The gadget also has the option of using `GMR_HELPCODE` instead of `GMR_HELPHIT`. `GMR_HELPCODE` is a little peculiar as a return value. Although `GMR_HELPCODE` is 32 bits long, Intuition identifies `GMR_HELPCODE` using only its upper 16 bits. If the upper 16 bits of `GM_HELPTEST`'s return value matches the upper 16 bits of `GMR_HELPCODE`, Intuition copies the lower word of the return value into the Code field of the `IDCMP_GADGETHELP` message. The Boopsi gadget is free to set the return value's lower word to any 16-bit value.

If the point is not within the gadget's ‘‘help spot’’, the gadget returns `GMR_NOHELPHIT`. Intuition will then look under that gadget for more gadgets. If a gadget's dispatcher passes the `GM_HELPTEST` method on to the gadgetclass dispatcher, the gadgetclass dispatcher will always return `GMR_HELPHIT`.

An application can put several windows in the same *help group*. This feature groups several windows so that as long as one of those windows is active, the application can receive `IDCMP_GADGETHELP` messages about all of those windows. For more information, See the HelpControl() Autodoc and the `WA_HelpGroup` section of the OpenWindow() Autodoc (both are in *intuition.doc*).

Boopsi Gadgets and ScrollRaster()

This section covers some fairly complicated aspects of Intuition Windows and how Intuition interacts with the Layers system. For a more in depth explanation, see the article, ‘‘Optimized Window Refreshing’’ from the July/August 1992 issue of *Amiga Mail*.

Scrolling an Intuition display with ScrollRaster() is unlike many other rendering operations because it can damage a window layer. ScrollRaster() is both a rendering function and a layering function (ScrollRaster() is a *graphics.library* function, but it is aware of the Layers system). If window X overlaps window Y, scrolling window Y can scroll a portion of window Y out from underneath window X:

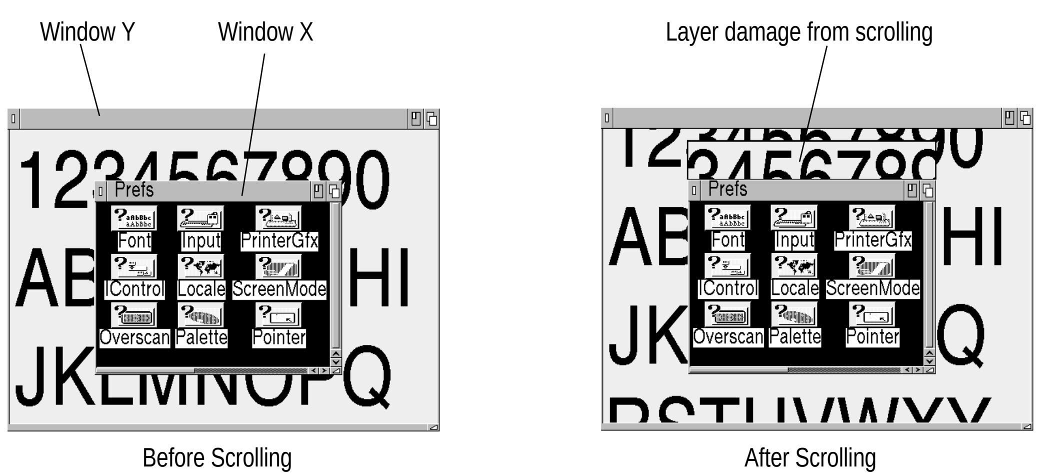

Figure 2 - Damage from a Scrolling Operation

For both smart refresh and super bitmap windows, this does not present a problem. Layers remembers what was underneath window X and restores that portion of the display when a layers operation reveals it. When a scrolling operation damages a smart refresh or super bitmap window, Layers repairs the display.

For a simple refresh window, Layers remembers which portion of the window layer is damaged. Layers does nothing to repair the damage. Layers leaves repairing the damage to Intuition. After performing a layers operation on a window (such as scrolling a portion of a window), Intuition must check if that operation caused layer damage (either to that window or other windows on the display).

In Release 2, when Intuition told a scrolling Boopsi gadget to render itself, Intuition did not check for layer damage. If that gadget damaged the display with the ScrollRaster() function, Intuition did not repair that damage.

As of Release 3.0, Intuition has a flag set aside in the MoreFlags field of the ExtGadget structure called `GMORE_SCROLLRASTER` (defined in <*intuition/intuition.h*>). If this flag is set, when the gadget damages the display with ScrollRaster() (or ScrollRasterBF()), Intuition redraws *all* `GMORE_SCROLLRASTER` gadgets in the damaged window. The class dispatcher should take care of setting this flag when creating the gadget in the `OM_NEW` method.

What is Layer Damage?

The Layers definition of "damage" is a little confusing. Looking at the illustration, two areas of window Y need to be refreshed: the area that used to be underneath window X, and the area that was scrolled into view at the bottom of window Y (the partially visible "RSTUVWXY"). Layers considers only one of these areas to be "damaged", the area the was scrolled out from underneath window X. Layers considers it damaged because the damage was caused by the presence of another window layer. The task that called ScrollRaster() has no idea that there is an overlapping window layer, so it is up to Layers to account for that damage. On the other hand, the task that called ScrollRaster() knows that the area at the bottom of window Y needs refreshing, so the task knows to fix that area of the window. To Layers, damage is the area of a layer that needs refreshing as a result of layers operation.

ScrollRaster() is not the only function in Release 3 that can scroll a window's contents. For Release 3, the Intuition library gained a function specifically to scroll a window's raster. That function, ScrollWindowRaster(), is similar to ScrollRaster(), but ScrollWindowRaster() is Intuition-friendly. Boopsi gadgets must not use this function, they should use ScrollRaster() or ScrollRasterBF() instead. Also any applications that use `GMORE_SCROLLRASTER` gadgets must use ScrollWindowRaster() for scrolling. Note that gadgets supplied by third parties or Commodore may be `GMORE_SCROLLRASTER` gadgets. For more information on ScrollWindowRaster(), see its Autodoc.