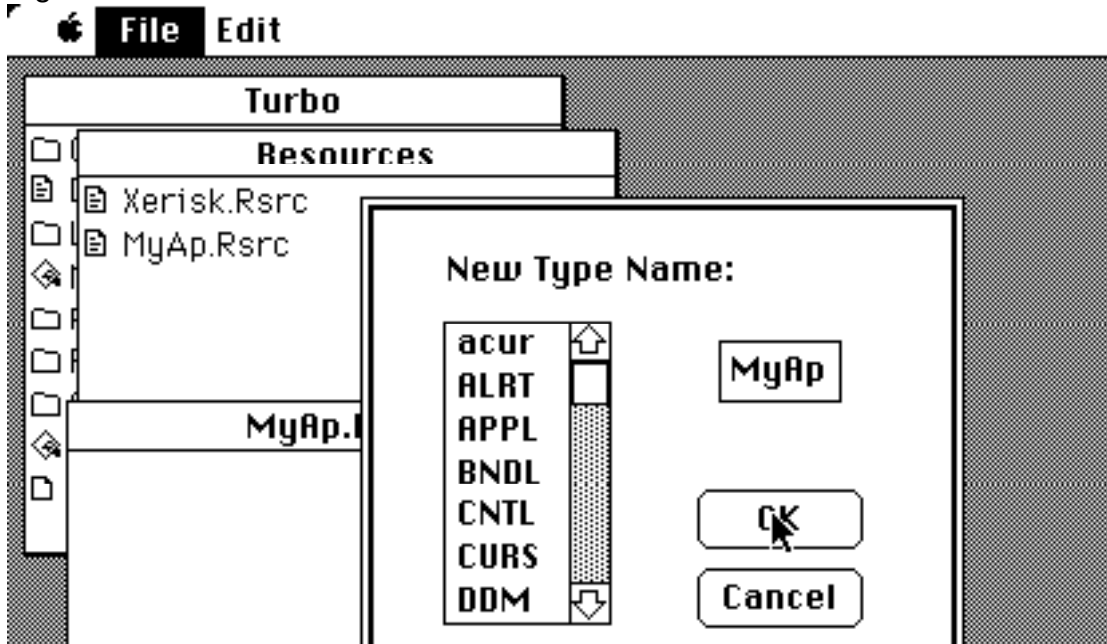
Figure 1

Some developement systems require you to give this file a specific name. For example, all of the resources in Mach.Rsrc are coppied into your application when you turnkey it. Lightspeed C uses a similar system but the file is named <Project Name>.Rsrc. In other systems, you can name this file anything you want. Turbo lets you specify a file from which to copy resources with a {$R FileName } compiler directive. McAssembly is the most flexible: you can include as many resource files as you want with % linker directives.
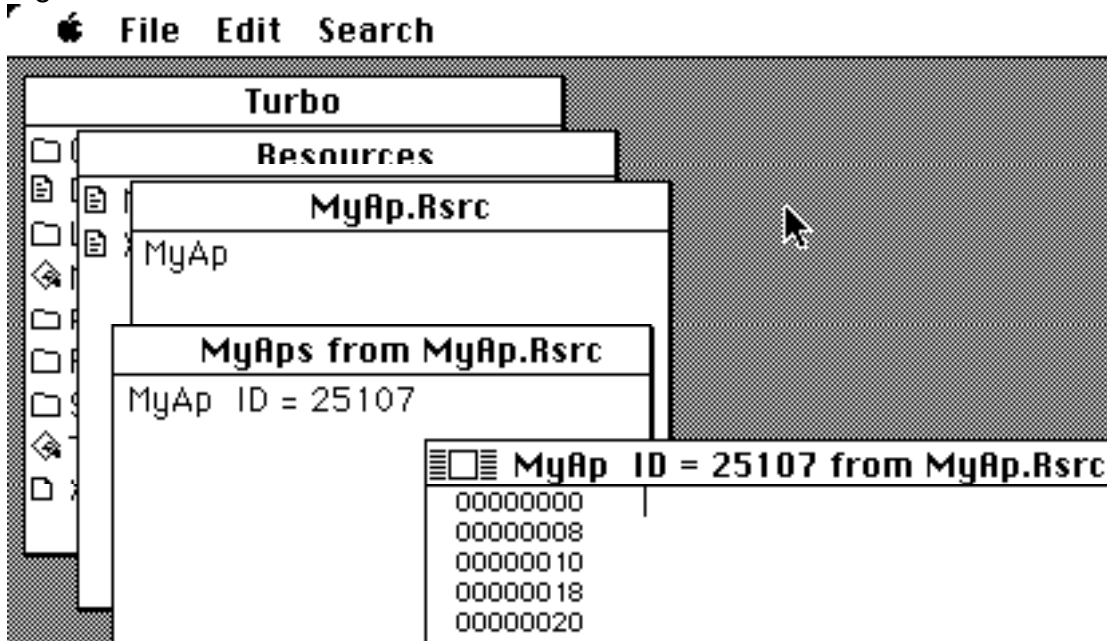
For illustrative purposes, let's assume that we have chosen MyAp as our program's signiture. With the MyAp.Rsrc window in front, again chose new from the File menu. We'll see something like Figure 2. Notice how 'New' is context sensitive in ResEdit. If a volume