Introduction

When Kaleidoscope was born, it was thought that people would enjoy using the included schemes and that the Kaleidoscope team would create a few more bundled schemes as time went along. However, the initial file format was unintuitive and difficult to edit, even for those involved in creating the bundled schemes. A new, flexible format was needed: one that would allow changes to the shapes of objects and do more than just provide alterations to the colors in the Apple platinum interface. Consequently, a simpler, cicn-based file format was created which resulted in schemes which were much easier to edit. Subsequently, the decision was made to open the floodgates and provide support for third-party designs. Today, some two years later, there are well over 3,000 different varieties of costume for the Mac OS. Although some 600 people have been brave enough to distribute their creations to the world, at least that many have written, pleading with me to unleash the mystery of scheme creation. Until recently, the answer has been quite simple: download a resource editing application and one of the many manuals from the page of utilities at the Kaleidoscope Scheme Archive and go to town. However, this mantra hasn't sufficed in recent times as people try to tackle the new scheme format released with Kaleidoscope 2.0.

It's quite clear why many people are no longer satisfied to work within the confines of the initial Kaleidoscope scheme format. What we view today as the simple task of recoloring your computer was, two years ago, a massive innovation in interface interaction and a quantum leap forward from the plain, slow-evolving standards of the Mac OS. It's time to up the ante. With Kaleidoscope 2.0, the scheme format has been completely renewed to allow for an infinitely more powerful system of interface creation. No longer are Macintosh users bound by the squared confines of a standard computer window; no longer are Macintosh artists bound by the limited canvas space of just yesterday. Kaleidoscope schemes now offer the user a chance to create wild and wonderful new environments, crisp and clean futuristic interfaces or rich and subtle surroundings.

However, a good Kaleidoscope scheme is a very difficult thing to create. This isn't meant to scare anyone away, and certainly isn't a deficiency of Kaleidoscope; rather, this is simply a reflection of the complexity of modern computer interfaces. No longer are you simply choosing colors for a space someone else has designed; out with the concept of color schemes and in with the broader concept of schemes. It's your turn to design the space and everything that goes in it: an entire computer interface. Since this is, of course, a much bigger task, there has been initial confusion about what to do, where to start, exactly *how* to go about dealing with all the new resources that have cropped up with the new Kaleidoscope scheme format. While conjuring up ideas for scheme designs might prove daunting, that part of the process is up to you, the scheme artist. The rest should be made as easy as possible: nobody thinks less of a painter who does not create his own paintbrushes or weave his own canvas and few are self-taught.

So, I've been asked to help create this guide to serve as a helpful how-to document for those of you having difficulty navigating the resources in the new scheme format. It's my hope that you're in for a treat. I've made it as simple and straightforward as I can so that you can concentrate more on the artistic side of scheme creation and less on worrying about what numbers go where and what, exactly, a cinf is. However, the guide does assume at least a basic understanding of resource files and a passing familiarity with how Kaleidoscope works. Anyone who has experience with previous scheme formats will do fine. Be patient, read carefully and make sure to send me the results of your labor so that the rest of the world can share in your success.

Happy scheming...

Eric Reid - Kaleidoscope Evangelist - evangelist@kaleidoscope.net

Chapter 1 • Parts Of A Scheme

Opening a scheme file in a resource editing application shouldn't intimidate anyone but, unfortunately, it does. You're greeted with an entirely new collection of symbols and letters that, for the average computer user, doesn't mean anything. So, before we jump into the process of scheme creation, let's take a minute to demystify the twenty or so resources you will encounter on your journey. There is nothing more difficult than trying to work in a confusing environment and very little you can accomplish without a thorough understanding of what you are trying to create.

After reading this chapter, don't worry if you are still a bit confused about how some of the different resources work. This chapter is simply meant to give you an introduction to the various resources you will be working with and, generally, what you need to change so that Kaleidoscope will display your scheme as you want it to. The documentation that is distributed in the Kaleidoscope package explains, in general terms, what to do with most of the resource types; however, this guide will delve much more deeply into the cinf and wnd# resources so that the new scheme format becomes less of a mystery.

cicn - Color Icon

These are the fun resources, the coloring book pages of a Kaleidoscope scheme. A color icon resource contains three parts: a color version, a black and white◊ version and a mask which indicates which of the pixels in the cicn should be displayed. Note that although the black and white versions of your cicns do not need to be as intricate as the color ones, you should complete these parts to ensure that your scheme will still look unified if the user should need to switch to black and white or if a certain application needs to use black and white resources for some tasks. These graphic sections are where you define the look of your scheme, either by drawing the parts directly in a resource editing application, such as ResEdit, Resourcerer or Designer's Studio, or by pasting in your artwork created in Photoshop or any accessible drawing program. Each cicn resource depicts a certain part, or parts, of your interface design and all of them have to be edited to ensure a completely finished product. If you open a cicn resource from Scherzo! or Antique, you will notice that they have been labeled with names that represent their roles in a scheme. Take some time to read the names and to look at each graphic, discovering what part of the scheme you are looking at and to see how each piece is drawn to fit in with the others. Each resource is numbered so that Kaleidoscope knows exactly where to find each piece of the interface when drawing windows and controls. All the cicn resources work this way. Your job is to create their shapes and colors so that they come together as a new scheme of your own.

cinf - Control Information

The cinf is one of the two brand new resource types that you'll encounter when creating a Kaleidoscope scheme. There is very little information stored in each cinf and that which you have to enter is extremely straightforward. Each cinf contains information about a specific cicn resource which will be used by Kaleidoscope to patch a variety of controls the user interacts with such as a menu, a slider, a push button or a tab. A cinf is paired with the cicn resource with the same ID, and it tells Kaleidoscope how to interpret the cicn when drawing the corresponding object. The cinf also specifies the text color and any background patterns for that control. All you have to do in each cinf is fill in the information that is asked for: only eight simple answers to questions about the matching cicn resource.

clut - Color Look Up Table

Kaleidoscope uses these collections of color swatches to compile the colors for divider lines and group boxes like the ones surrounding the different groups of options in the Kaleidoscope control panel. You must also specify a series of colors for other applications to use when they query the Appearance Manager to tell them how to draw their windows so that they match the Kaleidoscope scheme that is in use. This may sound overwhelming and technical, but since all you have to do is pick colors corresponding to different categories in your scheme, creating correct clut resources is quite simple.

Colr - Color Scheme Information

Completing the Colr resource is a simple survey-type process in which the author pushes a few radio buttons to determine whether or not the scheme follows certain rules: does it have accent colors, should the scroll thumbs be stretched from the center? As long as the questions are understood, the answers are as straightforward as a single mouse click.

crsr - Color Cursor

These small cicn-like resources are simple coloring exercises which allow you to create replacements for the standard Mac OS cursors. Since the cursor is on screen almost all the time the computer is in use, these are very important creatures to take your time with. A cursor which does not effectively stand out from the background can cause numerous problems, including eyestrain, poor navigational ability and loss of productivity. Note that most cursor designs incorporate a contrasting border so that they will stand out in any environment. Strive to achieve a balanced combination of artistry and functionality and ensure that the hot spot, marked in ResEdit with an x, is positioned logically.

DITL - Dialog Item List

The DITL resource contains the list of items used in the about box. Within the DITL environment, you can click, drag and add items from the item bar situated to the right of the editing window. You may then resize these items to allow text entry or picture display. Double clicking on these items will allow you to specify the IDs of the resources you wish to include in your about box.

DLOG - Dialog

This single resource is used only to define the position of the about box window, its window type and the corresponding DITL resource which should appear in the actual about box. Kaleidoscope uses ID -14320 for both the DITL and DLOG resources.

icxx - Icon

In a Kaleidoscope scheme, icon resources are used to replace the Finder's default folder, trash, application and document appearances. All the icon resources in a scheme are purely optional, although the author of a scheme with no custom icons would be hard pressed to explain why the default Mac OS icons look better than something more unifying. There are three standard sizes of icons used in the Mac OS: large, small and mini. The large icons are drawn using the icl8, icl4 and ICN# resources, creating 8 bit (256 color), 4 bit (16 color) and 1 bit (black and white mask) versions of each. The small icons are created using the same three bit levels in the icsx series of resources. There is also a set of mini icons which can be included, although some resource editing applications do not allow this directly. There are approximately 150 icons which can be used in a Kaleidoscope scheme, most of which are specific to Mac OS 8.5. These will be listed in Appendix B. The most important, in terms of a scheme's completeness, are marked for easy identification.

PICT - Picture

Each PICT resource can contain a picture with up to 24 bits of color information. You can use these resources to create colorful logos for your scheme, which will appear in the Kaleidoscope control panel, or other artwork to be displayed when the user clicks the "About This Scheme" button. Most resource editing applications do not support creating PICTs, so you will have to use an additional drawing program if you wish to include these resources. Also, insure that your PICT IDs fall in the "safe range" of -14320 through -14305: these are the resources that Kaleidoscope uses.

ppat - Pixel Pattern

There are many places in a scheme where the use of a repeating pattern can greatly add to the overall effect. Kaleidoscope allows you to use pixel patterns for the background of any cicn which is drawn in combination with a cinf. ppats are very similar to cicn resources in many ways, including having a limit of 256 custom colors, but they are specifically designed to be repeated in places such as the desktop or utility windows, or, with the aid of Kaleidoscope, in Finder window backgrounds, alert backgrounds, dialog backgrounds and more. They are also limited to certain size constraints, specifically heights and widths that equal powers of 2 (4, 8, 16, 32, 64, etc.). Kaleidoscope can also use these resources as animation cells when creating "moving" indeterminate progress bars.

snd - Sound

Although Kaleidoscope does not support a wide variety of sound resources, a suitable pair of window collapse/expand sounds will go a long way towards spiffing up your scheme. Some resource editing applications allow for direct sound recording if your computer is equipped with a microphone, but, if you want high quality sounds for your collapse box, you will need a sound editing program to open and copy the sound file so that you can paste it into the appropriate snd resource. ID -14336 holds the collapse sound and -14335 holds the expand sound.

