Introduction to the ClaroMac GUI Layout System

Part 1

This is a somewhat short intro into how Claro allows a person to configure a GUI.

You can use a coordinate system, like is frequently done in Windows and in some Linux GUI Toolkits, but then you're left with the problem of how to handle resize events.

One way is to render the window at a fixed, unchangeable size, which is fine for small utility programs, but for more complex programs this can be suboptimal.

Most, if not all, of the Linux GUI toolkits these days handle this by incorporating the "Box" concept, where the GUI objects are placed in various kinds of boxes. Based on how you've added the Object in the box, the Objects will resize or move into hopefully the desire position.

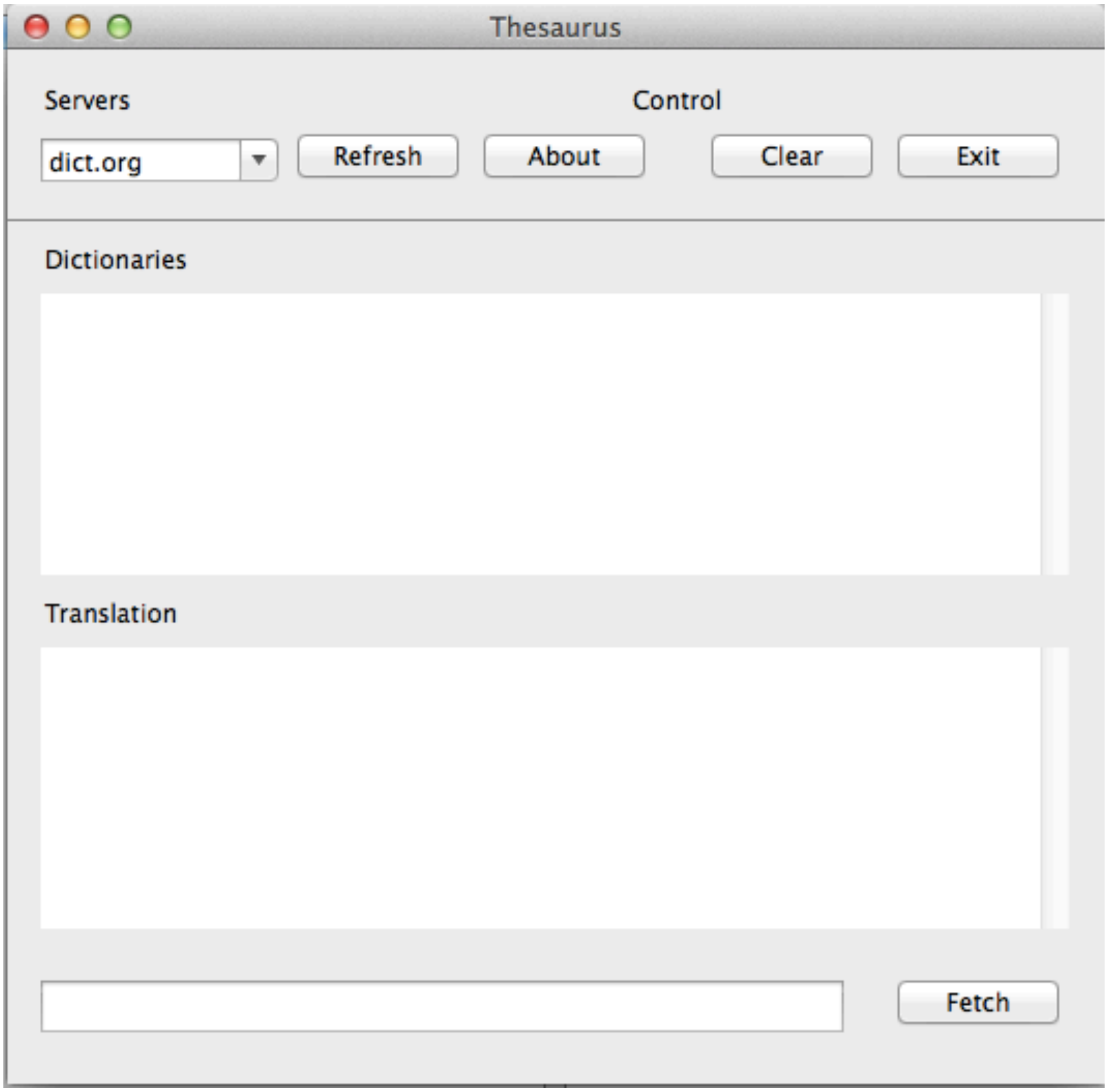
When coding using the Box paradigm manually, though, it can be difficult to visualize exactly how your code will "look" in the end.

Claro takes a different approach to this, ultimately making it easier to both conceptualize and create your layout.

The first thing you have to do is discard any thoughts of a coordinate system.

You won't be using or needing it...

