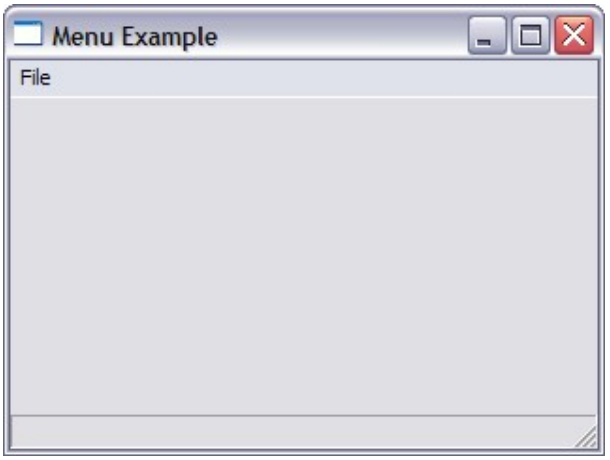
The MaxGUI Beginner Tutorial Series - Tutorial 14: Menus and Popup Menus
(c) Assari Jan 07 2005

Menus are another common GUI gadget. In this tutorial we are going to cover menus. Lets get started as usual with a very simple exmaple.

```
SuperStrict

Local MyWindow:TGadget=CreateWindow("Menu Example", 200,200,320,240)
Local FileMenu:TGadget=CreateMenu("File",0,WindowMenu(MyWindow))
UpdateWindowMenu MyWindow

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

We can see our **File** menu appearing on the menubar but we cannot do anything with it yet.



The requirement to create a menu and use it in our window is a bit more onerous that other gadgets we have seen previously. At the very least, we need to specify the following:-

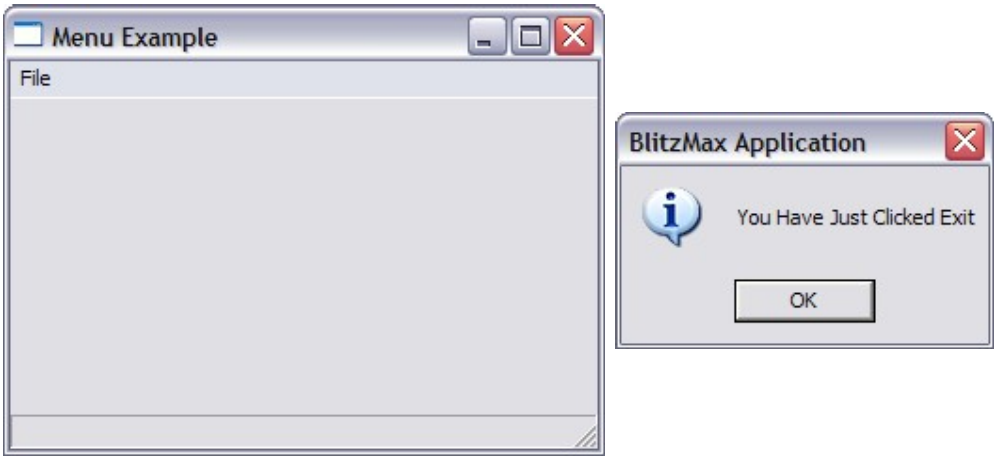
- A text label which will be displayed on the Menu bar
- A unique tag id number for this menu item
- A parent which uses the **WindowMenu** function
- Then we need to create a second level menu item (see example **CreateMenu "Exit"** below)
- Before the menu can be used, we need to call the **UpdateWindowMenu** function, otherwise our menu will not appear

In our example above, clicking the Menu actually does not do anything. Let's do something more exciting.

```
SuperStrict
```

```
Local MyWindow:TGadget=CreateWindow("Menu Example", 200,200,320,240)
Local FileMenu:TGadget=CreateMenu("&File",0,WindowMenu(MyWindow))
CreateMenu "Exit",101,FileMenu

UpdateWindowMenu MyWindow

Repeat
 WaitEvent()
 Select EventID()
 Case EVENT_WINDOWCLOSE
   End
 Case EVENT_MENUACTION
  Select EventData()
   Case 101
    Notify "You Have Just Clicked Exit"
    End
   End Select
  End Select
Forever
```

Running the above program will get us a File Menu with an Exit sub-menu, clicking it activate the Notify message and then ends the program.



Notice how we create the **Exit** menu which has the **FileMenu** gadget as its parent. A unique identifier 101 was used for this Menu Item. Any unique id will work as well. Never forget to call the **UpdateWindowMenu** function.