STR# - Text String
The text that you enter into this resource will be displayed opposite your scheme logo when the Kaleidoscope control panel is opened. This is a good place to put copyright and contact information and any other important things about your scheme. STR# -14320 must be used to store your text strings.

TMPL - Template
DO NOT TOUCH these three TMPL resources. These are bits of data, used by resource editing applications, to display hexadecimal information in a readable manner. Without these resources, entering the cinf, Colr and wnd# information would be nearly impossible. Leave these resources in your scheme.

vers - Version
The text that you enter into these resources will be displayed when a user does a "Get Info" command on your color scheme while in the Finder. This is the customary place to put your copyright information (ID 1) and the software package, eg. Siobhan's Rainforest Scheme, with which your scheme is distributed (ID 2).

wnd# - Window Part List
The wnd# resources, like the cinf resources, are new and can be source of scheme bugginess if not properly completed. Like all resources named with the pound sign, the wnd# resources contain a list of information about other parts of the resource file. In this instance, the list describes part locations present in window cicns. In the wnd# resources you are explaining to Kaleidoscope what the different parts of your window cicn are for. Before this, your window cicn is just a picture. You need to tell Kaleidoscope where the widgets are, where you want the title text to appear, where all the information in the window should be displayed and how to stretch the window cicn into a proper, full-sized window. A good trick to help you remember what wnd# resources are for is to see the "#" in the resource name as a scrap of graph paper. All you have to do to successfully edit the wnd# resources is to fill in the coordinates of the different sections, as if your window cicns were overlaid by a piece of graph paper.

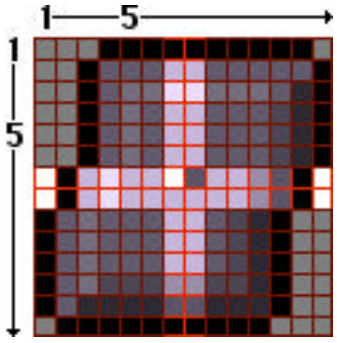
Chapter 2 • Control Information

Chapters 2 and 3 will discuss the new cinf and wnd# resources. It is important that, before beginning to create a Kaleidoscope scheme, you have a firm grasp of the manner in which these two resource types work. The numbers you use to fill in the blanks will determine whether or not your scheme works, in some cases, at all. Please read carefully and refer to Chapters 4 and 5 for examples.

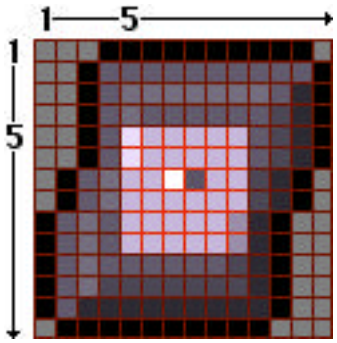
The cinf resources are extremely simple to decode. Every single one, no matter which cicn it refers to, has eleven blanks that you have to fill in. There is no rhyme or reason to the manner in which they are laid out; they are simply a collection of necessary bits of information about their corresponding cicn resources that you have to provide for Kaleidoscope to interpret and draw your your scheme controls* properly. A cinf resource tells Kaleidoscope how big to draw each of the corners, how to stretch the pixels between the corners to form the sides, how thick the sides should be, what color or patterns the main background of the control should be and what colors the text uses. In some cases, one or more of the eleven categories may not apply to the cicn you are describing. In these instances you would simply enter a 0 into the appropriate blank. That's it. So, although the cinf resource is new, it's not that complicated.

nb: the concept of corresponding cicn/cinf resources is crucial. All cinf resources refer to a cicn resource with the same ID.

The eleven categories which you must complete are:



Corner Size - Corner size can mean one of two things. In most cases, it will refer to the length of one side of a square are in the corner of a cicn that you want to remain static and undisturbed when it draws on screen.** For example, if you had a push button with a decorative circle in the corner which was five pixels in diameter, you would want to set your corner size to at least five so that it would never be cut off. In cicns without four distinct corners, those which are mainly horizontal or vertical in nature, corner size refers to the number of pixels from the right and left sides that you want to remain static. For example, Scherzo!'s progress indicator frame, cicn -10080, has a corner size of 9 to preserve the both the right and left decorative ends. Appendix A will eventually list which cicns use true corner sizes and which use "side sizes." However, it is, in the end, your decision how you want the different controls to be displayed.



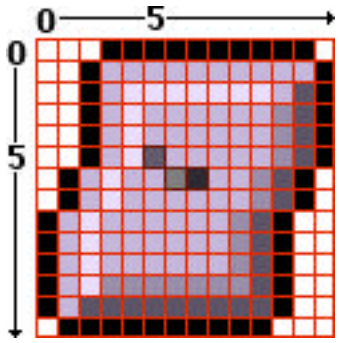
Side Thickness - Side thickness is very much like corner size except that it refers to the numbers of pixels from the outside edge of your cicn you want to remain static. For example, Scherzo!'s pull down menus have slightly beveled edges which always need to be drawn. Therefore, three pixels have been allotted to side thickness so that the bevel always appears. Side thickness is much like the "side sizes" described above, except that the pixel number specified applies to all four sides of the control.

Tile Sides - Tile sides can be set to 1 or 0. If tiles sides is set to 1, the pixels that have been chosen, in the side thickness category, to make up the edges of a cicn will be repeated, over and over, side by side, to fill in the entire width of a control. This works very similarly to Part Code 8 in wnd# resources, found in the next chapter. If tile sides is set to 0, Kaleidoscope will draw the static corners and then draw the remainder of the sides using the colors one pixel to the right of the left corner areas (for the top and bottom sides of the cicn) and one pixel down from the top corner areas (for the left and right sides of the cicn). This can be very useful if you have intricate corners and simple sides.

Pattern Anchor - The pattern anchor refers to the corner in which you want your background pattern to begin drawing, if you are using one. If you are not, set this value to 0. Patterns using value 1 will be drawn from the top/left corner of the control and repeat to the right and down, possibly leaving "half patterns" on the right and bottom sides of the control. There is no way to avoid these partial patterns, but, by using different anchors, you can choose on which sides of the control they appear. The other options work the same way but cause Kaleidoscope to start drawing the background pattern from their named corners.

0 - No Anchor
1 - Top/left
2 - Top/right
3 - Bottom/left
4 - Bottom/right

Background Pattern ID - If you are using a pattern to fill in the background of the control--background refers to any parts of the control not covered by the corners or sides-- insert the ppat ID number here so that Kaleidoscope will know which pattern to draw.



Background Pixel (y) - This number is the vertical coordinate of the pixel which you want to define as the background color for the control referred to by the cinf. It is good practice to fill this in even if you plan to use a pattern for the background of your control. There may be instances, in specific applications, where the pattern does not draw.

Background Pixel (x) - This number is the horizontal coordinate of the pixel which you want to define as the background color for the control referred to by the cinf. It is good practice to fill this in even if you plan to use a pattern for the background of your control. There may be instances, in specific applications, where the pattern does not draw.

Text Pixel (y) - This number is the vertical coordinate of the pixel which you want to define as the text color for the control referred to by the cinf.

Text Pixel (x) - This number is the horizontal coordinate of the pixel which you want to define as the text color for the control referred to by the cinf.

Embossing Pixel (y) - This number is the vertical coordinate of the pixel which you want to define as the text emboss color for the control referred to by the cinf. If you do not wish to use embossing, set this value to 0.

Embossing Pixel (x) - This number is the horizontal coordinate of the pixel which you want to define as the text emboss color for the control referred to by the cinf. If you do not wish to use embossing, set this value to 0.

If these are clear to you, there is now nothing that you do not understand about cinf resources, except, perhaps, which cicns need them and which do not. This information has been included for you in Appendix A so that you don't have to guess. If you are still a bit fuzzy about a few of the categories, the examples in Chapter 4 should clarify any remaining questions.

**A "control" can be defined as any functional part of the Mac OS interface that is drawn with a cicn and which needs some definition of color, pattern or behaviour which must be supplied by a cinf resource. An example of this would be a push button which needs an explanatory partner, the cinf, to define text color, pattern ID, etc. A good example of something which is *not* a control is a scroll arrow; it does not need a cinf because it is a static object which requires no stretching, text or pattern.*

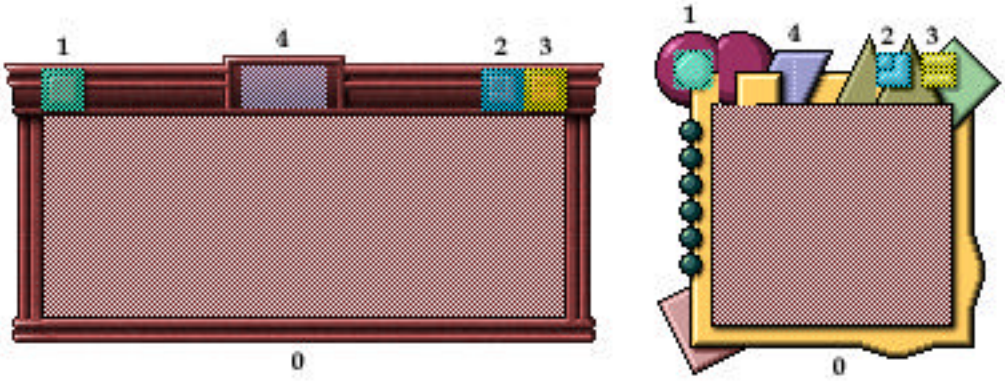
***nb: numbers entered in cinf resources refer to lengths, in pixels, of the various sections. These can be found by simply counting the pixels. Coordinates in cinf resources are also determined by counting pixels: the 0,0 pixel is located in the top/left corner. Numbers entered in wnd# resources, on the other hand, refer to point coordinates found on a grid *between* pixels, not to the number of pixels, themselves. For more information, consult Chapter 3 or see Chapters 4 and 5 for examples.*

Chapter 3 • Window Part List

The wnd# resources will take a little bit more explaining than the cinf resources. If, at any time while you are reading the remainder of this chapter, you feel that you are getting lost, refer to Chapter 5 for examples of how to determine the wnd# resource numbers from the various window cicns. This should, hopefully, clarify things and allow you to get back on track with this section.

