The Wayback Machine - http://web.archive.org/web/20070125214652/http://www.2dgamecreators...
The MaxGUI Beginner Tutorial Series - Tutorial 1: Using CreateWindow
(c) Assari Dec 22 2005

Introduction

This is the first of what I hope will be an extensive series of beginner level tutorials for MaxGUI, a BlitzMax extension which adds GUI capability to this excellent language.

These tutorials are pitched at beginners although I do expect some familiarity with using Blitzmax, the Editor MaxIDE, cutting&pasting code into the IDE , compiling and running BlitzMax programs. I'm also assuming some basic programming knowledge such as variables, loops and conditionals.

Note that these tutorials have been (will be) created on a Windows version of MaxGUI so I have no idea how they will look and behave on other systems. I'm assuming that the MaxGUI cross-platform works as claimed.

Using CreateWindow
Lets try out a simple MaxGUI blitzmax program which will create a window with a title called "My Window Title is Hello World"

```
MyFirstWindow:TGadget=CreateWindow("My Window Title is Hello World",
200,200,320,240)

Repeat
 WaitEvent()
Until EventID()=EVENT_WINDOWCLOSE

End
```

Cut and paste the above code into the MaxIDE and run the program. You should see the following (for Windows XP users):-

Go ahead and play with the window. It behaves as a normal window. You can move it, resize it, minimize it and finally close it.

Now lets try and see what all those lines of Blitzmax code means

```
MyFirstWindow:TGadget=CreateWindow("My Window Title is Hello World",
200,200,320,240)
```

The above code creates a window and displays it on the computer screen. The central piece here is the MaxGUI function **CreateWindow.**

In BlitzMax, just like other programming languages, its functions needs parameters. In the case of **CreateWindow** it minimally needs the following:-

- The text that should appear on the created window title bar

- Where to place the created window on the computer screen ie its coordinates
- How big the window should be ie its size

In the example above, the given parameters are:-

- Title text is **My Window Title is Hello World**
- Display location at coordinates **200,200**
- Window size of **320** by **240** pixels

Functions by their nature also returns a result. **CreateWindow** returns an object of type **TGadget** which we have chosen to call **MyFirstWindow**.

Objects are a big part of Blitzmax but we will not get into that for this tutorial, suffice to know that we must always supply a name of the Object that CreateWindow needs to return stuff to.

For now I will not explain in detail the rest of the code except to say that the code allows the program to wait until we close the created window by clicking on the close window icon.

```
Repeat
 WaitEvent()
Until EventID()=EVENT_WINDOWCLOSE

End
```

If we were to look at the Blitzmax Helpfile we will see the following syntax requirements

```
Function CreateWindow:TGadget(name$,x,y,w,h,group:TGadget=Null,style=15)
```

Lets go into more detail of the syntax:-

- The word **Function** is just to tell us that CreateWindow is a function. Nothing special here
- **CreateWindow:TGadget** tells us that the CreateWindow function will return an Object of type TGadget
- The items between brackets are parameters, lets go through them one by one:-
  - **name$** is the text that will appear in the title bar
  - **x,y** is the x and y coordinates of the location where the window will appear
  - **w,h** is the **w**idth and **h**eight of the window (ie its size)
  - **group:TGadget** is the group to whom this created window belongs to. Will get into this later in the tutorial
  - **style** is the style of the window. How it would appear etc. More later

Two of the parameters **group:TGadget** and **style** are written slightly differently then the rest of the parameters viz **group:TGadget=null** and **style=15** i.e. they have assignments.

Whenever you see an assignment in function parameters, this means that the parameter is optional and has default value as assigned. In this case **group** will be **null** (ie it does not belong to any other Gadget) and **style** will be **15** which gives you a window as displayed in our first example.

The CreateWindow Style

Lets play a little bit with this style parameter. MaxGUI has a set of predefine constants to help with setting the window style. From the Blitzmax Helpdocuments:-

Constant	Meaning
WINDOW_TITLEBAR	The Window has a titlebar that displays it's name
WINDOW_RESIZABLE	The Window can be resized by the user
WINDOW_MENU	The Window has a menubar
WINDOW_STATUS	The Window has a statusbar
WINDOW_TOOL	The Window is rendered on some platforms with a reduced titlebar
WINDOW_CLIENTCOORDS	The dimensions specified relate to the client area not the window frame

	WINDOW_HIDDEN	The Window is created in a hidden state
	WINDOW_ACCEPTFILES	Enable drag and drop operations

Lets see what happens to our window when we start setting the style with a different constant value

```
MyFirstWindow:TGadget=CreateWindow("My Window with
WINDOW_TITLEBAR", 200,200,320,240,Null,WINDOW_TITLEBAR)

Repeat
 WaitEvent()
Until EventID()=EVENT_WINDOWCLOSE

End
```

Note that I have also changed the title to reflect what we are doing. Cut and paste the above code and run the resulting program.

Now compare the two windows. A couple of obvious differences are the lack of min/max buttons and status line. The behaviour of the window is also now different. You cannot resize or minimize it anymore.

MaxGUI allows you to create windows with different combination of styles. In does this by using a special function called | which is pronounced OR.

