Using Your Own Tip Function

When you use the **HMShowBalloon** and **HMShowMenuBalloon** functions to display help balloons, you pass a pointer to a tip function in the tipProc parameter. Normally, you supply NIL in this parameter to use the **Help Manager**'s default tip function. However, you can also supply your own tip function. The **Help Manager** calls your tip function after calculating the size and the location of a help balloon and before displaying it. This allows you to examine and, if necessary, adjust the balloon before it is displayed. For example, if you determine that the help balloon would obscure an object that requires extensive redrawing, you might use a different variation code to move the balloon.

Here's how to declare a tip function called MyTip.

```
pascal OSErr MyTip (Point tip, RgnHandle structure, Rect *r, short
                    *variant);
```

The **Help Manager** returns the balloon tip in the tip parameter, a handle to its region structure in the structure parameter, the content region in the r parameter, and the variation code to be used for the help balloon in the variant parameter. The content region is the area inside the balloon frame; it contains the user help information. The structure region is the boundary region of the entire balloon, including the content area and the pointer that extends from one of the help balloon's corners. (The third Figure in **Help Balloon Display** illustrates the structure regions of the eight standard help balloons.)

If the help balloon that **HMShowBalloon** or **HMShowMenuBalloon** initially calculates is not appropriate for your current screen display, you can make minor adjustments to it by specifying a different rectangle in the r parameter (the **Help Manager** automatically adjusts the structure parameter so that the entire balloon is larger or smaller as necessary) or by specifying a different variation code in the variant parameter. (the third Figure in the section **Help Balloon Display**. shows the different balloon positions assigned to the standard variation codes.)

If you need to make a major adjustment to the help balloon, return the hmBalloonAborted result code and call **HMShowBalloon** or **HMShowMenuBalloon** with appropriate new parameter values. To use the values returned in your tip function's parameters, return the noErr result code.

Here's an example of using a tip function to refrain from displaying a balloon if it obscures an area of the screen that requires extensive drawing.

```
#include <Balloons.h>

OSErr  UseMyTipProc()
{
    Rect            temprect;
```

```
    Rect            DontObscureRect;
    Point           tip;
    RgnHandle       structure;
    HMMessageRecord aHelpMsg;
    OSErr           rc;

    // be sure to determine DontObscureRect and fill in aHelpMsg
    rc = HMShowBalloon(&aHelpMsg, tip, nil, (Ptr)MyTip, 0, 0,
            kHMRegularWindow);
    if(rc == noErr) {
        // test whether balloon obscures graphic in DontObscureRect
        if(SectRect(&(**structure).rgnBBox, &DontObscureRect,
                &temprect))
            // do not show this balloon but call HMShowMenuBalloon later
            return (hmBalloonAborted);
        else
            return (noErr);
    }
}
```