or folder window is frontmost, 'New' will let you create a new file for folder. If a file window is frontmost, 'New' creates a new resource type. The dialog box presents a list of standard resource types but we want to create an entirely new type so we just type in its four characters.
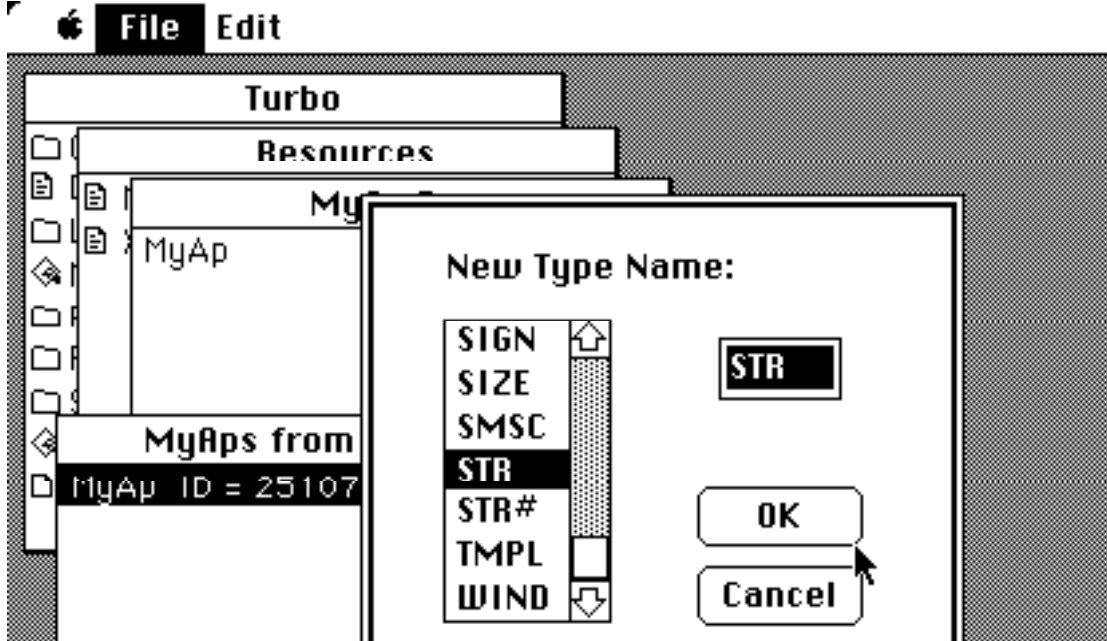
Figure 2

When we hit the OK button, ResEdit creates the new resource type MyAp and conveniently opens its window. Choose 'New' yet again to get a display like Figure 3. ResEdit has created a new, empty, MyAp resource and opened the default resource editor, a hex-ASCII editor. ResEdit has assigned this new resource a random ID number. This is another demonstration of ResEdit's sensivity to context. With a resource type window frontmost the 'New' menu choice produces a new resource of that type.
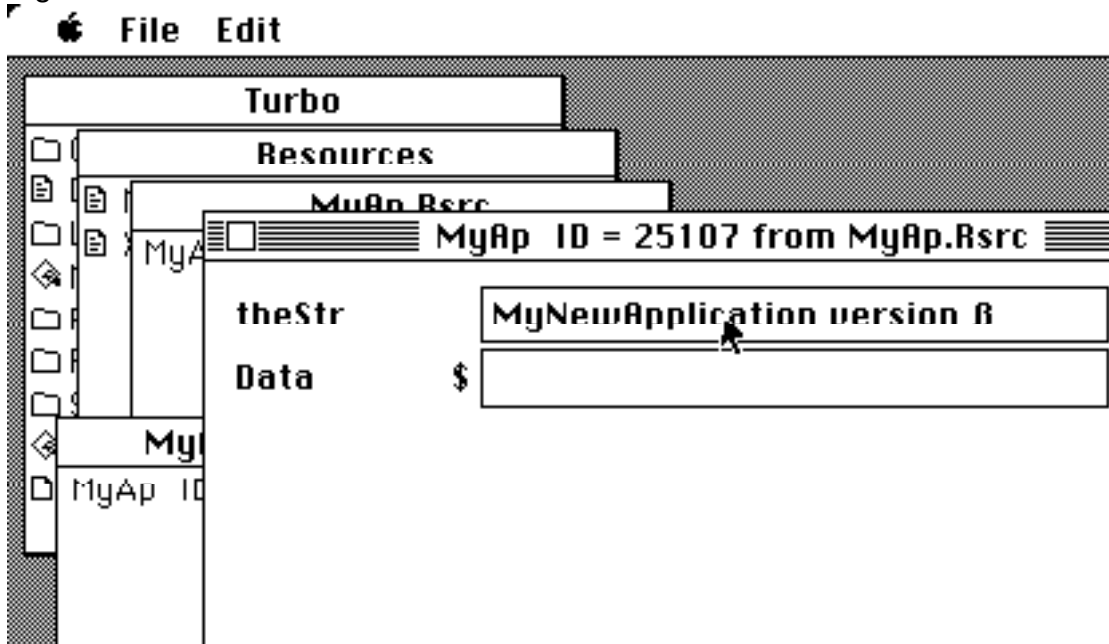
Figure 3