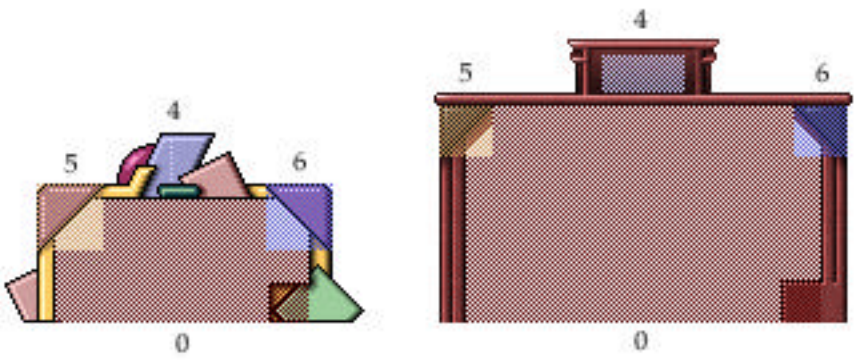
A Basic Overview Of The Window Part List

There are two main sections in each wnd# resource. The first, labeled "Rectangle List" at the top of each wnd# you open, is comprised of up to five sets of numbers corresponding to locations on your window cicns. You enter five numbers for each set: one rectangle code, which defines the section of the cicn you are referring to, and four numbers which correspond to vertical and horizontal locations in the cicn. Those four locations are, in order, the vertical coordinate of the top/left corner, the horizontal coordinate of the top/left corner, the vertical coordinate of the bottom/right corner and, not surprisingly, the horizontal coordinate of the bottom/right corner. Coordinates of what, you may ask? The five rectangular areas of each window that you may have to define are, with their rectangle codes:



0 - Content Rectangle
1 - Close Box Rectangle
2 - Zoom Box Rectangle
3 - Collapse Box Rectangle
4 - Title Text Rectangle

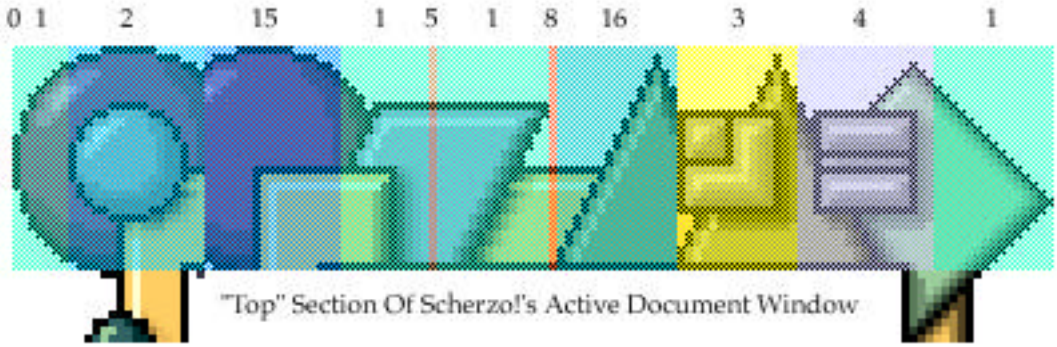
In wnd# resource ID -12320, which is paired with the Popup Window cicn, you will not have to use rectangle codes 1, 2 and 3, since Popup Windows do not have these widgets, but will have to add the following two rectangle codes to the others in the first section.



5 - Popup Window Left Grow Box Rectangle
6 - Popup Window Right Grow Box Rectangle

So, this first section, although it involves a little work, isn't very complicated once you understand what you're looking for. All you are doing is finding rectangular areas on your window cicns and entering their location numbers into a set of predefined slots. It might take a little while to get the hang of locating the correct corners, but once you do that, you've mastered rectangle codes.

The second section of each wnd# resource is actually made up of four small sub-sections. Each one represents one of the top, bottom, left and right areas of your window cicn. Each of these areas is defined in relation to the Content Rectangle from the first section. Anything above the Content Rectangle: the widgets, the text area, the top window border, etc., is part of the top section. Anything to the left of the Content Rectangle: the little round green decorations in Scherzo!, the left window side, etc., is part of the left section... and so on. You tell Kaleidoscope what the different parts of your scheme should do by using "part codes." You have a variety of codes which you can use to define how you want the different parts of each section of your window to behave. You simply fill in as many parts as you need to correctly display your scheme. However, instead of giving coordinates as you did in the first section, you give cumulative numbers to let Kaleidoscope know how far along each border your parts are located. To reiterate and simplify: each of the four sub-sections represents a portion of the window cicn to one side of the Content Rectangle. You have to let Kaleidoscope know, by entering location numbers, how the different graphical sections of your cicn picture should be translated into an actual working window.



You might ask: if all we're doing is counting across and down and telling Kaleidoscope what we want each section of our window cicns to do, when does it become difficult? If you asked that, you'll have no problems mastering wnd# resources.

Here are the various part codes you can use. Don't worry about understanding what they mean right now; each will be explained, in detail, below. However, reading the names of the part codes should give you some indication of what they're used for. They each tell Kaleidoscope to use a certain section of pixels, those outside the Content Rectangle, for unique purposes.

0 - Edge
1 - End Cap
2 - Close Box Section
3 - Zoom Box Section
4 - Collapse Box Section
5 - Title
6 - Title End Cap
8 - Stretch From Top/Left
10 - Crumple
11 - Stretch From Bottom/Right
12 - Period Repeat
13 - Period Fill From Top/Left
14 - Period Fill From Bottom Right
15 - No Close Box Section
16 - No Zoom Box Section
17 - No Collapse Box Section
18 - Stretch With Scaling

Note that although some of these part codes have the same numbers as some of the rectangle codes from section one, Kaleidoscope knows to treat them differently since they exist in different locations within each wnd# resource.

Stop here and take a breath. This may seem complicated, but, in practice, it isn't. Remember the trick to understanding wnd# resources from Chapter 1: "see the "#" in the resource name as a scrap of graph paper." In the first part of each wnd# resource, you are simply looking at the graph paper coordinates and telling Kaleidoscope where the different parts of your window exist. In the second part, you simply travel along the outside of each side of the window cicn and fill in part codes and location numbers to tell Kaleidoscope how each section behaves. If you can grasp these two concepts, the following section should help you understand exactly how they are carried out.

The Window Part List In Depth

To begin, you have to accept the fact that what is drawn in a window cicn is *not* a window. To illustrate this, do the following: while you are reading this, double click on the Scherzo! scheme so that it is active. Look around the outside of this document. On the right and bottom sides, outside the scroll bars, you'll see that almost the entire window sides, other than the corners, are made up of repeating waves. However, if you open Scherzo! in a resource editing application and look at cicn -14335, the resource for active document windows, you'll find only one wave on the right and bottom and not a scroll bar in sight. How does this happen? Why does Kaleidoscope repeat the waves?

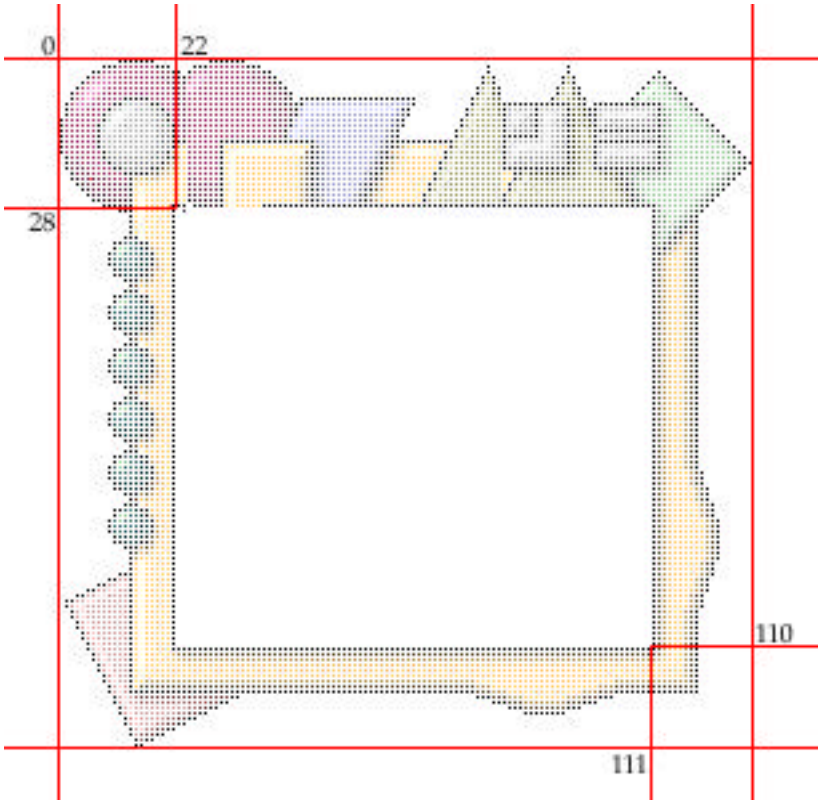
Scroll bars and scroll thumbs are not part of a window. Finder headers are not part of a window. Bevel buttons, folder icons, disclosure triangles, window background patterns and list view divider lines are not parts of a window. The words you are reading right now, although they are contained *in* a window, are not part of a window. All these parts make up the content of the

window and the actual window is what goes around them. There's a very good reason for not including these parts in the window: many applications use only some, or none, of them and your window would look quite silly if it had dependent parts missing. Keep this in mind as you design your scheme. If you look back at cicn -14335, you'll see a large, white, rectangular area in the middle of the cicn. You can probably guess what this area represents: this is the Content Rectangle. Once you have this in your window cicn and you tell Kaleidoscope where it is, all the window content we mentioned above will know where to draw itself. You will have defined where the window borders are and where the rest of the information is allowed to be drawn.

