The MaxGUI Beginner Tutorial Series - Tutorial 16: The TextArea Gadget
(c) Assari Feb 8 2006

This gadget is extensively used by the MaxIDE and I think as a result has the requisite functionality to power a heavy application. In keeping with the beginner theme of this tutorial we will not be wanting to tap too much into its extreme capability :)

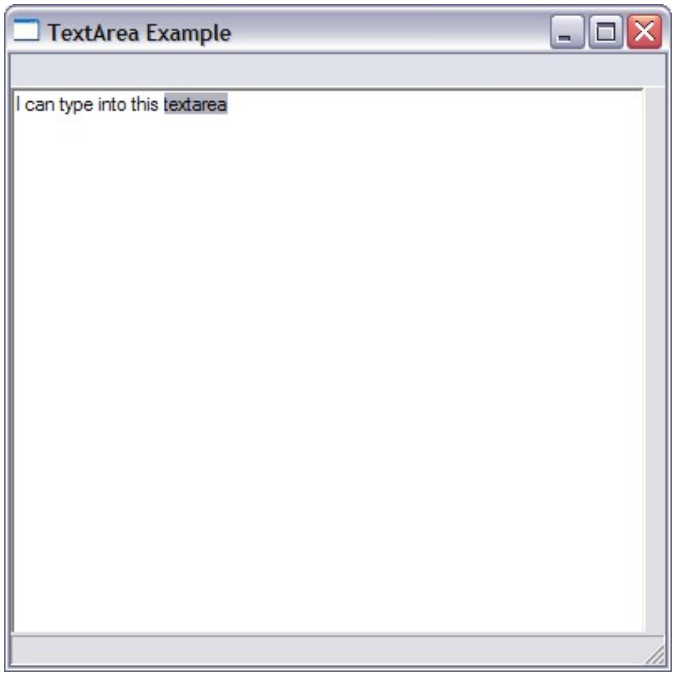
As this tutorial is rather long, I'm providing a table of contents here to help navigate through this, although as per usual there is a logic to the sequence the functionalities are presented.

Creating a TextArea Gadget using CreateTextArea

As usual, lets start with the simple and proceed up the learning curve. The first function to start with is of course the **CreateTextArea** function.

```
SuperStrict

Local MyWindow:TGadget=CreateWindow("TextArea Example", 40,40,400,400)
Global MyText:TGadget=CreateTextArea(0,0,380,360,MyWindow)

Repeat
  WaitEvent()
  Select EventID()
  Case EVENT_WINDOWCLOSE
    End
  End Select
Forever
End
```

Running the above application will give us a small area where we can start typing things in. It sort of feel a little bit like Notepad.



Adding Text into TextArea Gadget using AddTextAreaText

There are several ways to add text to our text area. The obvious way, as we did above, is to type text directly into the textarea. The other way is to programmatically enter it using the **AddTextAreaText** function like so.

```
SuperStrict

Local MyWindow:TGadget=CreateWindow("TextArea Example", 40,40,400,400)
Global MyText:TGadget=CreateTextArea(0,0,380,360,MyWindow)

AddTextAreaText(MyText,"The Quick Brown Fox ")
AddTextAreaText(MyText,"Jumps Over The Lazy Dog.")

Repeat
  WaitEvent()
  Select EventID()
  Case EVENT_WINDOWCLOSE
    End
  End Select
Forever
End
```

We can now see the text from the above two lines appearing in our TextArea.

Note that the CreateTextArea function accepts two styles, TEXTAREA_WORDWRAP and TEXTAREA_READONLY whose functionality should be obvious.

```
Function CreateTextArea:TGadget(x,y,w,h,group:TGadget,style=0)
```