Editing in hex is a pain. Fortunately, we can use ResEdit's resource type-casting feature.
Close the MyAp window and with the MyAp still selected in the resource type window choose
'Open As...' from the file menu. Your screen will look something like figure 4. Choose STR
(string) from the scrolling area in the dialog and click OK.

Figure 4

Your screen should look like figure 5. Type in a string describing your program. You can
type more characters than the window shows, up to 255. Ignore the 'Data' box. Finder never
opens your autograph resource but other programs, such as version verifiers, may, so putting
a short description of your program's name, version number, and perhaps compile date here
is a courtesy to other programmers. As a further courtesy, you should follow the convention
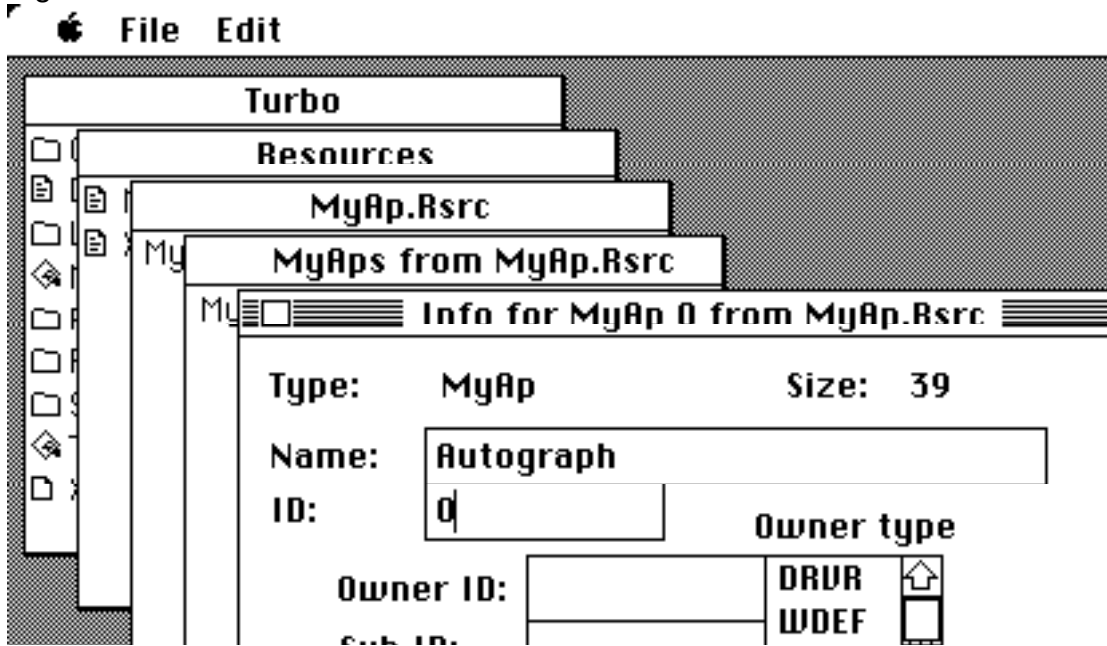of setting the autograph's resource ID to 0.

Figure 5

     After closing the dialog box where you entered the string, and with the MyAp resource still selected in the resource type window, choose 'Get Info…' from the File menu. In ResEdit 'Get Info…' is a somewhat of a misnomer: it could more appropriately be called 'Set Info….' You will get a dialog box like Figure 6. This dialog box alows you to set the resource ID of the selected resource. I think is it also a good idea to take the opportunity to name resources in this dialog.