```
Local FileMenu:TGadget=CreateMenu("&File",0,WindowMenu(MyWindow))
CreateMenu "Exit",101,FileMenu

UpdateWindowMenu MyWindow
```

When the user clicks the menu item, a MenuAction event is generated, we then need to check which menu item has been clicked by checking against the unique identifier specified earlier in the **CreateMenu** function.

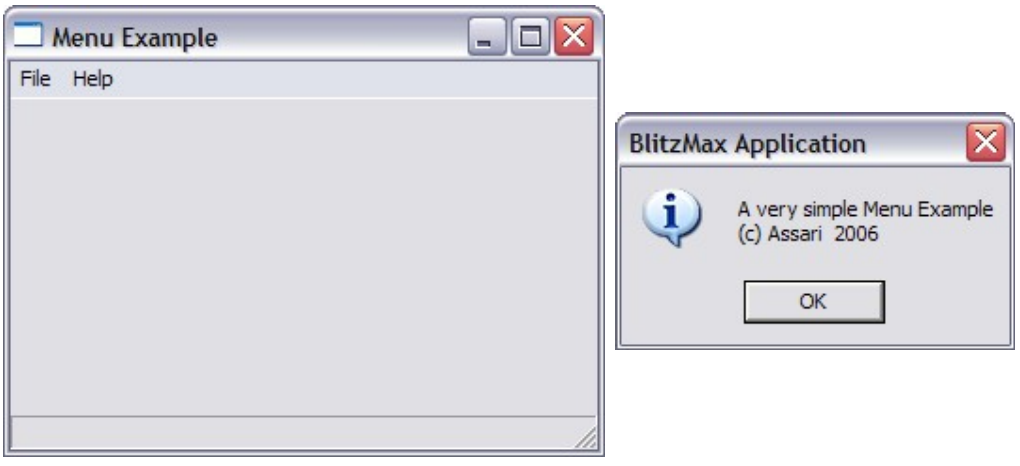
```
 Case EVENT_MENUACTION
  Select EventData()
   Case 101
```

```
      Notify "You Have Just Clicked Exit"
      End
    End Select
```

Let us see how we can cope with more menu items.

```
SuperStrict

Local MyWindow:TGadget=CreateWindow("Menu Example", 200,200,320,240)
Local FileMenu:TGadget=CreateMenu("File",0,WindowMenu(MyWindow))
CreateMenu "Open",102,FileMenu
CreateMenu "Exit",101,FileMenu

Local HelpMenu:TGadget=CreateMenu("Help",0,WindowMenu(MyWindow))
CreateMenu "About",123456,HelpMenu

UpdateWindowMenu MyWindow

Repeat
 WaitEvent()
 Select EventID()
 Case EVENT_WINDOWCLOSE
   End
 Case EVENT_MENUACTION
  Select EventData()
   Case 101
    Notify "You Have Just Clicked Exit"
    End
   Case 102
    Notify "You Have Just Clicked OPEN"
   Case 123456
    Notify "A very simple Menu Example~n(c) Assari  2006"

   End Select
  End Select
Forever
```

Run the program and click on Help About and you would see the Notify message as we have defined above in our program.



The first line below creates the **File** Menu and then we create the 2 sub-menu items **Open** and **Exit**

```
Local FileMenu:TGadget=CreateMenu("File",0,WindowMenu(MyWindow))
CreateMenu "Open",102,FileMenu
CreateMenu "Exit",101,FileMenu
```

Next we create the **Help** Menu and the **About** sub-menu item

```
Local HelpMenu:TGadget=CreateMenu("Help",0,WindowMenu(MyWindow))
CreateMenu "About",123456,HelpMenu
```

We then use the Select/Case statement on the **EventData**() function which actually returns the unique identifer which we have defined in the respective **CreateMenu** function above

```
Case EVENT_MENUACTION
 Select EventData()
  Case 101
   Notify "You Have Just Clicked Exit"
   End
  Case 102
   Notify "You Have Just Clicked OPEN"
  Case 123456
   Notify "A very simple Menu Example~n(c) Assari  2006"

  End Select
```

So there you go, building more menu is just an extension of the above method.

The syntax for using the **CreateMenu** function is as follows:-

```
Function CreateMenu:TGadget( text$,tag,parent:TGadget,hotKey=0,modifier=0 )
```

Let us now take a look at the **hotkey** parameter. This parameter allows you to attach a "hotkey" to the menu ie pressing the assigned key will cause the menu item to be triggered.

In the example below, we are now attaching the F4 ket to our File Exit menu item. Note the use of the constant **KEY_F4** in the **CreateMenu** function below

```
SuperStrict

Local MyWindow:TGadget=CreateWindow("Menu Example", 200,200,320,240)
Local FileMenu:TGadget=CreateMenu("&File",0,WindowMenu(MyWindow))
CreateMenu "Exit",101,FileMenu,KEY_F4

UpdateWindowMenu MyWindow

Repeat
 WaitEvent()
```

```
  Select EventID()
  Case EVENT_WINDOWCLOSE
    End
  Case EVENT_MENUACTION
   Select EventData()
    Case 101
      Notify "You Have Just Clicked Exit"
      End
    End Select
   End Select
Forever
```

Now run the above program and then press the F4 key. You will see the Notify message coming out just as if you have clicked on the File Exit menu item.

The next parameter we want to look at is the **modifier** parameter. What the modifier does is allow us to use the CTRL or ALT keys in conjuction with our assigned key ie CTRL-F4 to exit instead of just using the F4 key.

