Pasting Data from the Clipboard

We've seen one side of the equation, copying string data to the clipboard. But before we can "round-trip" that customer data and paste it back into another copy of our customer list application, we're clearly going to need to transfer the data in some more useful format. That is to say, if we were to receive the customer data in the format that it's currently stored on the clipboard:

```
#<struct Struct::Customer name="Joe Smith", address="123 Maple, Anytown, NC", zip=12345>
```

We'd have to parse that string and try to extract the relevant data from it. We can do better than that. The approach we'll use instead is to serialize and deserialize the objects using YAML. First, make sure that the YAML module is loaded by adding this line:

```
require 'yaml'
```

somewhere near the top of the program. Next, register a custom drag type for `Customer` objects. We can do that by adding one line to our main window's `create` instance method:

```
def create
  super
  @customerDragType = getApp().registerDragType("application/x-customer")
  show(PLACEMENT_SCREEN)
 end
```

Note that by convention, the name of the drag type is the MIME type for the data, but any unique string will do. In our case, we'll use the string "application/x-customer" to identify the drag type for our YAML-serialized `Customer` objects.

With that in place, we can now go back and slightly change some of our previous code. When we acquire the clipboard, we'd now like to be able to offer the selected customer's information either as plain text (i.e. the previous format) *or* as a YAML document, so we'll include *both* of these types in the array of drag types passed to `acquireClipboard()`:

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

Similarly, when we're handling the `SEL_CLIPBOARD_REQUEST` message, we now need to pay attention to which drag type (i.e. which data format) the requestor specified. We can do that by inspecting the `target` attribute of the `FXEvent` instance passed along with the `SEL_CLIPBOARD_REQUEST` message:

```
# Handle clipboard request
self.connect(SEL_CLIPBOARD_REQUEST) do |sender, sel, event|
  case event.target
    when FXWindow.stringType
```

```
      setDNDData(FROM_CLIPBOARD, FXWindow.stringType, Fox.fxencodeStringData(@clippedCustomer.to_s))
    when @customerDragType
      setDNDData(FROM_CLIPBOARD, @customerDragType, @clippedCustomer.to_yaml)
    else
      # Ignore requests for unrecognized drag types
  end
end
```

With these changes in place, we can now add a handler for the "Paste" button which requests the clipboard data in YAML format, deserializes it, and then adds an item to the customer list:

```
# User clicks Paste
pasteButton.connect(SEL_COMMAND) do
  data = getDNDData(FROM_CLIPBOARD, @customerDragType)
  if data
    customer = YAML.load(data)
    customerList.appendItem(customer.name, nil, customer)
  end
end
```

The `getDNDData()` method used here is the inverse of the `setDNDData()` method we used earlier to push data to some other application requesting our clipboard data. As with `setDNDData()`, the arguments to `getDNDData()` indicate the kind of data transfer we're performing (e.g. `FROM_CLIPBOARD`) and the drag type for the data we're requesting. If some failure occurs (usually, because the clipboard owner can't provide its data in the requested format) `getDNDData()` will simply return `nil`.