Once you understand this concept, that of defining a particular area of your window cicn to serve a certain purpose, you're more than halfway towards a good grasp of wnd# resources. In fact, once you discover how to find out what numbers define this Content Rectangle, you'll have mastered a full half of the information necessary to complete the wnd# resources.

As we learned in Chapter 1, "all you have to do to successfully edit the wnd# resources is to fill in the coordinates of the different sections, as if your window cicns were overlaid by a piece of graph paper." You can imitate this concept in most resource editing applications. For example, choose "Visible Grid lines," in the "Transform" menu, while you are looking at cicn -14335 in ResEdit. The white grid lines that are now drawn over the window cicn are the lines on your piece of virtual graph paper. If we call the leftmost grid line, the one between the border of the drawing area and the first line of pixels, Vertical Line 0 and the topmost grid line Horizontal Line 0, it should be clear why we can call the point where they intercept, at the very top/left corner, point 0,0. Point 1,1 would exist where the second topmost grid line and the second leftmost grid line intersect, one line down from the topmost grid line and one line over from the leftmost grid line. It must be noted here, that with respect to the Mac OS's method of graphing, when you encounter a set of point coordinates, the first number is always how many lines down, counting from the 0 line at the top, the point is and the second how many lines across, counting from the 0 line at the left, the point is. If you were to look for point 3,1, it would be the point at which the third line down from, and the first line to the right of, the 0,0 point cross.

If point coordinates are clear, what is the coordinate of the top/left corner of the Content Rectangle? All you have to do is get out your magnifying glass and count. If you do, you'll find that the point is located at 28,22 of cicn -14335. What if someone were to ask for the coordinate of the bottom/right corner of the Content Rectangle? Magnifying glass in hand, you'd find out it was at 110,111. Remember, instead of counting all 111 pixels, you can count backwards from the bottom/right: just check the cicn size to see where the bottom and right borders of the cicn are.



*nb: point locations are *not* pixel locations. Point locations refer to the grid lines in between the pixels. This is different from cinf coordinates which refer to the pixel numbers originating from the 0,0 pixel in the top/left corner.*

So what? You've just finished defining your first wnd# resource.

If you look back at the list of rectangle codes in Chapter 1, you will see that the code for the Content Rectangle is 0. Now, if you look at wnd# -14336, you'll see that the first entry is labeled part 0 and that the rectangle is defined as 28,22,110,111, the exact coordinates we just found in the cicn -14335 resource. That's it. All you do in the first part of a wnd# resource is enter the rectangle codes, from 0 to 6 (remember, some windows do not use all the rectangle codes) and define the top/left and bottom/right coordinates of each section of the cicn. Kaleidoscope will then know what to use those areas for and your scheme will work perfectly.

Moving on to the other rectangle codes, all you must do to define them is to imagine the smallest rectangle possible around the area you want to use as a close box, a zoom box, a collapse box and a title text area. Since Scherzo!'s close box is a circle, we'll have to imagine the corners of the rectangle sticking out past the confines of the circle and the sides of the rectangle just touching the circle. Counting the lines out to the top/left and bottom/right corners of this imaginary rectangle, we get the points 7,7 and 22,22.

nb: this is assuming that the pressed close box widget will occupy the same space as the active one. If, for some reason, your widget grows larger as you press it, you will need to map your Close Box Rectangle with the larger pressed close box coordinates. The same rule applies to the other widgets.

So, entry number two in our wnd# resource will be Rectangle Code 1 defined as 7,7,22,22. The zoom box and collapse box should be no problem at all since they are perfect rectangles already. You just need to find the top/left and bottom/right corners and you're set.

For popup windows, you will also need to define the grow boxes. Since a popup window is anchored to the bottom of the screen, there is no grow box at the bottom/right where you would normally find one. Instead, there are two which exist in the top/right and top/left corners of the cicn. Uniquely, these two components are drawn within the Content Rectangle due to space constraints. So, in order to differentiate them from the Content Rectangle, and to tell Kaleidoscope what they are to be used for, you must specify their locations with wnd# entries similar to the ones above.

The Title Text Rectangle works in a slightly different way since the text in the title bar of a window can vary anywhere from one to two-hundred fifty-five characters in length. With this variable and the ability to use different fonts for the System font, you have no way of determining just how long or short you should make your Title Text Rectangle so that any text will fit into it correctly. Look at wnd# -14336, in the Rectangle List, at Rectangle Code 4. The coordinates of the Title Text Rectangle are 10,52 and 26,53. This means that the rectangle is sixteen pixels high and only one pixel wide.

*nb: Title Text Rectangles for document, alert and dialog windows *must* be exactly 16 pixels high in all cases. This is to insure a proper fit with the largest System font available in the Kaleidoscope control panel and to allow the proper display of 16 pixel tall window proxy icons. Utility window Title Text Rectangles can be slightly smaller since they use a smaller font.*

Clearly, no text would ever fit into that area. Open cicn -14335 and look at the space that corresponds to those pixel numbers. You'll find it directly in the middle of the light blue area above the Content Rectangle. Obviously, this is the place you'd expect to find it since the Title Text Rectangle appears above the Content Rectangle in real windows and the title bar in Scherzo! is that same light blue color. How does Kaleidoscope know how long to make the title bar when you've only allotted one pixel to the Title Text Rectangle? It stretches; it's as simple as that. The more characters that appear in the title bar, the longer the title bar is stretched. By entering Rectangle Code 4 in your wnd# resource, you have told Kaleidoscope that the area you are going to define is for text and that it should stretch the area, horizontally, depending upon how long the string of characters in the title bar gets.

*nb: the Title Text Rectangle must always be contained in the window title and title end cap sections listed below (the sum of Part Codes 5 and 6): it is allowed to spill into the end caps. However, the Title Text Rectangle *must* completely contain the window title section (Part Code 5). Otherwise, the text will not display properly and will be cut off. See Chapter 5 for an example.*

So, this is what was meant when we discussed the fact that "what is drawn in a window cicn is *not* a window." There is no way to draw an indeterminately long Title Text Rectangle in your window cicns, so we have to let Kaleidoscope know that a section of the cicn *represents* the title bar, so that the control panel will know where to work its magic stretching routine. This is the most important concept in creating Kaleidoscope schemes. What you draw in your window cicns is not, in fact, the window as you would see it in a screen capture, but the window as it is made up of individual parts. It should now be clear why there is only one wave on the right and bottom sides of Scherzo!'s cicn -14335. That single wave has been defined as a Period Repeat: a section of the window cicn which will repeat over and over as long as there is room for it. Kaleidoscope has been informed that it is a special type of section and it draws the result according to the part code with which that area has been defined.

There is a wide variety of part codes that you can use to define what certain sections of your window cicns should do. These were listed above and you were encouraged to guess their functions from their names. The guesswork will now be taken out as we define each of the part codes.

0 - Edge - The edge is *not* the border of the cicn resource but is the point furthest to the left (on the top and bottom sections) or closest to the top (on the left and right sections) at which parts of the window actually begin to draw. This will usually be 0, but, if your window borders are irregular, can vary widely. Part Code 0 can also be used to tell Kaleidoscope not to draw certain pixels at all. For example, if you had a wildly intricate top section, with many, many part codes, it could get extremely wide. If your bottom section was a very simple line, you might wish to tell Kaleidoscope not to draw any of the bottom pixels, except one, and to stretch that pixel the whole width of the window. The fewer pixels Kaleidoscope has to deal with, the faster your windows will draw on screen.

1 - End Cap - The term "end cap" is something of a misnomer, as these sections can appear in any portion of your window cicn, not only the ends. However, they are almost always used at the ends, hence the name. End caps are static sections of pixels, sections that will be drawn *exactly* as they are in the cicn. They are not stretched or repeated but are displayed on screen precisely as you draw them in your window resource. For this reason, you will want to limit the size of your end caps. If they are too large, they will not display properly-- they may extend past the edges of the window border-- in small windows.

2 - Close Box Section - We have already defined the location of the close box with the Close Box Rectangle in the first section of the wnd# resource. However, some windows do not have close boxes and those sections of the window must be removed. Here is where you define the width of the close box so that it will not display when it isn't needed. Note that the two parts to the right and left of the Close Box Section should be drawn in such a way that if that section is removed, the adjoining sections will fit together nicely. If you wish to change what will be displayed, instead of simply removing a section, you may also use Part Code 15 to specify what the section without the close box will look like.

