Acquiring the Clipboard

Let's begin by augmenting the GUI to include a row of buttons along the bottom of the main window for copying and pasting:

```
require 'fox16'
require 'customer'

include Fox

class ClipMainWindow < FXMainWindow
  def initialize(anApp)
    # Initialize base class first
    super(anApp, "Clipboard Example", :opts => DECOR_ALL, :width => 400, :height => 300)

    # Horizontal frame contains buttons
    buttons = FXHorizontalFrame.new(self, LAYOUT_SIDE_BOTTOM|LAYOUT_FILL_X|PACK_UNIFORM_WIDTH)

    # Cut and paste buttons
    copyButton  = FXButton.new(buttons, "Copy")
    pasteButton = FXButton.new(buttons, "Paste")

    # Place the list in a sunken frame
    sunkenFrame = FXVerticalFrame.new(self,
                          LAYOUT_FILL_X|LAYOUT_FILL_Y|FRAME_SUNKEN|FRAME_THICK, :padding => 0)

    # Customer list
    customerList = FXList.new(sunkenFrame, :opts => LIST_BROWSESELECT|LAYOUT_FILL_X|LAYOUT_FILL_Y)
    $customers.each do |customer|
      customerList.appendItem(customer.name, nil, customer)
    end
  end

  def create
    super
    show(PLACEMENT_SCREEN)
  end
end

if __FILE__ == $0
  FXApp.new("ClipboardExample", "FXRuby") do |theApp|
    ClipMainWindow.new(theApp)
    theApp.create
    theApp.run
  end
end
```

Note that the lines which appear in bold face are those which have been added (or changed) since the previous source code listing.

The clipboard is a kind of shared resource in the operating system. Copying (or cutting) data to the clipboard begins with some window in your application requesting "ownership" of the clipboard by calling the `acquireClipboard()` instance method. Let's add a handler for the "Copy" button press which does just that:

```
# User clicks Copy
copyButton.connect(SEL_COMMAND) do
  customer = customerList.getItemData(customerList.currentItem)
  types = [ FXWindow.stringType ]
  if acquireClipboard(types)
    @clippedCustomer = customer
  end
end
```

The `acquireClipboard()` method takes as its input an array of drag types. A *drag type* is just a unique value, assigned by the window system, that identifies a particular kind of data. In this case, we're using one of FOX's pre-registered drag types (`stringType`) to indicate that we have some string data to place on the clipboard. Later, we'll see how to register customized, application-specific drag types as well.

The `acquireClipboard()` method returns `true` on success; since we called `acquireClipboard()` on the main window, this means that the main window is now the clipboard owner. At this time, we want to save a reference to the currently selected customer in the `@clippedCustomer` instance variable so that if its value is requested later, we'll be able to return the *correct* customer's information.