     Congratulations! If you have followed this article so far, you have now succeeded in giving your application an autograph resource. It is a good time to save your work. Click on the close box of the file window, the one named MyAp.Rsrc in our example. ResEdit will ask you if you want to save changes. You bet you do. Treat yourself to a beer: You deserve it.
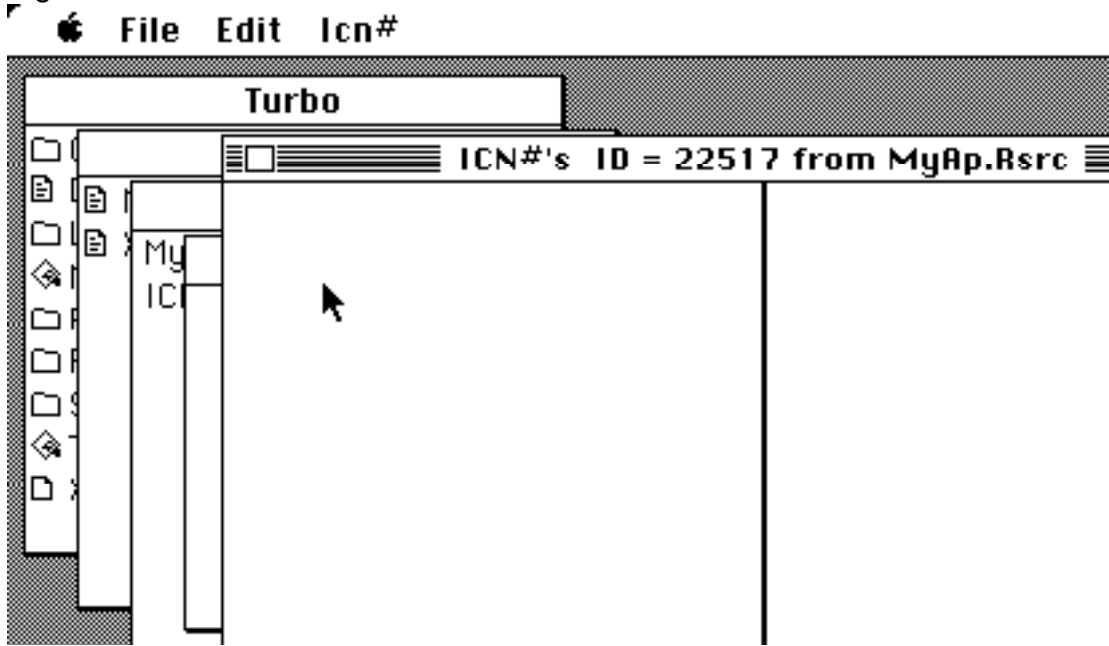
Figure 6

Can You Picture That?

This is the fun part. There are several public domain and shareware ICN# editors available but the one built into ResEdit is intuitive and reasonably powerful as long as you don't need special effects. If you have ever used FatBits in a Mac painting program then you already know most of what you need for creating an ICN#. If you haven't, stop right now. You shouldn't be programming on this machine till youv'e messed around with MacPaint for an hour or so. You don't understand the user interface yet. What! You only run your Mac II under UNIX and you don't care about the interface, but you thought those icon thingees were kinda neat. Well then, soldier on.
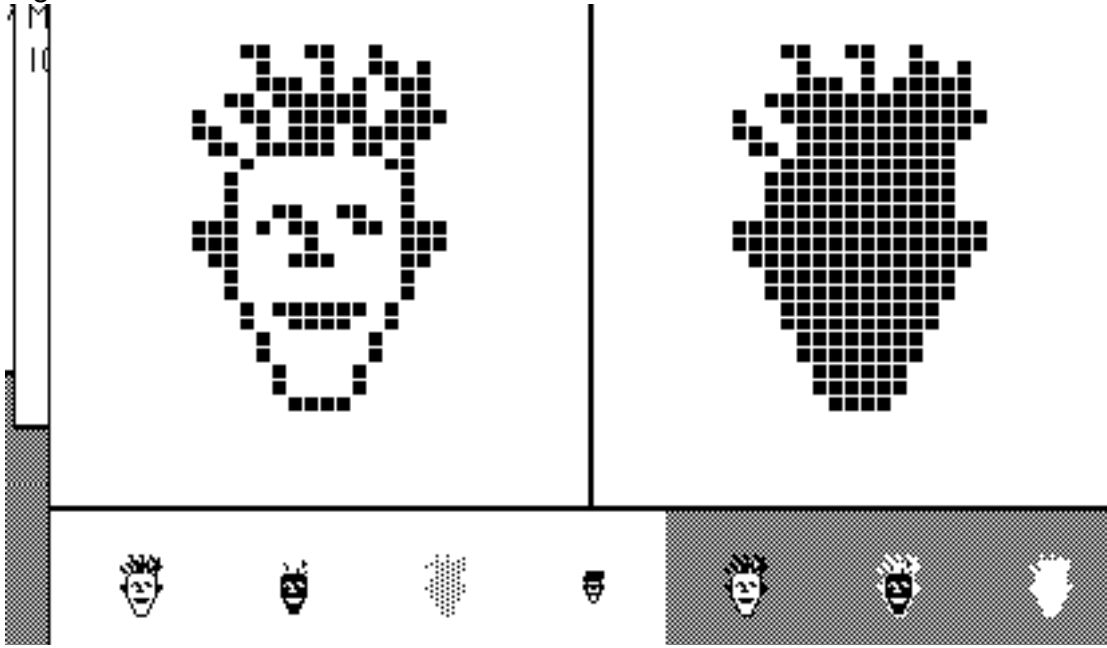
Repeat the process of double clicking on the file to open a window into it, and choosing 'New' from the File menu. This time choose ICN#. Now choose 'New' again to create a new ICN#. The screen should look like Figure 7.