3 - Zoom Box Section - We have already defined the location of the zoom box with the Zoom Box Rectangle in the first section of the wnd# resource. However, some windows do not have zoom boxes and that section of the window must be removed. Here is where you define the width of the zoom box so that it will not display when it isn't needed. Note that the two parts to the right and left of the Zoom Box Section should be drawn in such a way that if that section is removed, the adjoining sections will fit together nicely. If you wish to change what will be displayed, instead of simply removing a section, you may also use Part Code 16 to specify what the section without the zoom box will look like.

4 - Collapse Box Section - We have already defined the location of the collapse box with the Collapse Box Rectangle in the first section of the wnd# resource. However, some windows may not have collapse boxes and that section of the window must be removed. Here is where you define the width of the collapse box so that it will not display when it isn't needed. Note that the two parts to the right and left of the Collapse Box Section should be drawn in such a way that if that section is removed, the adjoining sections will fit together nicely. If you wish to change what will be displayed, instead of simply removing a section, you may also use Part Code 17 to specify what the section without the collapse box will look like. Note: at this time there are no windows which are drawn without collapse boxes. However, for future compatibility, you may wish to include Part Code 17.

5 - Title - This part code is used to define that section of the window cicn which is to appear behind the title text, ie. behind the Title Text Rectangle you defined in the previous part of the wnd# resource. Any pixels that appear in this section will be repeated to fill the area as it stretches or contracts with changing window names.

6 - Title End Cap - These sections are very similar to normal end caps, in that they are static sections which appear on screen exactly as they are drawn in the window cicn, but they will only appear when the window that is being displayed has a visible title section. Title end caps are used to embrace the title area from the left and right in order to set it apart from the rest of the window sections, to give the text some "breathing room." Part Code 6 can only be used in window cicns which make use of Part Code 5.

7 - [reserved for future use]

8,11 - Stretch From Top/Left - Scott Naness, in his discussion of wnd# resources describes a stretch region beautifully: "If you define the stretch area as a single pixel wide, then [Kaleidoscope] takes that pixel and stretches it... to the width that it will need, depending on the width of the window. If you define the stretch area as more than a single pixel, then [Kaleidoscope] takes the entire width [of the stretch region], draws it, and repeats it as many times as it needs to in order to fill the [available] space... If, after repeating the pattern [a number of] times, there are still more pixels left to draw, but not enough for the pattern to fully repeat again, then [Kaleidoscope] will draw as much as it can and truncate the rest." It should be noted that stretch regions defined by Part Code 8 will be drawn from the top to the bottom and from left to right depending upon the side of the cicn it occupies. Stretch regions defined by Part Code 11 will be drawn from the bottom to the top and from right to left depending upon the side of the cicn it occupies.

9 - [reserved for future use]

10 - Crumple - A crumple zone is a static region which, unlike an end cap, will be removed from the window side if there is not enough room to fit it in its entirety. An end cap will always be displayed as it is drawn, but a crumple zone will disappear if the space that is left for it is less than its complete length. The two sections on either side of the crumple zone will be joined as if Part Code 10 never existed. Note that all crumple zones on one side of a window collapse simultaneously. If there is not enough room for the largest, they will all crumple and be replaced by expanding stretch regions.

12 - Period Repeat - A period repeat is a section of the window cicn, like the wave we examined in Scherzo!, which is designed to be repeated, over and over, to fill a section of window border.* Unlike a stretch region, which will combat remaining space by drawing as much of the region as possible, and cutting off the rest, a period repeat is always repeated in its entirety and is never partially drawn. Leftover space must be completed by a period fill or stretch region.

13,14 - Period Fill From Top/Left, Bottom/Right - A period repeat must be accompanied by either Part Code 13, Part Code 14 or one of the stretch regions. The leftover space created by a period repeat can be completed with a period fill which acts as a miniature stretch region for that remaining period repeat space. Note that if you choose to use a stretch region instead of a period fill, the rule outlined in the footnote applies. Period fill regions defined by Part Code 13 will be drawn from the top to the bottom or from left to right depending upon the side of the cicn it occupies. Period fill regions defined by Part Code 14 will be drawn from the bottom to the top or from right to left depending upon the side of the cicn it occupies.

15 - No Close Box Section - Discussed in Part Code 2, above. A replacement section to be displayed in lieu of a close box in windows which do not contain that element.

16 - No Zoom Box Section - Discussed in Part Code 3, above. A replacement section to be displayed in lieu of a zoom box in windows which do not contain that element.

17 - No Collapse Box Section - Discussed in Part Code 4, above. A replacement section to be displayed in lieu of a collapse box in windows which do not contain that element.

18 - Stretch With Scaling - Instead of repeating the stretch section side by side until the window is full, this part code will stretch each pixel in the region by the same amount. For example, if you had a stretch region five pixels wide, with each pixel of width being a different color, when that area of the actual window was stretched to fifty pixels, each of the five colors would be ten pixels wide. This part code is extremely useful for creating smooth gradients. Note: this part code is new to Kaleidoscope 2.1. If you include stretch with scaling zones in your scheme, the minimum version string in your Colr resource should be set to $21.

Remember, when you describe your window parts in a wnd# resource, you have to provide *cumulative* widths; you do not give pairs of coordinates as you did for the Rectangle Codes. For example, your scheme's edge could begin at 0 and you would enter Part Code 0, Border 0. You might then have a section, ten pixels wide, which you always want displayed no matter the size of the real window. You would enter Part Code 1, Border 10. You might then have a close box which is twenty pixels wide, so you would enter Part Code 2, Border 30, the *absolute* value of the end of the close box. The numerical value of each border entry is cumulative and dependent upon the previous one.

That's it. If you managed to get through all that information unscathed, you can now create the most wildly intricate Kaleidoscope scheme possible. All the difficult concepts and tasks are now out of the way and the rest of the scheme should be a breeze. Carefully study the examples in Chapters 4 and 5 to get a good feel of how these descriptions of what to do relate to the actual practice of scheme creation. After that, move on to Chapter 6 which deals with all the other cicn resources which do not need cinf or wnd# partners.

*It should be noted that stretch regions and period repeats are "equal" in the eyes of Kaleidoscope. If one side of a window cicn were to have a single period repeat and a single stretch region, each would claim half the real estate available. If a crumple zone or an end cap were placed between them, Kaleidoscope would center the static section along the window side and give the remaining spaces to each grow region. This also applies to two stretch regions or two period repeats.

Chapters 4-7

The following sections are not yet completed. While they are not crucial to an understanding of how to create schemes using the newer Kaleidoscope format, they certainly might provide a useful addendum to Greg's wonderful notes that are included in the Kaleidoscope distribution. The sections on these pages will be completed if and when there is time to do so. If you feel that you have the knowledge and interest to complete a chapter, or even an example or two, please do so and I will be more than pleased to include your additions in the appropriate sections.

Chapter 4 • Creating Controls Examples

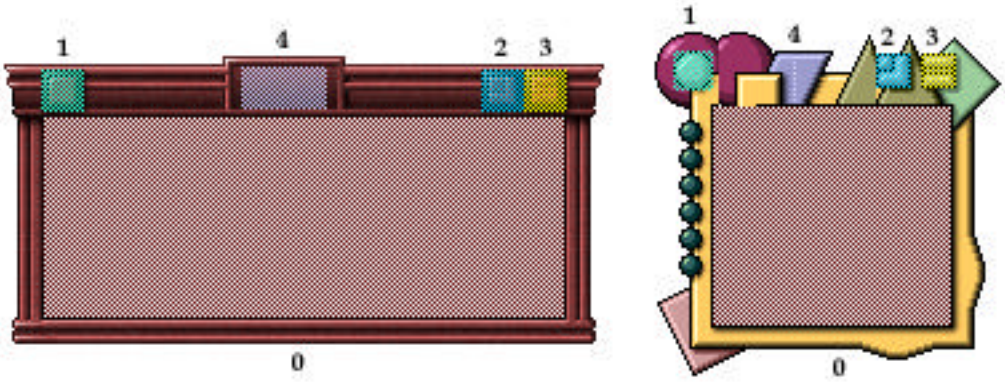
The following are graphical examples of how to create a number of the more complicated cicn/cinf pairs. Each control cicn is labelled and paired with its corresponding cinf resource so that you can see from exactly where each of the entered numbers originates.

To be completed...

Chapter 5 • Creating Windows Examples

The following are graphical examples of how to create a number of the more complicated cicn/wnd# pairs. Each window cicn is labelled and paired with its corresponding wnd# resource so that you can see from exactly where each of the entered numbers originates.

Document Windows - Rectangle Codes



The two graphics above are cicns, both ID -14335, from Antique and Scherzo! All the necessary rectangle parts are labelled so that you can see where to properly place them on your scheme.

Part Code 0: Note how Rectangle Part 0 is completely within the black border; the border is not included when figuring the coordinates for the content rectangle. Otherwise, objects within the content rectangle might overlap the border and cause unsightly clashes.

Part Codes 1-3: While figuring out the rectangles for the various widgets is quite elementary, it is worth noting that the close box widget in Scherzo! overlaps the actual circle used for that control. The reasoning behind this is discussed in Chapter 3, but is worth repeating here: the coordinates for each Rectangle Code must encompass the entire widget they represent. If that widget is not rectangular in shape, you must include all the extraneous parts even if that means increasing the size of your rectangle past the borders of the widget.

Part Code 4: You'll notice that the rectangles for the Title Text Rectangles differ greatly in these

Part Code 4: You'll notice that the rectangles for the Title Text Rectangles differ greatly in these two schemes. There is no operational reason for this. As was mentioned in Chapter 3, the Ttitle Text Rectangle stretches to accommodate different lengths of text. In this case, the background to Antique's Title Text Rectangle has an intricate texture that Arlo wanted repeated. Scherzo!, on the other hand, has a simple, single color, light blue Title Text Rectangle background that does not need to be treated specially.