Replacing text in a TextArea Gadget using SetTextAreaText

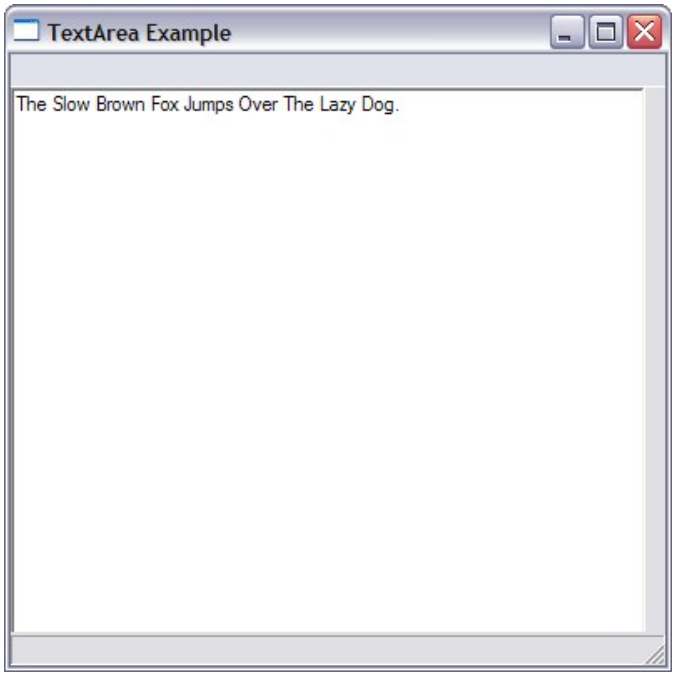
Another interesting function is **SetTextAreaText** which allows us to insert/replace text in our TextArea.

```
Function SetTextAreaText(
textarea:TGadget,text$,pos=0,length=TEXTAREA_ALL,units=TEXTAREA_CHARS )
```

Using it as shown below causes the the word **Quick** (a 5 letter word at the 4th position) to be replaced by the word **Slow**.

```
SuperStrict

Local MyWindow:TGadget=CreateWindow("TextArea Example", 40,40,400,400)
Global MyText:TGadget=CreateTextArea(0,0,380,360,MyWindow)

AddTextAreaText(MyText,"The Quick Brown Fox ")
AddTextAreaText(MyText,"Jumps Over The Lazy Dog.")
SetTextAreaText(MyText,"Slow",4,5)

Repeat
  WaitEvent()
  Select EventID()
  Case EVENT_WINDOWCLOSE
    End
  End Select
Forever
End
```

Notice the word Slow instead of the word Quick in our TextArea .

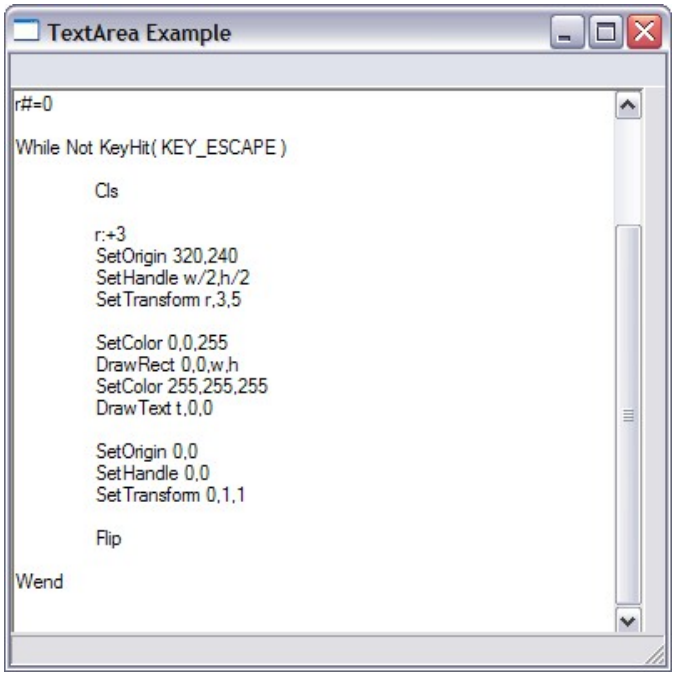


Adding text into a TextArea Gadget from file using ReadStream and AddTextAreaText

A common way to add text into the TextArea is to load a text file from the hard disk. The following program actually does this by reading in one of the sample program into the text area. We then read in each line and insert the text into our TextArea via the **AddTextAreaText** function. The trick there is to also include a "~n" which causes each line to wrap to the next line. The **ReadStream** function opens the file to be read and the **Readline** function reads in a line at a time.

```
SuperStrict

Local MyWindow:TGadget=CreateWindow("TextArea Example", 40,40,400,400)
Global MyText:TGadget=CreateTextArea(0,0,380,360,MyWindow)

Local In:Tstream=ReadStream(Blitzmaxpath()+"/samples/spintext
/spintext.bmx")
While Not Eof(In)
    Local text:String=ReadLine(In)
    AddTextAreaText(MyText,Text+"~n")
Wend
CloseStream(In)

Repeat
  WaitEvent()
  Select EventID()
  Case EVENT_WINDOWCLOSE
     End
  End Select
Forever
End
```