8

Figure 7

Just start drawing on the left half of the window. If you mousedown on a white pixel, the cursor becomes a drawing instrument for as long as you hold down the button. If you start on a black pixel, it erases. Every so often choose Data -> Mask from the Icn# menu to make a standard mask in the right half of the window. Notice in Figure 8 how ResEdit gives you lots of visual feedback. The bottom of the editing window shows how the icon will look on the desktop. From left to right, the views are unselected, selected, when the disk is ejected, and when then the 'View by Small Icon' mode is chosen in Finder's View menu. Then the selected, unselected and ejected modes are shown against a gray background. These actual size views are updated only when the mouse button is up so draw or erase a little and then lift you finger off the mouse button to see the effects of your changes.

Figure 8

The Data -> Mask choice in the Icn# menu makes a standard mask. Enlarge the mask or erase holes in it to create special effects. Special effect icons only look good against a white background so few Macintosh programs use them. Nevertheless, it's fun to play around with ResEdit's ICN# editor just to see what effects you can create. You can always standardize your icon by choosing Data -> Mask again.
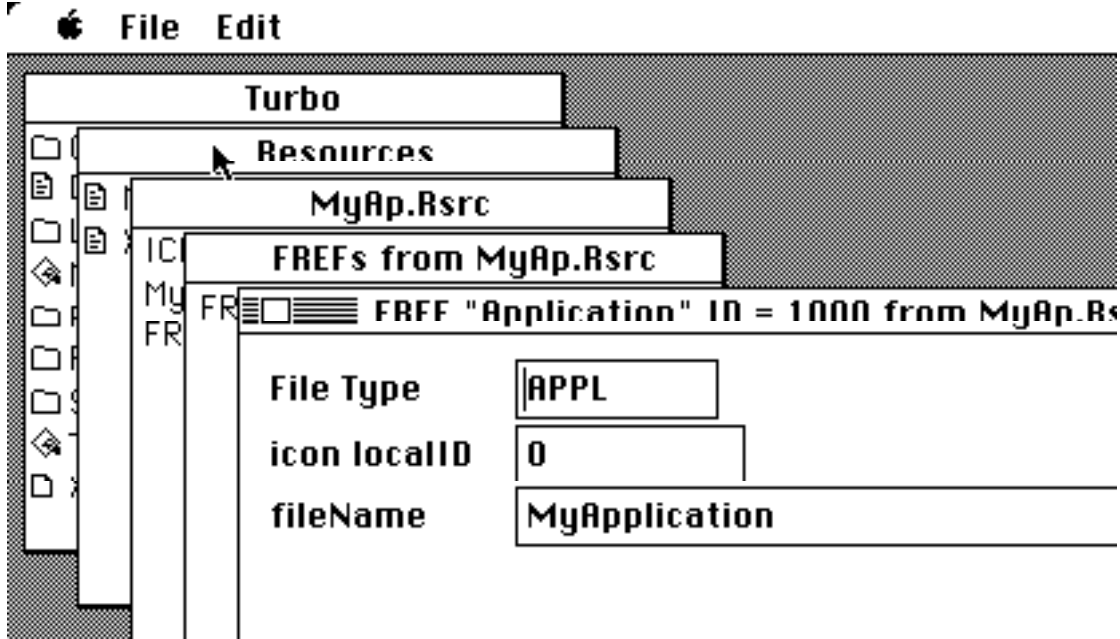
When you are happy with your ICN#, click the close box and, with the ICN# selected in the resource type window, use ResEdit's 'Get Info…' menu choice to name it and change its ID. Pick an ID number greater than 127. Don't use negative ID numbers for finder resources either. Apple has reserved all resource numbers less than 128. (This doesn't apply to resource types, like your autograph resource, that you create youself.) For this example, assume that the ID of the ICN# resource is 1000.

At this point it's a good idea to save the ICN# by closing your resource file again.