For example to have a window with a title and make it resizable we do the following

```
MyFirstWindow:TGadget=CreateWindow("My Window with WINDOW_TITLEBAR",
200,200,320,240,Null,WINDOW_TITLEBAR|WINDOW_RESIZABLE)

Repeat
 WaitEvent()
Until EventID()=EVENT_WINDOWCLOSE

End
```

We can now see that the newly created window can be resized, minimized and maximized.

As a matter of taste I prefer to use a local variable to define the style so that my **CreateWindow** function does not look so messed up viz:-

```
Local style =
WINDOW_TITLEBAR|WINDOW_RESIZABLE|WINDOW_MENU|WINDOW_STATUS

MyFirstWindow:TGadget=CreateWindow("My Window with WINDOW_TITLEBAR",
200,200,320,240,Null,style)

Repeat
 WaitEvent()
Until EventID()=EVENT_WINDOWCLOSE

End
```

The variable style contains the style number to use. Coincidently if we were to see what the value of this variable is like so:-

```
Local style =
WINDOW_TITLEBAR|WINDOW_RESIZABLE|WINDOW_MENU|WINDOW_STATUS
Print "The Value of Style is "+style

MyFirstWindow:TGadget=CreateWindow("My Window with WINDOW_TITLEBAR",
200,200,320,240,Null,style)

Repeat
 WaitEvent()
Until EventID()=EVENT_WINDOWCLOSE

End
```

You can see from the MaxIDE **Output** window that the value of the variable **style** is 15

Its no wonder than that this created window is the same as the one we saw for our first program. As stated in the help file "...the default window style is WINDOW_TITLEBAR | WINDOW_RESIZABLE | WINDOW_MENU | WINDOW_STATUS. " which happens to be 15 (see Function Defintion above, style=15)

WINDOW_CLIENTCOORDS
One other style which is interesting to look at is **WINDOW_CLIENTCOORDS**. Let's see what this does

```
Local style = WINDOW_TITLEBAR | WINDOW_RESIZABLE | WINDOW_MENU | WINDOW_STATUS | WINDOW_CLIENTCOORDS

MyFirstWindow:TGadget=CreateWindow("My Window with WINDOW_TITLEBAR", 200,200,320,240,Null,style)

Repeat
 WaitEvent()
Until EventID()=EVENT_WINDOWCLOSE

End
```

Cut and paste the above program into the MaxIDE and run it. Compare this window with the ones we had before

What we now see is that the size of 320 x 240 that we have specified in the **CreateWindow** parameter is actually now specifying the size of the central panel (the client) of this window. The other items like the status bar, the menu bar, the borders as well as the titlebar now is added on top of the central panel thus resulting in a bigger window being created.

WINDOW_STATUS
The WINDOW_STATUS constant as we have seen earlier allows for the creation of the status bar at the bottom of the window.

An example of the use of the status bar is as follows:-

```
Local style = WINDOW_TITLEBAR | WINDOW_RESIZABLE | WINDOW_STATUS |
WINDOW_CLIENTCOORDS

MyFirstWindow:TGadget=CreateWindow("My Window with WINDOW_TITLEBAR",
200,200,320,240,Null,style)

Repeat
 WaitEvent()
  SetStatusText MyFirstWindow, "My current size (w,h) is " + ClientWidth(MyFirstWindow)+
"," + ClientHeight(MyFirstWindow)

Until EventID()=EVENT_WINDOWCLOSE

End
```

Cut and Paste the above code and run it. You should see the following screen

Notice the text appearing in the status bar. Now resize the window and you should see the size changing once the window has been resized.

We have achieved this via this line of code.

```
SetStatusText MyFirstWindow, "My current size (w,h) is " + ClientWidth(MyFirstWindow)+ "," 
+ ClientHeight(MyFirstWindow)
```

SetStatusText has the following syntax:-

```
Function SetStatusText( window:TGadget,text$ )
```

It sends the text as defined in text$ to the Window Gadget as defined in window. So in our example above it sends the text **"My current size (w,h) is "** to the window called **MyFirstWindow**.

Now I also ask MaxGUI to calculate the most current size of the window client-area by using another MaxGUI function called **ClientHeight** and **ClientWidth** whose functionality I hope is self explanatory. A clearer way to do this would be as follows:-

```
Local style = WINDOW_TITLEBAR | WINDOW_RESIZABLE | WINDOW_STATUS | 
WINDOW_CLIENTCOORDS

MyFirstWindow:TGadget=CreateWindow("My Window with WINDOW_TITLEBAR", 
200,200,320,240,Null,style)

Repeat
 WaitEvent()
  Local text$= "My current size (w,h) is " + ClientWidth(MyFirstWindow)+ "," + 
ClientHeight(MyFirstWindow)
  SetStatusText MyFirstWindow, text$

Until EventID()=EVENT_WINDOWCLOSE

End
```

That is all for our very first tutorial. I'm hoping that this is giving you an easy introduction into the world of MaxGUI. The other styles such as WINDOW_HIDDEN and WINDOW_ACCEPTFILES are a bit more advance styles so we will not cover them here. Feel free to experiment with the other styles and the various combinations possible.

Before we go, lets recap what we have learnt:-

- We learnt about the **CreateWindow** function and its required parameters
- We covered a few styles and showed their different impacts on the created window
- We also used some other MaxGUI functions such as **SetStatusText**, **ClientWidth** and **ClientHeight**

In the next tutorial we will be covering buttons and introducing ourselves to the concept of events. Until then, take care :)