Notice also how a vertical scrollbar now appears to accomodate the fact that the number of lines of text far exceed that height of the TextArea. This is automatically done by MaxGUI

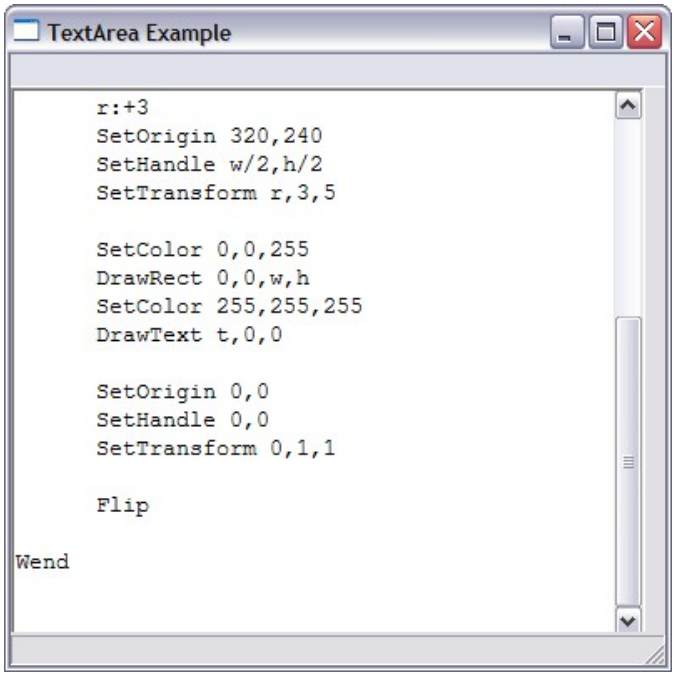


Changing the font in a TextArea Gadget using SetTextAreaFont

Let us see how we can make our TextArea to be looking more like the MaxIDE. First let us change the font to a more programmer friendly font of **Courier New**. We first need to load the font via the **LoadGUIFont** function and then use the loaded font to set the font via the **SetTextAreaFont** function. Note that we have to set the font before we start adding text to our text area for that font to take effect.

```
SuperStrict

Local MyWindow:TGadget=CreateWindow("TextArea Example", 40,40,400,400)
Global MyText:TGadget=CreateTextArea(0,0,380,360,MyWindow)

Local GUIFont:TGuiFont=LoadGuiFont( "Courier New",12)
SetTextAreaFont(MyText,GUIFont)

Local In:Tstream=ReadStream(Blitzmaxpath()+"/samples/spintext/spintext.bmx")
While Not Eof(In)
   Local text:String=ReadLine(In)
   AddTextAreaText(MyText,Text+"~n")
Wend
CloseStream(In)

Repeat
  WaitEvent()
  Select EventID()
  Case EVENT_WINDOWCLOSE
     End
  End Select
Forever
End
```

Notice now our font has changed to the desired Courier font of point (size) 12.



Changing the font and background color of a TextArea Gadget using SetTextAreaColor