To be completed...

Chapter 6 • Other Color Icons

There are a number of necessary scheme cicns which use neither a wnd# nor a cinf resource. However, they must still conform to certain specifications and rules so that Kaleidoscope understands how to interpret them correctly. They must also have both an accompanying black and white resource and an appropriate mask, just as those with wnd# and cinf resources did. The following is a complete list of those resources along with the particularities of each type.

To be completed...

Chapter 7 • Graphic Completeness

Once you have successfully completed all your cicn, cinf and wnd# resources, you are well on your way towards a unified scheme. However, there are a number of other graphic components which must be completed before your scheme will look professionally crafted. This chapter will explain how to properly create color look up tables, cursors, system and folder icons, any pictures you may need to include and ppat resources for using in combination with other resources and as desktop and utility patterns.

To be completed...

Chapter 8 • Text, Sound, About Boxes

Now that you have finished the graphics for your scheme, there are a few small resources that must be added to insure proper functionality and total completeness. Chief among these is the Colr resource which provides crucial information to Kaleidoscope about a number of different scheme elements. As well, you should be sure to complete the STR# and vers resources and provide your scheme with some nifty window shade sounds and a cool about box.

The Colr resource is one of those which can only be filled in if the TMPL resource is present. This is why you were warned not to remove it. Here are explanations for the nine different categories you must define.

Colr version - This should always be set to 1. This may change if the Colr resource is expanded, but, for now, 1 is appropriate.

Color scheme file format version - This should always be set to 1. This could change if the manner in which Kaleidoscope uses its resources is altered, but, for now, 1 is appropriate.

Minimum Kaleidoscope version - Different versions of Kaleidoscope allow certain advancements in terms of scheme characteristics. For example, Kaleidoscope 2.1 is the first version to allow Part Code 18 to be used in scheme files. If your scheme used this part code, you would need to enter $21 in this blank. The entry is always "$" followed by the Kaleidoscope minimum Kaleidoscope version without intermediate "."s. Since there is no reason for Kaleidoscope users *not* to be using the newest version, you are always safe using that.

Has accent colors - Does your scheme have alternate versions of the scroll thumb, menu highlight colors and progress bars? If so, click True. If not... and most schemes which are not variations of Apple platinum will not... click False.

Stretch scroll bar thumb from center (for SmartScroll) - If the value is 1, Kaleidoscope stretches the thumb from the center. If the value is 0, the thumb is stretched three pixels in from either edge. If the value is negative, Kaleidoscope takes the absolute value (removes the negative sign) and stretches the thumb that many pixels from either edge. Note that scroll bar thumbs are never tiled. Kaleidoscope repeats the row that it is stretching for the entire stretched region (like a one pixel wide stretch part in a wnd#). You cannot have patterns in scrollbar thumbs.

Menu highlight cicn overlays normal menu cicn - If this option is turned off, each entire menu (including all items) will act as a unified object. Each menu highlight will act as a section of the menu and be displayed as such. With this option turned on, each menu item is treated as the whole menu is in the off setting. So, you can have individual beveled items with this options set to True or one large beveled area with this option set to False.

Unified scroll bar track - Does your scheme have scroll bars as described in Chapter 6? If so, click True. If not, and you are still using scroll bars of the style used by Apple platinum, Onyx, or other bundled schemes, click False.

Windows style scroll bars - If this option is enabled, clicking inside the scroll bar track will highlight only the part of the track which you have pressed. The section on the other side of the scroll thumb will not be affected. Normally, clicking anywhere in the scroll bar track would cause both sections, on either side of the scroll thumb, to changed to the pressed state. This option takes effect only when unified scroll bar tracks are turned OFF.

Extended scroll bar arrows - Normal scroll arrows extend 16 pixels from the edge of the window

Extended scroll bar arrows - Normal scroll arrows extend 16 pixels from the edge of the window into the scroll bar track. Scroll bar arrows that are much longer can be achieved by using this option. In that case, the scroll bar cicn is divided in half, either horizontally or vertically, depending upon orientation, and each half is treated as one of the scroll bar arrows. This allows you have extra large arrows that merge into the track. The hitch is that since the cicn is used for drawing the larger pressed arrows, your whole track cannot highlight with this option turned on. This option takes effect only when unified scroll bar tracks are turned ON.

You've now completed the Colr resource and you are finished with the compulsory sections of your scheme. However, you will want to provide additional information so that people understand that this is, indeed, your scheme and that it is copyrighted and should not be abused.

The STR# resource contains the textual information that is displayed in the Kaleidoscope control panel next to the scheme logo that was described in the last chapter. You can place whatever text you want here, but the standard protocol is to enter the name of your scheme, the copyright information, such as "©1998 by Joe Schemer" and any personal information you wish to include.

The two vers resources are more traditional and should be filled out as described in Chapter 1:

vers 1 - DoodleScheme ©1998 by Miss Schemer
vers 2 - Scheme 1 In The DoodleScheme Package

Including window shade sounds in a scheme is not like any other part of scheme creation. Some resource editing applications, like ResEdit, have the ability to record sounds directly from your computer's microphone. You simply choose Record New Snd.. and you are presented with the standard Macintosh recording dialog. Any sound you record will be inserted into the snd resources in your scheme and you will simply have to renumber it to the appropriate ID: -14336 for the collapse sound and -14335 for the expand sound.

However, unless you have a magnificent microphone attached to your computer, the results of this recording will be far from satisfactory. You will want to obtain a dedicated sound application to tweak your sounds: reduce the ambient noise, add effects, change volumes, etc. Among the shareware options that are readily available on the web are SoundEffects, Sound Sculptor II, D-SoundPRO and the new Amadeus, which has a very good noise reduction option. All of these applications allow you to record sounds, alter them and to save them in a variety of sound formats. The trick is to get the created sound files into your scheme.

If you create your sound, or open it, in one of the sound applications mentioned above, all you have to do is to highlight the sound (choose Select All from the Edit menu) and copy it into your computer's memory. Opening your scheme in ResEdit and creating a new snd resource will allow you to paste the sound into your scheme. Renumber the ID and you're set!

Another method is to use the freeware application SoundApp to convert your created sound file into a snd resource file. You can then open that in ResEdit and copy and paste it into your scheme file. The choice is yours, but make sure you have some good, appropriate sounds: they add a tremendous amount to the overall effect of your scheme.

The last special effect you can add to your scheme is a **custom about box**. This is what people will see when they hit the "About This Scheme" button in the Kaleidoscope control panel. In essence, this is your chance to design the packaging for your scheme: a completely personal splash screen complete with graphics and text.

Upon opening a DLOG resource, you will be presented with a large window divided into many parts. The main section in the middle will document any changes you make to the setting which appear around the outside of this window. The settings are divided into three sections: the top section, which allows you to select a type of window in which to display your about box, the bottom section, which positions your about box where you want it on the screen, and the right section, which determines the colors that are used to draw the edges of the about box window.

The numbers that comprise the bottom section can be whatever you like: as long as you make the area large enough to contain any text and pictures that you want to place there, any size will do. The "Top" and "Left" coordinates are for the top/left corner of the about box window. They correspond to the location on your screen that you want the about box to start drawing. The "Height" and "Width" numbers determine the size of your window, in pixels.

To select a window type, simply click on the picture of your choice in the top section. The style you have chosen will appear in the window below the top section.

To the right is a small selection of color swatches. Simply click and hold on each of the swatches and choose the color you want from the popup window. The results will be displayed in the preview window. Insure that the "Initially visible" check box is checked. The close box option does not matter as the about box disappears with a single click. Last, insure that the DITL ID box says -14320 and you're finished. Off to the DITL resource.

When opening the DITL resource, you will see a blank window of the size you specified in the DLOG resource. There is also a small tool bar with a variety of interface options attached to it. You simply drag and drop the picture, text or icon tools onto the blank window and change them to display the way you like. Text boxes can be stretched to accommodate longer entries. When you add a picture or an icon, you must double-click on the generic place holder and insert the ID of the PICT or icon resource you wish to use in your about box. This will tell Kaleidoscope which pictures to display when someone clicks the "About This Scheme" button.

You may have to play with the different placement values, both in the DLOG and DITL resources, a little to get your about box displayed exactly the way you like it, but you can't destroy anything so fiddle until you have everything perfect.

That's a good place to stop this discussion of Kaleidoscope schemes: fiddle until you have everything perfect.