```
SuperStrict

Local MyWindow:TGadget=CreateWindow("Menu Example", 200,200,320,240)
Local FileMenu:TGadget=CreateMenu("&File",0,WindowMenu(MyWindow))
CreateMenu "Exit",101,FileMenu,KEY_F4,  MODIFIER_COMMAND

UpdateWindowMenu MyWindow

Repeat
  WaitEvent()
  Select EventID()
  Case EVENT_WINDOWCLOSE
    End
  Case EVENT_MENUACTION
   Select EventData()
    Case 101
      Notify "You Have Just Clicked Exit"
      End
    End Select
   End Select
Forever
```

Now run the above program and then press the CTRL-F4 key. You will see the Notify message coming out just as if you have clicked on the File Exit menu item.

A list of keys and modifiers can be found in the helpfiles.

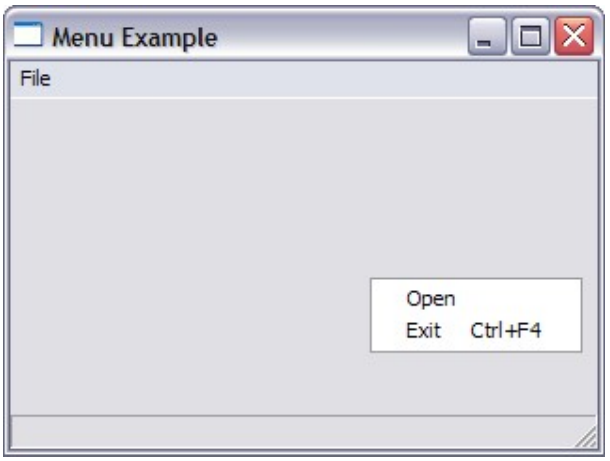
POP UP MENU

MaxGUI also allows the creation of a pop menu when the user clicks the right-mouse button. Let us see how to do this

```
SuperStrict

Local MyWindow:TGadget=CreateWindow("Menu Example", 200,200,320,240)
Local MyPanel:TGadget=CreatePanel(0,0,300,200,MyWindow,PANEL_ACTIVE)
```

```
Local FileMenu:TGadget=CreateMenu("File",0,WindowMenu(MyWindow))
CreateMenu "Open",102,FileMenu
CreateMenu "Exit",101,FileMenu

UpdateWindowMenu MyWindow

Repeat
 WaitEvent()
 Select EventID()
 Case EVENT_WINDOWCLOSE
   End

 Case EVENT_MOUSEDOWN
  If EventSource()=MyPanel And EventData()=2 Then
    PopupWindowMenu MyWindow,FileMenu
  EndIf

 Case EVENT_MENUACTION
  Select EventData()
   Case 101
    Notify "You Have Just Clicked Exit"
    End
   Case 102
    Notify "You Have Just Clicked OPEN"

   End Select
  End Select
Forever
```

Right-clicking on the panel now causes the File menu to pop up at our mouse location.



First we need to have gadgets which can handle mouse events. In this example we are going to use the panel gadget. We just need to remember to create the panel gadget with the PANEL_ACTIVE style so that mouse events can be captured.

```
Local MyPanel:TGadget=CreatePanel(0,0,300,200,MyWindow,PANEL_ACTIVE)
```

Then we need to catch the mouse event usent the **WaitEvent**() and **Select/Case** statements. We then need to check that the mouse event came from our panel

(**EventSource**()=MyPanel) and also that the right mouse button has been clicked (**EventData**()=2). Once both of these have been satisfied, we can then call the **PopupWindowMenu** function.