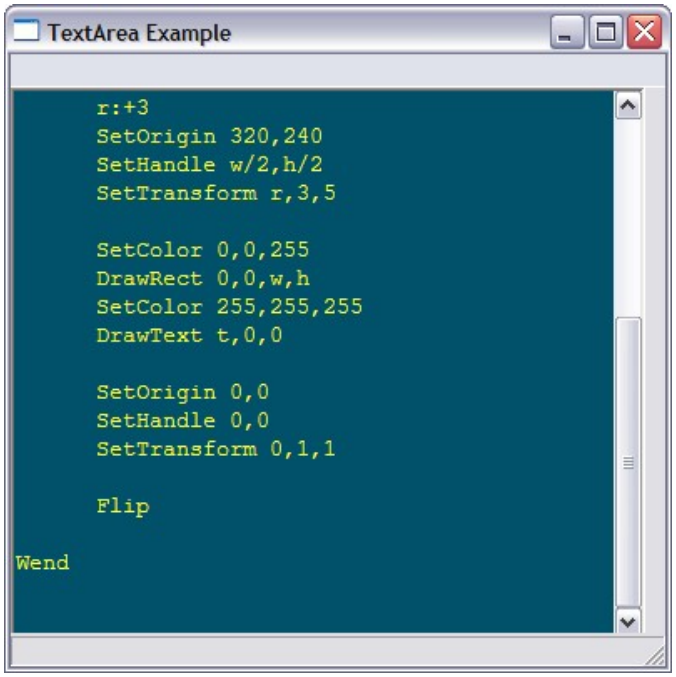
We can change the background and font color by using the **SetTextAreaColor** twice, once to change the background and the second time to change the font (also called foreground in the helpfiles)

```
SuperStrict

Local MyWindow:TGadget=CreateWindow("TextArea Example", 40,40,400,400)
Global MyText:TGadget=CreateTextArea(0,0,380,360,MyWindow)

Local GUIFont:TGuiFont=LoadGuiFont( "Courier New",12)
SetTextAreaFont(MyText,GUIFont)

Local In:Tstream=ReadStream(Blitzmaxpath()+"/samples/spintext/spintext.bmx")
While Not Eof(In)
   Local text:String=ReadLine(In)
   AddTextAreaText(MyText,Text+"~n")
Wend
CloseStream(In)

SetTextAreaColor(MyText,1,81,107,True)
SetTextAreaColor(MyText,255,255,50,False)

Repeat
  WaitEvent()
  Select EventID()
  Case EVENT_WINDOWCLOSE
     End
  End Select
Forever
End
```

The syntax for the SetTextAreaColor is

```
Function SetTextAreaColor( textarea:TGadget,r,g,b,bg=False )
```

Where r,g,b is the red, green and blue component of the desired color and bg being true or false depending on whether we want to set the foreground or background color. Our TextArea now is more colorful.



Counting the number of characters and text in a TextArea Gadget using TextAreaLen

Let us now look at some other interesting functions that we can use on our TextArea.

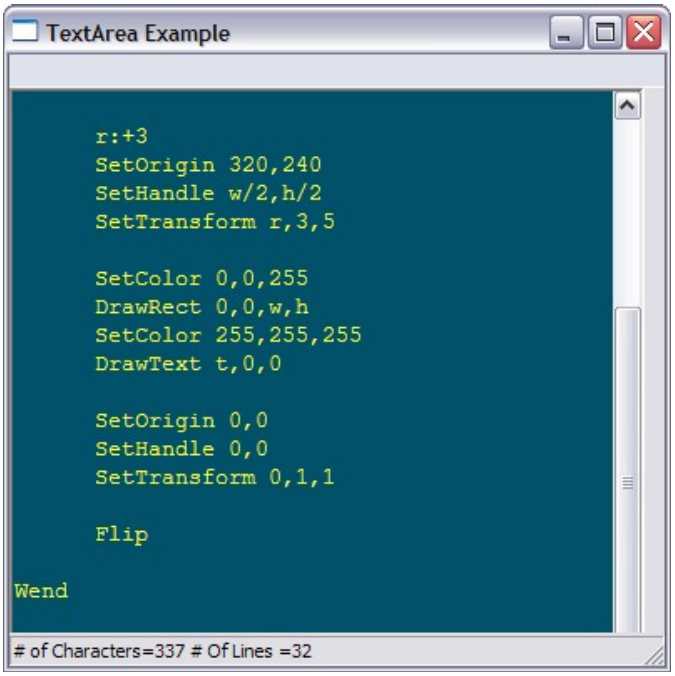
We can perform some statistical function on our TextArea by using the **TextAreaLen**gth function which has the following syntax:-

Function TextAreaLen(textarea:TGadget,units=TEXTAREA_CHARS)

```
SuperStrict

Local MyWindow:TGadget=CreateWindow("TextArea Example", 40,40,400,400)
Global MyText:TGadget=CreateTextArea(0,0,380,360,MyWindow)

Local GUIFont:TGuiFont=LoadGuiFont( "Courier New",12)
SetTextAreaFont(MyText,GUIFont)

Local In:Tstream=ReadStream(Blitzmaxpath()+"/samples/spintext/spintext.bmx")
While Not Eof(In)
   Local text:String=ReadLine(In)
   AddTextAreaText(MyText,Text+"~n")
Wend
CloseStream(In)

SetTextAreaColor(MyText,1,81,107,True)
SetTextAreaColor(MyText,255,255,50,False)

Repeat
  WaitEvent()
  Select EventID()
  Case EVENT_WINDOWCLOSE
    End
  End Select

  Local char:Int=TextAreaLen(MyText,TEXTAREA_CHARS)
  Local lines:Int=TextAreaLen(MyText,TEXTAREA_LINES)
```

```
    SetStatusText MyWindow,"# of Characters="+char+" # Of Lines ="+Lines

  Forever
  End
```