Appendix A • Resource List

*nb: resources prefixed with a "w" or "c" *must* be accompanied by a corresponding wnd# or cinf resource, respectively, with the same ID. Resources appended with an /s/ suffix use the "side size" option of Corner Size, discussed in Chapter 2.*

cicn - Color Icon

Document Windows

w-14336 Inactive Document Window
-14335Document Window
-14334Inactive Grow Box
-14333Grow Box
w-14332 Inactive Collapsed Document Window
-14331Collapsed Document Window
-14330Widget Down States
-14329Alternate Zoom Box Widgets

Dialog Boxes And Alerts

w-14328 Inactive Dialog
-14327Dialog
w-14326 Inactive Alert
-14325Alert
w-14324 Inactive Movable Modal
-14323Movable Modal
w-14322 Inactive Movable Alert
-14321Movable Alert
-14310Inactive Movable Modal Grow Box
-14309Movable Modal Grow Box

Utility Windows

w-14304 Inactive Titled Utility Window
-14303Titled Utility Window
-14302Inactive Large Utility Window Grow Box
-14301Large Utility Window Grow Box
w-14300 Inactive Collapsed Titled Utility Window
-14299Collapsed Titled Utility Window
-14298Pressed Utility Window Widgets
-14297Utility Window Alternate Zoom Box Widgets
w-14296 Inactive Side Floating Utility Window
-14295Side Floating Utility Window
-14294Inactive Side Floating Utility Window Grow Box
-14293Side Floating Utility Window Gow Box
w-14292 Inactive Collapsed Side Floating Utility Window
-14291Side Floating Utility Window
-14290Pressed Side Floating Utility Window Widgets
-14289Side Floating Utility Window Alternate Zoom Box Widgets
w-14288 Inactive No Title Utility Window
-14287No Title Utility Window
-14286Inactive Small Utility Window Grow Box
-14285Small Utility Window Grow Box

-14285Small Utility Window Grow Box
w-14284 Inactive Collapsed No Title Utility Window
-14283Collapsed No Title Utility Window
-14278Inactive Small Side Floating Utility Window Grow Box
-14277Small Side Floating Utility Window Grow Box

Popup Windows

w-12320 Popup Window

Menus

c-12240 Menu Bar
c-12239 Menu Bar Item
c-12238 Selected Menu Bar Item
c-12237 Pull Down Menu Background
c-12236 Selected Pull Down Menu Item
c-12235 Divider Line /s/
c-12234 Solo Menu Background
c-12233 Selected Solo Menu Background
-12231Application Menu Grip
-12230Pressed Application Menu Grip
-12229Inactive Application Menu Grip

Push Buttons

c-10240 Inactive Push Button
c-10239 Push Button
c-10238 Pressed Push Button
-10232Inactive Default Ring
-10231Default Ring

Scroll Bar Thumbs (Also See Scroll Bars)

-10208Normal Vertical Thumb
-10207Pressed Vertical Thumb
-10206Normal Horizontal Thumb
-10205Pressed Horizontal Thumb

Bevel Buttons

c-10176 Inactive Small Bevel Button
c-10175 Small Bevel Button
c-10174 Pressed Small Bevel Button
c-10173 Inactive On Small Bevel Button
c-10172 On Small Bevel Button
c-10171 Pressed On Small Bevel Button
c-10167 Inactive Bevel Button
c-10166 Bevel Button
c-10165 Pressed Bevel Button
c-10164 Inactive On Bevel Button
c-10163 On Bevel Button
c-10162 Pressed On Bevel Button

Sliders

c-10144 Inactive Down Pointing Slider Track /s/
c-10143 Down Pointing Slider Track /s/
c-10142 Pressed Down Pointing Slider Track /s/
-10141Down Pointing Slider Thumbs
-10140Inactive Down Pointing Slider Tick
-10139Down Pointing Slider Tick
c-10138 Inactive Up Pointing Slider Track /s/
c-10137 Up Pointing Slider Track /s/
c-10136 Pressed Up Pointing Slider Track /s/
-10135Up Pointing Slider Thumbs
-10134Inactive Up Pointing Slider Tick
-10133Up Pointing Slider Tick
c-10132 Inactive Non-Directional Horizontal Slider Track /s/
c-10131 Non-Directional Horizontal Slider Track /s/
c-10130 Pressed Non-Directional Horizontal Slider Track /s/
-10129Non-Directional Horizontal Slider Thumbs
c-10128 Inactive Right Pointing Slider Track /s/
c-10127 Right Pointing Slider Track /s/
c-10126 Pressed Right Pointing Slider Track /s/
-10125Right Pointing Slider Thumbs
-10124Inactive Right Pointing Slider Tick
-10123Right Pointing Slider Tick
c-10122 Inactive Left Pointing Slider Track /s/
c-10121 Left Pointing Slider Track /s/
c-10120 Pressed Left Pointing Slider Track /s/
-10119Left Pointing Slider Thumbs
-10118Inactive Left Pointing Slider Tick
-10117Left Pointing Slider Tick
c-10116 Inactive Non-Directional Vertical Slider Track /s/
c-10115 Non-Directional Vertical Slider Track /s/
c-10114 Pressed Non-Directional Vertical Slider Track /s/
-10113Vertical Non-Directional Slider Thumbs

Disclosure Triangles

-10112Inactive Right Pointing Disclosure Tringle
-10111Right Pointing Disclosure Triangle
-10110Pressed Right Pointing Disclosure Triangle
-10109Disclosure Triangle Animation
-10108Disclosure Triangle Animation
-10107Disclosure Triangle Animation
-10106Disclosure Triangle Animation
-10105Disclosure Triangle Animation
-10104Inactive Down Pointing Disclosure Triangle
-10103Down Pointing Disclosure Triangle
-10102Pressed Down Pointing Disclosure Triangle
-10101Disclosure Triangle Animation
-10100Disclosure Triangle Animation
-10099Disclosure Triangle Animation
-10098Disclosure Triangle Animation
-10097Disclosure Triangle Animation

Progress Indicator

c-10080	Progress Indicator Frame /s/
c-10079	Full Progress Indicator Section /s/
c-10078	Progress Indicator Track /s/
c-10077	Inactive Progress Indicator Frame /s/
c-10076	Inactive Full Progress Indicator Section /s/
c-10075	Inactive Progress Indicator Track /s/

Little Arrows

-10048Inactive Little Arrows
-10047Little Arrows
-10046Pressed Up Little Arrow
-10045Pressed Down Little Arrow

Tabs

c-9984 Large System Font Disabled Rear Tab /s/
c-9983 Large System Font Rear Tab /s/
c-9982 Large System Font Pressed Rear Tab /s/
c-9981 Large System Font Front Disabled Tab /s/
c-9980 Large System Font Front Tab /s/
c-9978 Large System Font Disabled Tab Pane
c-9977 Large System Font Tab Pane
c-9976 Small System Font Disabled Rear Tab /s/
c-9975 Small System Font Rear Tab /s/
c-9974 Small System Font Pressed Rear Tab /s/
c-9973 Small System Font Front Disabled Tab /s/
c-9972 Small System Font Front Tab /s/
c-9970 Small System Font Disabled Tab Pane
c-9969 Small System Font Tab Pane

Background Colors

c-9776 Active Alert/Dialog Background
c-9775 Inactive Alert & Dialog Background
c-9568 Inactive Finder Header
c-9567 Finder Header
c-9552 Desktop Text Color And Background
c-9551 Icon View Background Color
c-9550 List View Non-Sort Color
c-9549 List View Sort Color
c-9548 List View Divider Line Color

Check Boxes

-9504	Inactive Un-Checked Check Box
-9503	Un-Checked Check Box
-9502	Pressed Un-Checked Check Box
-9501	Inactive Checked Check Box
-9500	Checked Check Box
-9499	Pressed Checked Check Box
-9498	Inactive Mixed Check Box

-9497	Mixed Check Box
-9496	Pressed Mixed Check Box
-9495	Checked Alternate Checked Check Box
-9494	Alternate Checked Check Box
-9493	Pressed Alternate Checked Check Box

Radio Buttons

-9492	Inactive Off Radio Button
-9491	Off Radio Button
-9490	Pressed Off Radio Button
-9489	Inactive On Radio Button
-9488	On Radio Button
-9487	Pressed On Radio Button
-9486	Inactive Mixed Radio Button
-9485	Mixed Radio Button
-9484	Pressed Mixed Radio Button

Scroll Bars

-8288	Inactive Horizontal Scroll Bar Track
-8287	Horzontal Nothing To Scroll Track
-8286	Horizontal Scroll Bar Track/Arrows
-8285	Pressed Horizontal Scroll Bar Track/Arrows
-8283	Double Headed Nothing To Scroll
-8282	Doubled Headed Horizontal Scroll Bar
-8281	Pressed Horizontal Doubled Headed Scroll Bar
-8280	Inactive Vertical Scroll Bar Track
-8279	Vertical Nothing To Scroll Track
-8278	Vertical Scroll Bar Track/Arrows
-8277	Pressed Vertical Scroll Bar Track/Arrows
-8275	Vertical Double Headed Nothing To Scroll
-8274	Vertical Double Headed Scroll Bar
-8273	Pressed Vertical Double Headed Scroll Bar
-8272	Vertical Thumb Ghost
-8271	Horizontal Thumb Ghost
-8270	Inactive Small Horizontal Nothing To Scroll
-8269	Small Horizontal Nothing To Scroll
-8268	Small Horizontal Scroll Bar
-8267	Pressed Small Horizontal Scroll Bar
-8265	Small Doubled Headed Nothing To Scroll
-8264	Small Doubled Headed Horizontal Scroll Bar
-8263	Pressed Small Doubled Headed Horizontal Scroll Bar
-8262	Inactive Small Vertical Nothing To Scroll
-8261	Small Vertical Nothing to Scroll
-8260	Small Vertical Scroll Bar Track
-8259	Pressed Small Vertical Scroll Bar Track
-8257	Small Nothing To Scroll Double Headed
-8256	Small Double Vertical Headed Scroll Bar
-8255	Pressed Small Vertical Double Headed Scroll Bar
-8254	Small Vertical Thumb Ghost
-8253	Small Horizontal Thumb Ghost
-8252	Small Vertical Thumb
-8251	Pressed Small Vertical Thumb

-8250 Small Horizontal Thumb
-8249 Pressed Small Horizontal Thumb

Popup Menus

c-8208 Inactive Popup Menu Text Section
c-8207 Popup Menu Text Section
c-8206 Pressed Popup Menu Text Section
c-8205 Inactive Popup Menu Arrow Section
c-8204 Popup Menu Arrow Section
c-8203 Pressed Popup Menu Arrow Section
c-8202 Inactive Popup Menu Arrow Only
c-8201 Popup Menu Arrow Only
c-8200 Pressed Popup Menu Arrow Only
-8199 Inactive Large Popup Menu Arrow
-8198 Large Popup Menu Arrow
-8197 Pressed Large Popup Menu Arrow
-8196 Inactive Small Popup Menu Arrow
-8195 Small Popup Menu Arrow
-8194 Pressed Small Popup Menu Arrow

clut - Color Look Up Table

-14336Active Header
-14335Inactive Header
300 Appearance1.0.X Accent Color Ramp

Colr - Color Scheme Information

129 Kaleidoscope Scheme Information

crsr - Cursor

-20488Context Menu Finder Cursor
-20487Make Alias Finder Cursor
-20486Dupicate Item Finder Cursor
0 Custom Arrow Cursor