```
Case EVENT_MOUSEDOWN
 If EventSource()=MyPanel And EventData()=2 Then
    PopupWindowMenu MyWindow,FileMenu
 EndIf
```

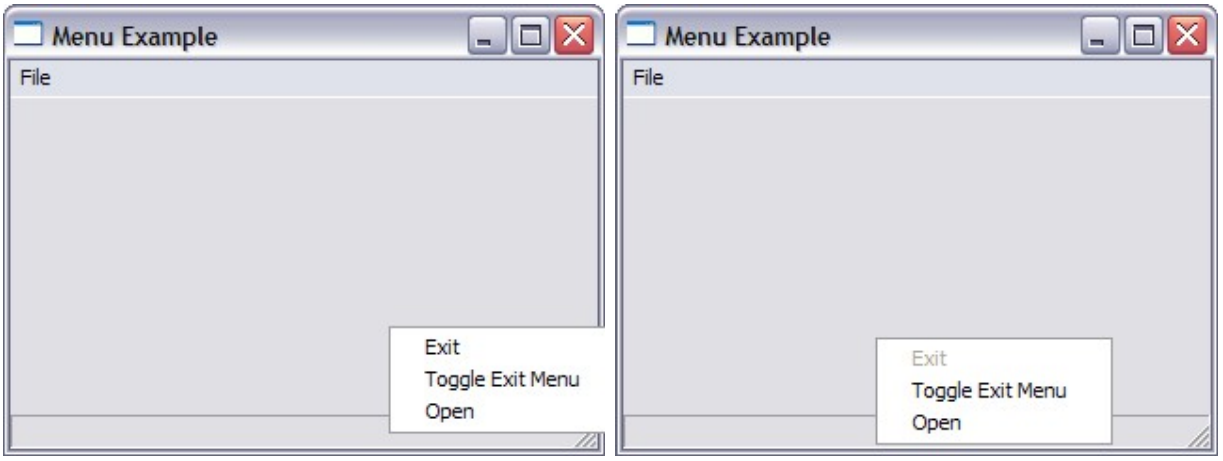
DisableMenu

There are several functions that can be used to manipulate menu items. We will not cover all of them here. One function we will cover is DisableMenu which as the name suggests disables a menu item. Let us see how that works

```
SuperStrict

Local MyWindow:TGadget=CreateWindow("Menu Example", 200,200,320,240)
Local MyPanel:TGadget=CreatePanel(0,0,300,200,MyWindow,PANEL_ACTIVE)
Local FileMenu:TGadget=CreateMenu("File",0,WindowMenu(MyWindow))

Local ExitMenu:TGadget=CreateMenu("Exit",101,FileMenu)
CreateMenu "Toggle Exit Menu",103,FileMenu
CreateMenu "Open",102,FileMenu

UpdateWindowMenu MyWindow

Repeat
  WaitEvent()
  Select EventID()
  Case EVENT_WINDOWCLOSE
     End

  Case EVENT_MOUSEDOWN
   If EventSource()=MyPanel And EventData()=2 Then
      PopupWindowMenu MyWindow,FileMenu
   EndIf

  Case EVENT_MENUACTION
   Select EventData()
     Case 101
       Notify "You Have Just Clicked Exit"
       End
     Case 102
       Notify "You Have Just Clicked OPEN"
     Case 103
       If MenuEnabled(ExitMenu)=1
          DisableMenu ExitMenu
       Else
          EnableMenu ExitMenu
       EndIf
       UpdateWindowMenu MyWindow
```

```
    End Select
  End Select
Forever
```

From the screenshots below we can see our menu being disabled after we have clicked on the toggle menu item. The disabled menu item has been greyed out.



The first thing we need to do is to have a name for the menu-item that we want to disable. For the other menu-items we have seen that this was not necessary

```
Local ExitMenu:TGadget=CreateMenu("Exit",101,FileMenu)
```

When the **Toggle Exit Menu** was clicked, the menu action event will return the 103 identifier. We then check for the state of the menu item via the **MenuEnabled** function. Based on this we either enable or disable our menu item via the **DisableMenu** and **EnableMenu** function.

```
Case 103
 If MenuEnabled(ExitMenu)=1
   DisableMenu ExitMenu
 Else
   EnableMenu ExitMenu
 EndIf
 UpdateWindowMenu MyWindow
```

Note that the UpdateWindowMenu function must always be called after we make changes to our menu

Summary

Menus are fairly simple gadgets and as we can see from above are fairly simple to use as well.

To recap what we have learnt so far

- We create menu using the **CreateMenu** function.
- Menus need to be activated using the UpdateWindowMenu function before they can

be used.
- Menus can either be triggered by clicking on them or by hotkeys which we assign via the hotkey and modifier parameter of the **CreateMenu** function.
- Popup menus can be used via the **PopupWindowMenu** function
- The menu events are captured via the menu action events and the relevant menu item identifier captured via the **eventdata**() function.
- Menus can be enabled or disabled via the the **EnableMenu** and **DisableMenu** functions respectively

So thats ends our tutorial for now. Back to Tutorial Index.