Using script colorizing and formatting to help debugging

by Sarah Reichelt

As well as the conventional debugging tools like the Variable Watcher, Script Debugger, and Message Watcher, Revolution offers another very easy way to check your scripts for simple errors and typos: Turn on Script Colorization!

Yes, it really is that simple, but once you have become accustomed to the colors, you can tell at a glance whether you have used a reserved word for a variable name, or whether you have misspelled one of the internal property names. This is particularly important if you are copying & pasting scripts from another xTalk language. Revolution has so many more reserved words, that it is easy to insert an invalid variable name.

Look at this example script - copy and paste it into a script editor window:

```
on mouseUp
  answer file "Select a file:"
  put it into fileName
  put the lenght of fileName into len
  put 255 into max
  if len > max then
    answer "Sorry - filename too long!"
  end if
  end mouseUp
```

At first glance, you may see nothing wrong with this, but now go to the ? Script? menu and choose "Colorize":

```
on mouseUp
  answer file "Select a file:"
  put it into fileName
  put the lenght of fileName into len
  put 255 into max
  if len > max then
    answer "Sorry - filename too long!"
  end if
end mouseUp
```

(If you have changed any of the color preferences, you may see different colors here, but these are the defaults.)

This shows us several errors:

- fileName is shown in red, so it must be a reserved word - a property - so we can't use it as a variable name
- len & max are both shown in orange, so they are also reserved words - functions - so again using them as variable names is wrong
- in the 4th line, I misspelled the word "length" - if I had got it right, it should have been orange, but as it was not colored, that tells me that the script compiler didn't recognize it.

There is one catch when using internal functions: in the example above, I used (or tried to use) "the length of fileName", but this can also be written "length(fileName)". If you have "Live Colorization" turned on, the word "length' will turn orange when you type the last character BUT when you add the bracket, it will revert to black. If you leave a space before the opening bracket, the function name will stay orange, but I have heard reports of functions not working if called like that, so you'll just have to test what works for you.

"Live Colorization" can be turned on in Preferences or in the "View" menu of the Script Editor. The specific colors can be set using "Colorization Settings..." in the Script Editor's "View" menu, where you can also define your own words for coloring.

The other way to check your scripts quickly is to use "Format" which indents the lines of your scripts, to show the start and end of logical blocks: handlers, functions, if ... end if, repeat loops etc. Unlike colorization, formatting is always turned on but it doesn't always update itself correctly. If this happens, either press Tab to force a re-formatting of the current handler, or choose "Format" from the "Script" menu to re-format the entire script.

The quickest way to spot a problem is to look at the last line of your handler. If the "end" is not at the left edge of the script editor or if there is more than one line at the left edge, then you have a problem!

Here are some examples of script problems that can be found using format (I increased the indent distance to make it easier to see):

1. I used "exit repeat" instead of "end repeat":

```
on mouseUp
  repeat with x = 1 to 100
    put x
    exit repeat
end mouseUp
```

2. Here I forgot the ?then? at the end of the ?else if? line, making the ? end if? line too far to the left.

```
on mouseUp
  if field "Age" < 18 then
    answer "You can't vote yet"
  else if fld "Age" > 65
    answer "You can retire"
  end if
end mouseUp
```

3. Here's an easy mistake to make when nesting multiple "if" statements. I meant the "else" to apply to the first "if", but the script compiler has applied it to the second, so then I seem to be missing an "end if".

```
on mouseUp
  if field "Year" < 2000 then
    if fld "Month" > 6 then answer "After June"
  else
    answer "Welcome to the new millenium"
  end if
end mouseUp
```

The solution is to explicitly end the inner loop with it's own "end if" before going to the "else".

One point to note is that using script colorising does increase the size of your stack because it stores the colored version as well as the plain version. This is not usually a problem because when you build an application, Revolution removes the colored version along with any extra temporary development information. However if you want to upload your stacks to a web site, email them or store them on a small drive, it may become significant. In that case, I recommend using Chipp Walter's AltCleanStack.rev available at http://www.altuit.com/webs/altuit2/altPluginDownload/Downloads.htm

This removes the extra copies of the scripts as well as any other temporary properties stored by Revolution in your stack. It will not stop your stack functioning and the script colors can be restored at any time, but it is recommended that you clean a backup stack rather than your working copy.