Include it by Reference

Now you are ready to make your FREF. Follow the same procedure outlined that you used to make the autograph and ICN# resources except this time choose FREF from the resource type dialog. Figure 9 shows the FREF editor. We are assigning the icon to the application itself so the file type is APPL. If, for example, our application created TEXT documents and we had made an ICN# for those we would create another FREF that might associate the File Type TEXT with local ID 1. You can pick any local ID number as long as the FREF and the BNDL are consistent. The filename field is just a comment, you may leave it blank. After closing the FREF editor window, it's a good idea to use the 'Get Info…' menu choice to give the FREF the same resource ID as the ICN# resource it references. This makes it easier to keep the resource IDs straight in the BNDL resource.
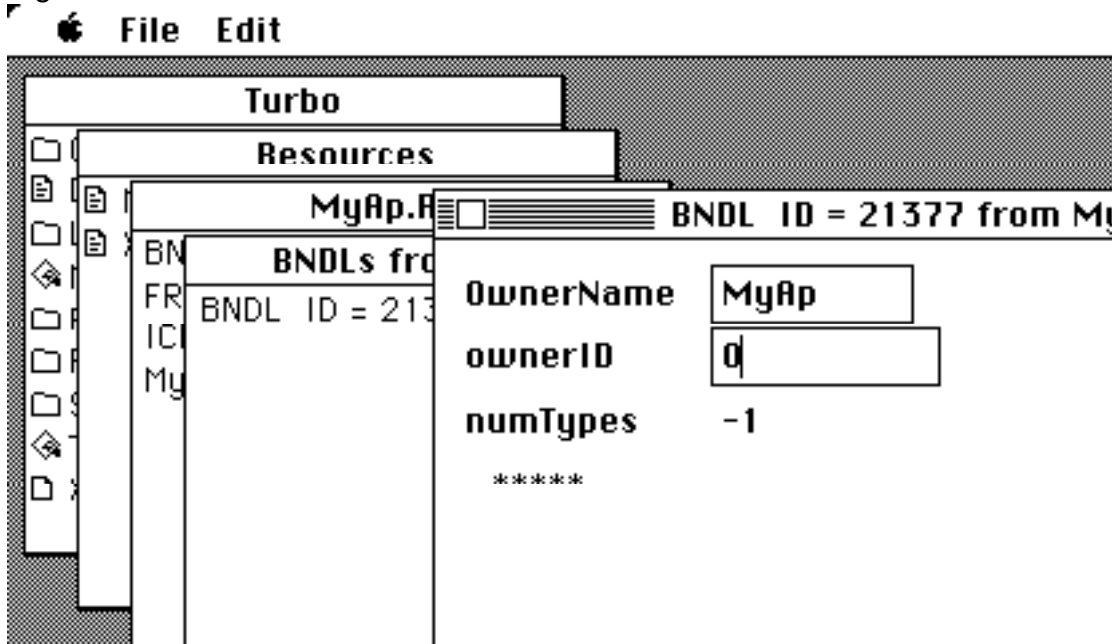
Figure 9

Bundle Up Your Baby

Now we are ready to make the last finder resource, the BNDL. I won't bore you with the details of creating a new resource. The BNDL editing dialog does have a feature that we haven't seen before. Figure 10 shows the editing as it appears when you create a new BNDL resource. Only the type and signature of the autograph resource have been filled in. Obviously there needs to be more in the BNDL but it may not be clear how to add it.

Figure 10

The answer is to select the row of asterisks at the bottom of the window and choose 'New' from the File menu as illustrated in Figure 11. The editing window will expand, allowing you to enter more data.
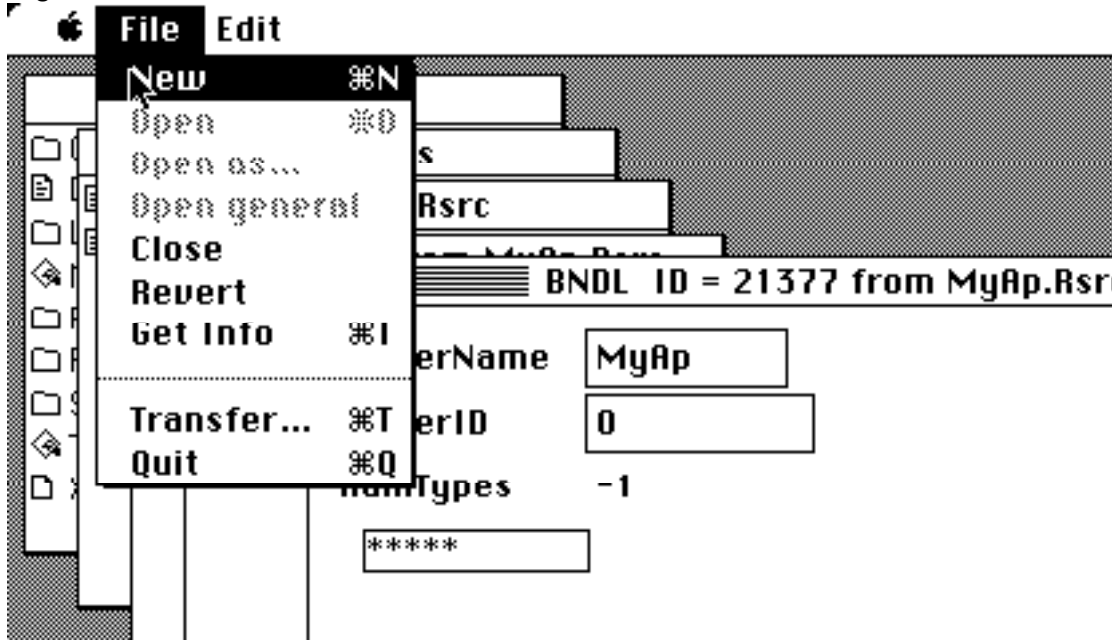
Figure 11