We can now see a running total of the number of characters and lines in our TextArea. Go ahead and key in some text and insert a couple of lines you will see the text on the status line changing.



Manipulating Text in a TextArea Gadget using TextAreaText

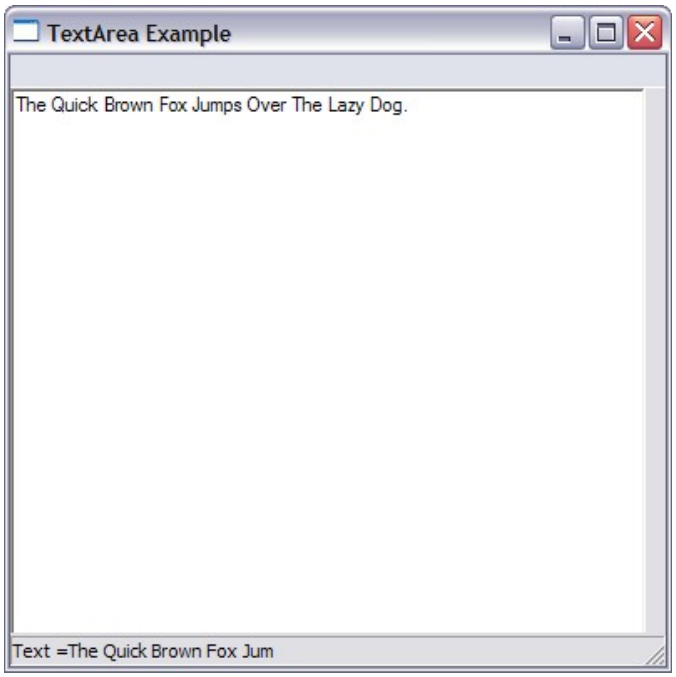
TextAreaText is a very useful function in that it allows us to retrieve text from our TextArea gadget for further manipulation within our Blitzmax program.

```
Function TextAreaText$(
textarea:TGadget,pos=0,length=TEXTAREA_ALL,units=TEXTAREA_CHARS )
```

To illustrate how we can use this function, let us modify one of our earlier program as below and at the same time introduce ourselves to yet another useful function, the **TextAreaCursor** function.

```
  SuperStrict

  Local MyWindow:TGadget=CreateWindow("TextArea Example", 40,40,400,400)
  Global MyText:TGadget=CreateTextArea(0,0,380,360,MyWindow)

  AddTextAreaText(MyText,"The Quick Brown Fox ")
  AddTextAreaText(MyText,"Jumps Over The Lazy Dog.")
  Local cursorpos:Int

  Repeat
    WaitEvent()
    Select EventID()
    Case EVENT_WINDOWCLOSE
      End
    Case EVENT_GADGETSELECT
      cursorpos=TextAreaCursor(MyText)
    End Select
    SetStatusText MyWindow, "Text ="+TextAreaText(MyText,0,cursorpos)
  Forever
  End
```

Go ahead and run the above program and click on any part of the text and watch the text in the status display changing. The text starting from the start (pos=0) until the cursor position, where we click, is retrieved and displayed in the status bar. The cursor position is obtained via the **TextAreaCursor** function.