DITL - Dialog Item List

-14320Scheme About Box Dialog Information

DLOG - Dialog

-14320Scheme About Box Dialog

icxx - Icon

-16455Custom Scheme Icon

PICT - Picture

-14336Scheme Logo
-14320Scheme About Box Picture

ppat - Pixel Pattern

-10080 Indeterminate Progress Step 1
-10079 Indeterminate Progress Step 2
-10078 Indeterminate Progress Step 3
-10077 Indeterminate Progress Step 4
-10072 Inactive Indeterminate Progress Step 1
-10071 Inactive Indeterminate Progress Step 2
-10070 Inactive Indeterminate Progress Step 3
-10069 Inactive Indeterminate Progress Step 4
c17 Desktop Pattern
c42 Utility Pattern

snd - Sound

-14336 Collapse Sound
-14335 Expand Sound

STR# - Text String

-14320 Control Panel Info Strings

TMPL - Template

128 Colr
129 cinf
1240 wnd#

vers - Version

1 Scheme Finder Information
2 Scheme's Creator Finder Information

Appendix B • Mac OS 8.5 Icons

* Indicates most commonly used icons; should be incorporated.
- Indicates less commonly used icons; may be incorporated.
" Indicates duplicate icon IDs; icon is incorporated elsewhere.
? Indicates icons of unknown usage.
8 Indicates icons also compatible with Mac OS 8.

Connect To Icon	-20802
Internet Search Sites Folder Icon	-20801*
Unlocked Icon	-20798
Generic PC Card Icon	-20799
Scripts Folder Icon	-20797*
Internet Location AppleTalk Zone Icon	-20796
FTP Server Icon	-20795
TrueType MultiFlat Font Icon	-20794
AFP Server Icon	-20793
ColorSync Folder Icon	-20792*
Sort Descending Icon	-20791*
Sort Ascending Icon	-20790*
Alias Badge Icon	-20789-
Shared Badge Icon	-20788
Mounted Badge Icon	-20787
Locked Badge Icon	-20786
Generic Font Scaler Icon	-20752
Sharing Privs Writable Icon	-20751
Sharing Privs Unknown Icon	-20750
Sharing Privs Not Applicable Icon	-20749
Sharing Privs Read Only Icon	-20748
Sharing Privs Read Write Icon	-20747
Grid Icon	-20746*
Keep Arranged Icon	-20745*
Internet Folder Icon	-20744*
Backward Arrow Icon	-20741
Forward Arrow Icon	-20742
AppleScript Badge Icon	-20740
Internet Location Generic Icon	-20739-
Internet Location News Icon	-20738-
Internet Location Mail Icon	-20737-
Internet Location File Icon	-20736-
Internet Location AppleShare Icon	-20735-
Internet Location FTP Icon	-20734-
Internet Location HTTP Icon	-20733-
TrueType Flat Font Icon	-20732
IP File Server Icon	-20731
Recent Items Icon	-20730*

Recent Items Icon	-20730*
Favorites Folder Icon	-20729*
Favorite Items Icon	-20729"
Shortcut Icon	-20728
AppleTalk Zone Icon	-20727
AppleTalk Icon	-20726
Speakable Items Folder	-20724*
Appearance Folder Icon	-20723*
Help Icon	-20271*
Apple Menu Icon	-20270?
Clipboard Icon	-16509*8
System Suitcase Icon	-16494*8
Finder Icon	-16482*8
Generic Extension Icon	-16415-8
Apple Logo Icon	-16386*8
Generic Document Icon	-4000 *8
Generic Folder Icon	-3999 *8
Generic Floppy Icon	-3998 *8
Open Folder Icon	-3997 *8
Generic Application Icon	-3996 *8
Generic Hard Disk Icon	-3995 *8
Private Folder Icon	-3994 8
Trash Icon	-3993 *8
Desktop Icon	-3992 -8
Generic Desk Accessory Icon	-3991 8
Generic Edition File Icon	-3989 8
Generic RAM Disk Icon	-3988 8
Generic CDROM Icon	-3987 *8
Generic WORM Icon	-3987 "8
Generic Stationery Icon	-3985 -8
Full Trash Icon	-3984 *8
System Folder Icon	-3983 *8
Apple Menu Folder Icon	-3982 *8
Startup Items Folder Icon	-3981 *8
Shutdown Items Folder Icon	-3981 "8
Owned Folder Icon	-3980 -8
Drop Folder Icon	-3979 -8
Shared Folder Icon	-3978 -8
Mounted Folder Icon	-3977 -8
Control Panel Folder Icon	-3976 *8
Print Monitor Folder Icon	-3975 *8
Preferences Folder Icon	-3974 *8
Extensions Folder Icon	-3973 *8
Generic File Server Icon	-3972 8
Generic Preferences Icon	-3971 -8
Generic Suitcase Icon	-3970 -8
Generic Mover Object Icon	-3969 8
Fonts Folder Icon	-3968 *8
Generic Shared Libary Icon	-3967 8
Recent Documents Folder Icon	-3966 *8
Documents Folder Icon	-3966 "8

Recent Applications Folder Icon		-3965		*8
Applications Folder Icon		-3965		"8
Recent Servers Folder Icon	-3964		*8	
MacOS ReadMe Folder Icon	-3963		*8	
Control Strip Modules Folder Icon	-3962		*8	
Help Folder Icon		-3960		*8
Scripting Additions Folder Icon		-3959		*8
Internet PlugIn Folder Icon		-3958		8
Shared Libraries Folder Icon	-3956		8	
Text Encodings Folder Icon	-3955		*8	
Printer Driver Folder Icon		-3954		8
Printer Description Folder Icon		-3953		-8
Voices Folder Icon		-3952		-8
Control Panel Disabled Folder Icon	-3951		*8	
Extensions Disabled Folder Icon		-3950		*8
Startup Items Disabled Folder Icon	-3949		*8	
Shutdown Items Disabled Folder Icon		-3948		*8
System Extension Disabled Folder Icon		-3947		*8
Assistants Folder Icon	-3946		*8	
Utilities Folder Icon		-3945		*8
Application Support Folder Icon		-3944		*8
Apple Extras Folder Icon		-3943		*8
Contextual Menu Items Folder Icon	-3942		*8	
Generic Control Strip Module Icon	-3851		-8	
Generic Component Icon		-3850		8
Generic Control Panel Icon	-3824		-8	
Locked Icon		-3823		8
Alert Stop Icon		-3822		*8
Alert Caution Icon		-3821		*8
Alert Note Icon		-3820		*8
Generic Removable Media Icon		-3817		-8
Group Icon		-3816		8
Owner Icon		-3815		8
User Icon		-3814		8
Guest User Icon		-3813		8
Workgroup Folder Icon		-3812		8
User Folder Icon		-3811		8
Sound File Icon		-3810		-8
Internation Resources Icon	-3809		8	
TrueType Font Icon		-3808		-8
Generic Font Icon		-3807		8
Keyboard Layout Icon	-3806		8	
Font Suitcase Icon		-3804		-8
Clipping Sound Type Icon		-3803		-8
Clipping Text Type Icon		-3802		-8
Clipping Picture Type Icon		-3801		-8
Clipping Unknown Type Icon	-3800		-8	

No Folder Icon		-3784	8
No Write Icon	-3783	8	
No Files Icon		-3782	8
Protected System Folder Icon	-3774	8	
Protected Application Folder Icon	-3773	8	

Appendix C • Etcetera

Legal Note

The data structures of the new resource types defined by Kaleidoscope 2.0 ('wnd#' and 'cinf') are the intellectual property of the authors, and these resources may not be programmatically interpreted by sofware other than Kaleidoscope without prior written consent. In other words, you are welcome to edit these resources in a general purpose resource editor such as ResEdit or Resorcerer using the templates we provide, but if you write a program that reads these resources, you need our permission.

ResEdit Hack

ResEdit normally limits the size of its cicn's to between 8 and 64 pixels. However, Kaleidoscope 2.0 needs cicn's bigger than that, and in some cases you might want smaller ones to save space. So, here is a hack to let ResEdit set your cicn's to any size from 1 to 1024 pixels:

Use ResEdit to open up another copy of ResEdit, and open the RSSC 157 resource using the hex editor. Choose "Find Hex..." from the Find menu and specify:

Find Hex:
00404FEF00106E080C4500086D0276001E0376010C4600406E080C460008

Change To:
04004FEF00106E080C4500016D0276001E0376010C4604006E080C460001

Hit Change All and save. While you are messing with ResEdit, you should increase its RAM partition so that it can handle all the huge icons you will be throwing at it.

Speed Note

Tiling parts of icons, while it looks cool, can be slow, so if you want your scheme to run as fast as possible, use stretching instead of tiling wherever you can. For cicn's, turn off the Tile Sides option in the cinf resource whenever it is not needed. For windows, Kaleidoscope runs fastest if the stretch regions are only one pixel high or wide (so it can stretch the pixels instead of tiling them). If a one pixel stretch region does not fit into your window frame, you could try splitting a larger stretch region into two regions, making the first pixel the stretch region and the remainder a null region (part code 0) that does not draw.