Figure 12 show the screen after the 'New' menu choice when the row of asterisks was selected. Here the first type, ICN#, has been filled in. Order is immaterial, just make sure that you reference both your ICN#s and your FREFs in the Bundle. Note that when we add a type, the number of types that ResEdit maintains for us goes from -1 to 0. The '# of type' field for ICN#s is still -1 in Figure 11 because we haven't yet added any references to specific

ICN#s. These numbers will always seem to be one less than what they should be. Don't worry about it; lots of resources work this way. For the curious, the MC68000 looping instuctions all terminate when the index register gets to -1, not zero. The resource manager takes advantage of this by using 'one less than' numbers.

How do we add the information about our ICN# to this bundle? Those who guessed 'By selecting the row of dashes and choosing 'New' from the File menu,' go to the head of the class.
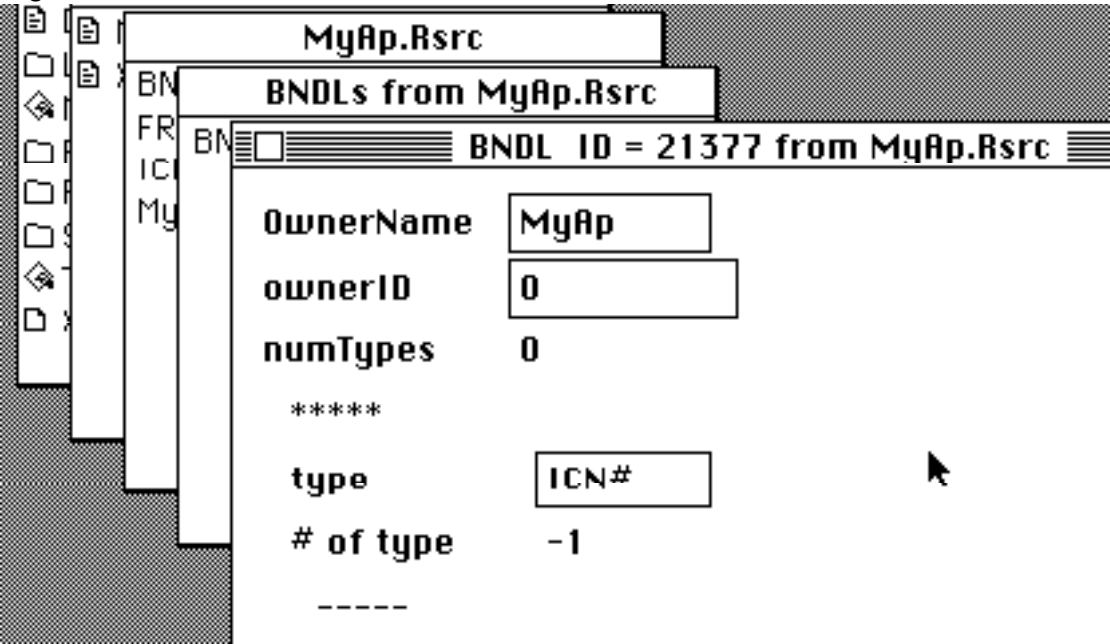
Figure 12

Figure 13 shows the BNDL editing dialog after we have entered the information for our ICN#. If we had additional ICN#, we would just continue selecting a row of dashes and chosing 'New' until we had space to enter them all. Note that the 'localID' is the ICN#s local ID as defined in the corresponding FREF while the rsrcID is just the ID of the ICN# that we set with the 'Get Info…' menu choice.

Figure 13

When the ICN# information has been typed in, select a row of asterisks and choose 'New' from the File menu again. Now type FREF in the type box, select the row of dashes and choose 'New' again. Set the localID of the FREF to 0 (or whaterver you like) and the resource ID to 1000 (or whatever the you set as the resource ID of FREF with the 'Get Info…' menu choice).   Figure 14 shows an example. If your application has more than one FREF, keep adding references to them in the BNDL by selecting a row of dashes and choosing 'New.'