Notice also the event that we look for to catch an action in the TextArea, the **EVENT_GADGETSELECT**.

```
Case EVENT_GADGETSELECT
  cursorpos=TextAreaCursor(MyText)
```

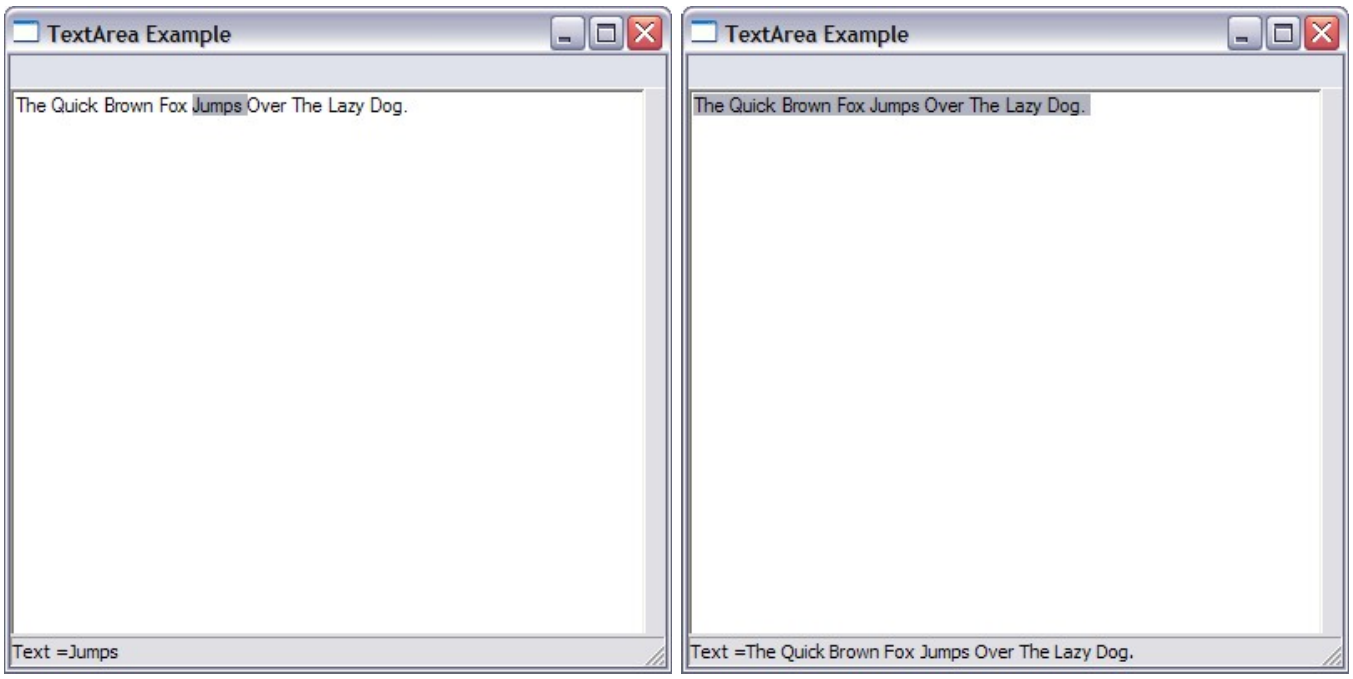
Selecting a word (or a line) in a TextArea Gadget using TextAreaText

The TextArea gadget has some inherent functionality whereby a double click on any text in the TextArea will cause the word under the cursor to be selected and triple-clicking will cause the whole line to be selected. Let us see how we can make use of this behaviour:-

```
SuperStrict

Local MyWindow:TGadget=CreateWindow("TextArea Example", 40,40,400,400)
Global MyText:TGadget=CreateTextArea(0,0,380,360,MyWindow)

AddTextAreaText(MyText,"The Quick Brown Fox ")
AddTextAreaText(MyText,"Jumps Over The Lazy Dog.")
Local cursorpos:Int
Local SelectedLength:Int

Repeat
  WaitEvent()
  Select EventID()
  Case EVENT_WINDOWCLOSE
    End
  Case EVENT_GADGETSELECT
    cursorpos=TextAreaCursor(MyText)
    SelectedLength=TextAreaSelLen(MyText)
  End Select
  SetStatusText MyWindow, "Text ="+TextAreaText(MyText,cursorpos,SelectedLength)
Forever
```

```
End
```

We are now introduced to a new function call **TextAreaSelLen** which returns the length of the selected text in the TextArea gadget. The **TextAreaCursor** function now returns the first position of the selected text instead of the cursor position previously.



If you manually select text by dragging the cursor across the text, you can also see the text in the statusbar changing accordingly.

Changing the format of text in a TextArea Gadget using FormatTextAreaText

Now we are going to get a bit more ambitious and show how we would implement some of the actions you see in the MaxIDE, namely keyword highlighting.

