15.5 DesktopWatcher

Die Klasse *DesktopWatcher* (gb.desktop) implementiert ein Objekt, das Ereignisse des Fenster-Managers überwacht.

15.5.1 Eigenschaften

Die Klasse *DesktopWatcher* besitzt nur die eine Eigenschaft *.RootWindow* (Boolean) und gibt True zurück, wenn nur das RootWindow überwacht werden soll oder setzt mit True das Root-Window als zu beobachtendes Fenster.

15.5.2 Ereignisse

Methoden besitzt die Klasse nicht – aber acht Ereignisse. Das ist verständlich, denn die Klasse hat vor allem überwachende Aufgaben.

Ereignis	Beschreibung
ActiveWindow()	Das Ereignis wird ausgelöst, wenn sich das aktuelle, aktive Fenster geändert hat.
Change()	Das Ereignis wird ausgelöst, wenn sich der aktuelle virtuelle Desktop geändert hat.
Count()	Das Ereignis wird ausgelöst, wenn sich die Anzahl der virtuellen Desktops geändert hat.
Geometry()	Das Ereignis wird ausgelöst, wenn sich die Größe eines Desktops verändert hat.
WindowGeometry(W As DesktopWindow)	Das Ereignis wird ausgelöst, wenn das angegebene Fenster in der Größe geändert oder verschoben wurde.
WindowIcon (W As DesktopWindow)	Das Ereignis wird ausgelöst, wenn sich das Fenster-Icon verändert hat.
WindowName (W As DesktopWindow)	Das Ereignis wird ausgelöst, wenn sich der Name oder der angezeigte Name des Fensters geändert hat.
WindowState(W As DesktopWindow)	Das Ereignis wird ausgelöst, wenn sich der Status des angegebenen Fensters geändert hat. *State* kann mehrere Werte annehmen.
Windows()	Das Ereignis wird ausgelöst, wenn sich die Liste der Fenster geändert hat, weil ein Fenster zerstört oder erzeugt wurde oder wenn sich die Reihenfolge in der existierenden Liste verändert hat.

Tabelle 15.5.2.1: Übersicht ausgewählter Events der Klasse DesktopWatcher

15.5.3 Projekt

Das vorgestellte Projekt wurde 2012 von *Richard Walker* als *WindowExplorer* entwickelt. Die Adaption beschränkt sich auf die Demonstration des Zusammenspiels der Klassen DesktopWindow, Desktop → *Kapitel 15.1.0* und DesktopWatcher. Der Quelltext wurde um einige Funktionen erweitert – erfordert dann aber, dass das Programm 'wmctrl' installiert sein muss.

```
' Gambas class file

Private aDTWindowsList As DesktopWindow[]
Private aDTWindowsList2 As Integer[]
Public DTWatcher As DesktopWatcher

Public Sub _new()
  DTWatcher = New DesktopWatcher(True) As "MyDTWatcher"
  txbPatternBox.Text = "*"
End ' _new()

Public Sub Form_Open()
  FMain.Center

  GetWindowInfo()
  GetDesktopInfo()
  GetWindowsList()
End ' Form_Open()
```

```
Public Sub GetWindowsList()
  Dim iCount As Integer

  txaWindowList.Clear
  aDTWindowsList = Desktop_FindWindow(txbPatternBox.Text)
  lblWindowCount.Text = aDTWindowsList.Count
  For iCount = 0 To aDTWindowsList.Count - 1
      txaWindowList.Text &= Str(iCount + 1) & "\t" & (1 + aDTWindowsList[iCount].Desktop)
      txaWindowList.Text &= "      " & aDTWindowsList[iCount].Id & "\t"
      txaWindowList.Text &= aDTWindowsList[iCount].Name & gb.NewLine
  Next ' iCount
  aDTWindowsList = Null

End ' btnGetWindowsList1_Click()

Public Function GetCurrentDesktopName() As String
  Dim sOutput, sZeile, sElement As String
  Dim aMatrix As String[]

' Ausgabe der Namen der virtuellen Desktops (VDesktop)
  Exec ["wmctrl", "-d"] To sOutput
  For Each sZeile In Split(sOutput, gb.NewLine)
      If InStr(sZeile, "*") Then ' Der aktuelle Desktop wird durch ein * gekennzeichnet
         aMatrix = Split(sZeile, " ")
         Return aMatrix[aMatrix.Max]
      Endif ' InStr(sZeile, "*") ?
  Next ' FOR EACH sZeile
End ' Function GetDesktopName()

Public Sub GetDesktopInfo()
  lblDesktopCount.Text = Str(Desktop.Count)
  lblDTCurrentValue.Text = Str(Desktop.Current + 1)
  lblDesktopType.Text = Desktop.Type
  lblCurrentDesktopName.Text = GetCurrentDesktopName()
End ' GetDesktopInfo()

Public Sub GetActiveWindow()
  Dim DTWindow As DesktopWindow

  lblActiveWindowValue.Text = Str(Desktop.ActiveWindow)
  DTWindow = New DesktopWindow(Val(lblActiveWindowValue.Text))
  lblActiveWindowName.Text = DTWindow.Name
End ' GetActiveWindow()

Public Sub GetWindowInfo()
  lblRootWindowID.Text = Str(Desktop.RootWindow)
  GetActiveWindow()
End ' GetWindowInfo()

Private Function Desktop_FindWindow(sPattern As String) As DesktopWindow[]
  Dim DTWindow As DesktopWindow
  Dim DTWList As New DesktopWindow[]

  For Each DTWindow In Desktop.Windows ' Desktop.Windows enthält die Liste aller Fenster
      If DTWindow.Name Like sPattern Then
         DTWList.Add(DTWindow)
      Endif ' DTWindow.Name Like sPattern ?
  Next ' DTWindow
  Return DTWList
End ' Function Desktop_FindWindow(..)

Public Function SetTime() As String
  Return Format(Now(), "hh:nn:ss") & "\t"
End ' SetTime()

Public Sub MyDTWatcher_ActiveWindow()
  txaDTWatcherEvents.Text &= SetTime() & "* Das aktive Fenster hat sich geändert!" & gb.NewLine
  GetActiveWindow()
End ' MyWindowWatcher_ActiveWindow()

Public Sub MyDTWatcher_Windows()
  txaDTWatcherEvents.Text &= SetTime() & "~ Die Fenster-Liste hat sich geändert." & gb.NewLine
  GetWindowsList()
End ' MyDTWatcher_Windows()

Public Sub btnRefresh_Click()
  GetWindowInfo()
  GetDesktopInfo()
  GetWindowsList()
End ' btnRefresh_Click()

Public Sub btnEnde_Click()
  FMain.Close
End ' btnEnde_Click()
```