Figure 14

When you finish adding the reference to the FREF in the BNDL, close the BNDL editing window, set the resource ID of the BNDL to something reasonable and quit ResEdit. Make sure you click OK when ResEdit asks if you want to save changes. Your application now has a complete set of Finder resources. Have a brew. Hell, have a six-pack.
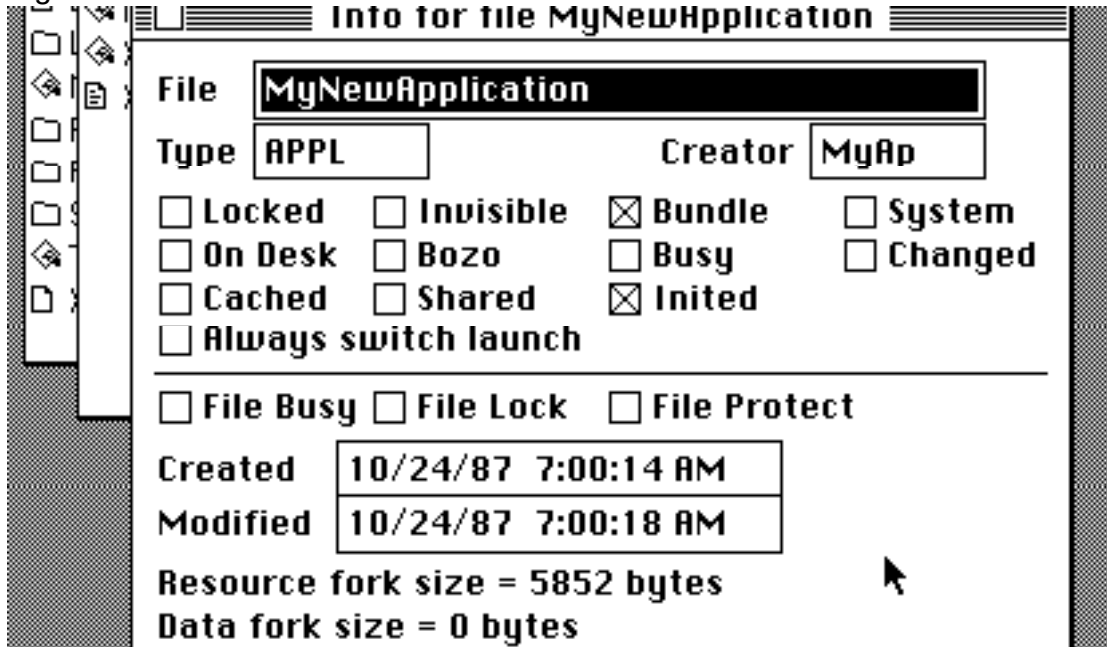
Huh, It Didn't Work. What's Wrong?
Even if you have followed this article perfectly, your applcation may appear not to have its own icon. This problem can also manifest itself when you change its icon by editing the ICN# and the changes don't show up on the desktop. One or more of three things have gone wrong:

1   The resources aren't being coppied from the resource file into your application. Compile your application to disk. Launch ResEdit and open your application. If your Finder resources were coppied into the application go on to problem 2 If they weren't check the documentation that came with your developement system. Maybe you are forgetting a compiler directive or the resource file that you are trying to copy from has the wrong name. Maybe it's hidden in some folder where the compiler can't find it.

2   The bundle bit isn't set or the files creator doesn't match the autograph. The finder won't even look in a file for finder resources unless that file's bundle bit is set. Even if the bundle bit is set, nothing will work unless the file's creator matches the resource type of the autograph resource exactly (case counts). In ResEdit, bring the volume window to the front and select your application without opening it. Now choose 'Get Info…' from the File menu. You will get a window like that shown in Figure 15. Here you can check on the state of the bundle bit and the file's creator. You can also change the creator and the state of the bundle bit in this window. If you recompile your application, however, the

state of the bundle bit and the application's creator bytes will be reset by your compiler so check the documentation of your development system if you have this problem

Figure 15

Info for file MyNewApplication

File MyNewApplication

Type APPL   Creator MyAp

☐ Locked   ☐ Invisible   ☒ Bundle   ☐ System
☐ On Desk   ☐ Bozo   ☐ Busy   ☐ Changed
☐ Cached   ☐ Shared   ☒ Inited
☐ Always switch launch

☐ File Busy ☐ File Lock   ☐ File Protect

Created   10/24/87 7:00:14 AM
Modified 10/24/87 7:00:18 AM

Resource fork size = 5852 bytes
Data fork size = 0 bytes

3   If the finder resources in the application are OK and the bundle bit and creator check out but your icon still isn't right on the desktop, then the problem is with the DeskTop file itself. It's pretty easy for Finder to be convinced that the DeskTop file doesn't need updating even though you have changed your application's Finder resources. You will have to rebuild the DeskTop file on the volume with your application. If your application is on a hard disk, read the following carefully:

ARE YOU CRAZY? DON'T YOU KNOW THAT USING RESEDIT TO MODIFY A FILE ON A HARD DISK CAN BREAK YOUR COMPUTER? GET THAT APPLICATION ON A FLOPPY RIGHT NOW!!

First unmount the disk containing your application. A disk is unmounted when its icon doesn't show up on the desktop in any form. Just ejecting a disk isn't enough to unmount it. If it isn't the startup disk, you can unmount it by trowing it in the trash. The easitst way to unmount a startup disk is to shut down or restart the Mac. Now remount the disk, i.e. stick it in a drive, while holding down the command and option keys. You'll get a dialog asking if you want to rebuild the DeskTop file. Click OK. Your application, resplendent in its brand new icon, will show up on the desktop. If you havn't polished off that six-pack, do so now.