To do this, we need to know about the FormatTextAreaText function which has the following syntax:-

```
Function FormatTextAreaText(
textarea:TGadget,r,g,b,flags,pos=0,length=TEXTAREA_ALL,units=TEXTAREA_CHARS
```

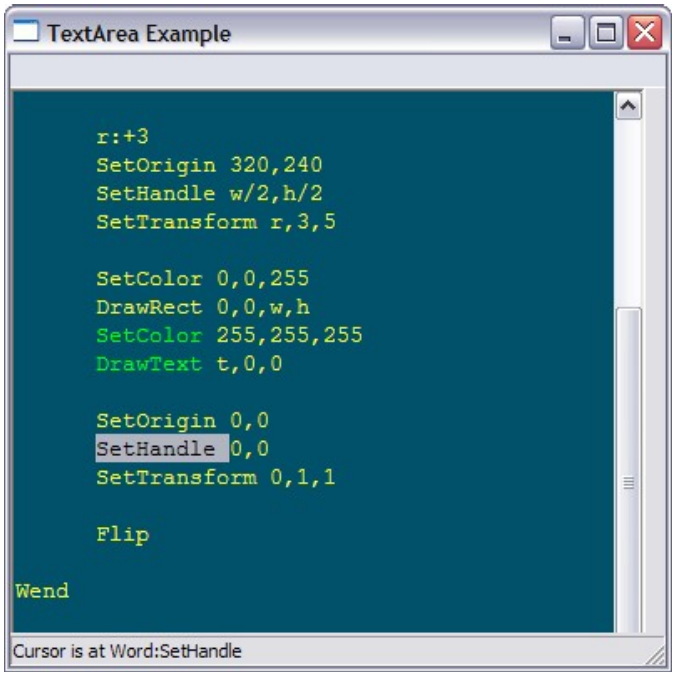
This function allows us to format selected parts of text via the r,g,b part of the function parameters and some form of formatting via the flags (you will need to experiment with this to find out what they are, the manual is silent on this)

So let us see how we might implement a very simple syntax highlighting using this function. For the sake of simplicity we will be looking for the word Set and assumes any word with this in it is a valid syntax.
The line changed is only one, see bold line below (do not forget that the two dots is a line continuation)

```
SuperStrict

Local MyWindow:TGadget=CreateWindow("TextArea Example", 40,40,400,400)
Global MyText:TGadget=CreateTextArea(0,0,380,360,MyWindow)

Local GUIFont:TGuiFont=LoadGuiFont( "Courier New",12)
SetTextAreaFont(MyText,GUIFont)

Local In:Tstream=ReadStream(Blitzmaxpath()+"/samples/spintext/spintext.bmx")
While Not Eof(In)
   Local text:String=ReadLine(In)
```

```
    AddTextAreaText(MyText,Text+"~n")
  Wend
  CloseStream(In)

  SetTextAreaColor(MyText,1,81,107,True)
  SetTextAreaColor(MyText,255,255,50,False)
  Local cursorpos:Int
  Local SelectedLength:Int
  Local MyWord:String
  Local Keyword:String="SetColor |DrawText |"

  Repeat
   WaitEvent()
   Select EventID()
   Case EVENT_WINDOWCLOSE
     End
  Case EVENT_GADGETSELECT
    cursorpos=TextAreaCursor(MyText)
    SelectedLength=TextAreaSelLen(MyText)
    MyWord=TextAreaText(MyText,cursorpos,SelectedLength)
    If Instr(KeyWord,MyWord+"|") Then ..
       FormatTextAreaText(MyText,0,255,0,0,Cursorpos,SelectedLength)
   End Select

   SetStatusText MyWindow,"Cursor is at Word:"+MyWord

  Forever
  End
```

When we compile and run the above program, we can click on a text in the TextArea and the word under the cursor will be highlighted. The status bar will also display the word under the cursor.



At the moment our program only recognises two keywords, go ahead and modify to program so that it can recognise more keywords. Perhaps you may want to change the color it highlights to by changing the rgb values.

Summary

As you can see, the TextArea gadget is a very powerful and flexible gadget. For those of us using the default MaxIDE, we use the TextArea gadget extensively everytime we program in BlitzMax.

I will not summarise the functionalities this time as they are available at the top of this tutorial :)

That's all for now. Return to Main Index.