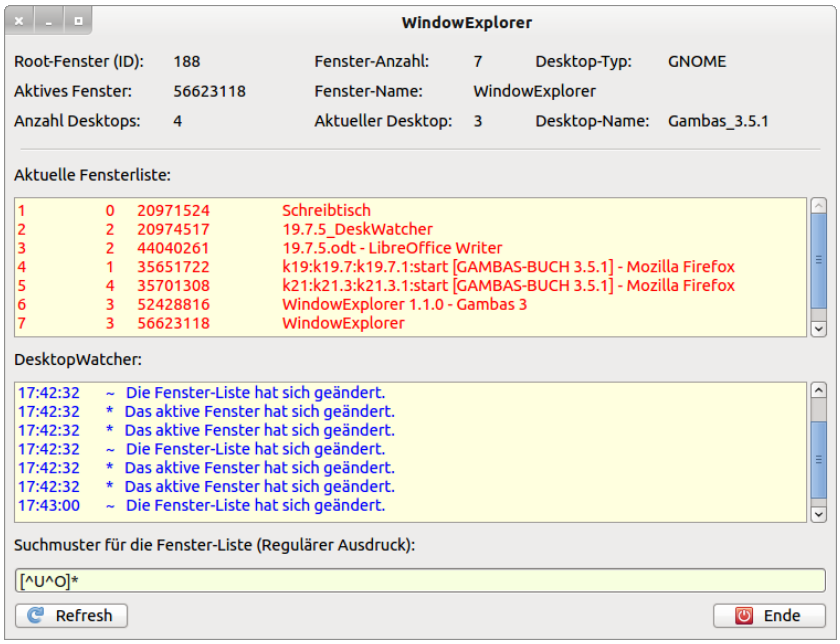

Abbildung 15.5.3.1: WindowExplorer mit geändertem Suchmuster

Es ist schon beachtlich, welche Vielfalt an Informationen das Programm anzeigt. Der Refresh-Button wird benötigt zur Anzeige der Fensterliste bei geändertem Suchmuster → Abbildung 19.7.5.3.1 und nach einer Verschiebung des Programmfensters auf einen anderen (virtuellen) Desktop, damit der aktuelle Desktop mit Nummer und Namen neu abgefragt und angezeigt werden kann.

Diese sechs Prozeduren könnten Sie dem WindowExplorer noch hinzufügen. Der Quelltext muss noch erweitert werden, da Sie die Werte der Parameter der ersten drei Prozeduren bereitstellen müssen.

```
Public Sub MyDTWatcher_WindowGeometry(w As DesktopWindow)
  txaDTWatcherEvents.Text &= SetTime() & w.Name & ": VDesktop-Geometrie geändert." & gb.NewLine
End ' MyDTWatcher_WindowGeometry(w As DesktopWindow)

Public Sub MyDTWatcher_WindowIcon(w As DesktopWindow)
  txaDTWatcherEvents.Text &= SetTime() & w.Name & ": VDesktop-Icon geändert." & gb.NewLine
End ' MyDTWatcher_WindowGeometry(w As DesktopWindow)

Public Sub MyDTWatcher_WindowName(w As DesktopWindow)
  txaDTWatcherEvents.Text &= SetTime() & w.Name & ": VDesktop-Name geändert." & gb.NewLine
End ' MyDTWatcher_WindowName(w As DesktopWindow)

Public Sub MyDTWatcher_Change()
  txaDTWatcherEvents.Text &= SetTime() & "Aktueller VDesktop hat sich geändert." & gb.NewLine
End ' MyDTWatcher_Change()

Public Sub MyDTWatcher_Count()
  txaDTWatcherEvents.Text &= SetTime() & "Die VDesktop-Anzahl hat sich geändert." & gb.NewLine
End ' MyDTWatcher_Count()

Public Sub MyDTWatcher_Geometry()
  txaDTWatcherEvents.Text &= SetTime() & "Die VDesktop-Geometrie hat sich geändert." & gb.NewLine
End ' MyDTWatcher_Geometry()
```