Consider the following App, with the GUI portion written using ClaroMac:

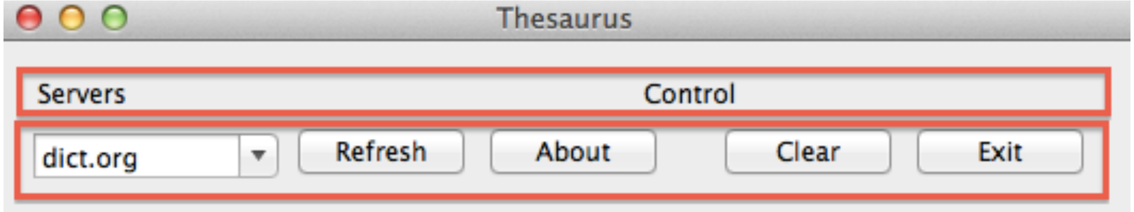


The layout code for the above looks like this:

```
CONST L1$ = "[{16}][(16)|servers|control<|(140)]"
CONST L2$ = "[(16)|drop|>fetch|>about|(20)|>clear|>exit|(16)]"
CONST L3$ = "[{16}][{1}line][{10}]"
CONST L4$ = "[(16)|label3][_(16)|list|(16)] [{10}][(16)|label4|(16)]
[_(16)|text|(16)][]"
CONST L5$ = "[(16)|entry|(20)|>dflt|(16)][]"
```

At first glance, I'm sure that's confusing so let me break down the first line (L1$) for you.

One thing to keep in mind that should make this clear in a moment is that in Claro, it helps to break down the GUI interface into horizontal blocks, working top -> down.



dict.org

So in the first "block", the first two *identifiable* items we see are the LABELS "Servers" and "Control". That's one horizontal block, or "line" if it's easier to think in those terms.

The second horizontal block consists of the Combobox and the "Refresh/About/Clear/Exit" Buttons. That corresponds to "L2$" in the example above.

In each, you should see text "Identifiers", which let's you see at a glance what is contained in a given block. Again referencing the top block, in "L1$" we find "servers|controls" which correspond to the to LABELS I spoke of earlier.

With that out of the way, let me now break down "L1$"

In Claro/ClaroMac, the brackets ("[]"") signify the beginning of a new horizontal block.

So the very first thing we see in "L1$" is "[{16}]".

Now if you're following me ok up to this point, you'll realize that the first bracket is being used to give us some space from the top of the Window.

Otherwise, the GUI Objects will butt up right to the top.

So what does the "{16}" *within* the brackets signify, you might ask?

Here, we are setting the actual height of that initial spacer-bracket, in this case 16 pixels. So the "[{16}]" is how we tell the Gui that we want a 16 pixel border from the top.

The next part of "L1$" consists of:

```
[(16)|servers|control<|(140)]
```

This is where the actual LABELS are being laid out on the Window. Let's go through each part in sequence.

The first entry, "(16)", is telling Claro that we want a 16 pixel border on the left. In Claro, "{..}" is used to control Height, and "(..)" is used to control Width. So in this case, we want to include a left-border that is 16 pixels wide.

Next, we have "|servers|control<". Note that each "item" within the square-brackets "[..]" is separated by "|" which is used to group items within a block.

So we have the label "servers", which is undecorated (more on that in a moment) followd by the "controls" label.

In Claro, when an item is not decorated (either by a size or the angle brackets), that item will expand horizontally by default. So even though you don't see it in this label, the "servers" label is in fact expanding when the window is resized to a larger size.

The "controls" label, on the other hand, *is* decorated, with an angle bracket.

What this means is that the label will NOT expand horizontally, and it will anchored to the right side of the block. Without the following item, "(140)", it would remain anchored in that position . This last bit is pretty cool because what we are doing is padding the right side of the block with a width of 140 pixels, thus preventing the "controls" label from shifting right. So in this case, it will always be positioned between the "About" and "Clear" buttons, no matter what else is going on in the form.

"L2$" marks the start of a new horizontal block. Again, we begin by creating a horizontal padding of 16 pixels like we did in the previous block.

```
[(16)|drop|>fetch|>about|(20)|>clear|>exit|(16)]
```

"drop" above is the Identifier I chose for the combobox. As you can see, it is undecorated so it will expand horizontally as the window is resized.

"fetch" and "about" are decorated with the angle-bracket, which means that they won't expand but *will* slide towards the right side of the window when it's resized.

"(20)" by itself is providing a 20 pixel gap between the "about" button and the "clear" button. This is how one would go about fine-tuning positioning between objects.

Like "fetch" and "about", "clear" and "exit" are decorated with the angle-bracket, preventing them from resizing but allowing them to slide right.

Lastly, we create a padding of 16 pixels on the *right* side of the block.

That pretty much covers the beginning of what one would need to know to use Claro's Layout system. I still need to cover creation of the actual Layout **objects** that drive this, but that will come in the next installment.

April 5, 2014, AIR.