

13.4 Kontext-Menü

Eine besondere Form eines Menüs ist das *Kontextmenü* als PopUp-Menü. Der Kontext ist ein sichtbare Komponente. Das bedeutet, dass das PopUp-Menü sich nur auf diese Komponente bezieht. Ein Menü wird erst durch eine spezielle Zuweisung zu einem Kontextmenü. Wenn Sie den Menü-Editor verwenden, dann können Sie eine Liste von unterschiedlichen Menüs definieren und diese im Quelltext einzelnen Komponenten – hier ComboBox und PictureBox – als Kontextmenü zuweisen:

```
Public Sub ComboBox1_Menu()
    mnuComboBox1.Popup
End ' ComboBox1_Menu()

Public Sub PictureBox1_Menu()
    mnuPictureBox1.Popup
End ' PictureBox1_Menu()
```

Diese 2 Anweisungen sparen Sie ein, wenn Sie die Zuweisung bereits im Objektinspektor bei der Eigenschaft *'PopupMenu'* aus der dort angebotenen Liste definierter Menüs vornehmen.

Das Kontextmenü öffnet sich nach einem Klick mit der rechten Maustaste über der Komponente an der Position des Mauszeigers. Ein Kontextmenü enthält eine Liste von Menüs, die nach einem Klick mit der linken Maustaste einen Unter-Programmaufruf auslösen. Sie können das Kontextmenü zur Entwurfszeit im Menü-Editor anlegen und konfigurieren oder es im Quelltext codieren. Folgende Komponenten können Sie mit einem Kontextmenü ausrüsten:

```
Button, ButtonBox, CheckBox, ColorButton, ColorChooser, ColumnView, ComboBox, Container, Control, DateBox, DateChooser, Dial, DirChooser, DirView, DrawingArea, Editor, Embedder, Expander, FileChooser, FileView, FontChooser, Form, Frame, GridView, HBox, HPanel, HSplit, IconPanel, IconView, LCDNumber, Label, ListBox, ListContainer, ListView, MaskBox, MenuButton, MovieBox, Panel, PictureBox, ProgressBar, RadioButton, ScrollArea, ScrollBar, ScrollView, Separator, SidePanel, Slider, SpinBox, TabPanel, TabStrip, TableView, TextArea, TextBox, TextEdit, TextLabel, ToggleButton, ToolButton, ToolPanel, TrayIcon, TreeView, UserControl, UserControl, VBox, VPanel, VSplit, ValueBox, Window, Wizard, _IconPanelContainer, _TabPanelContainer, _WizardContainer.
```

Gut zu wissen: Wenn Sie zum Beispiel über das Ereignis *TextArea_Menu()* und *mnuContextTextArea.Popup* ein Kontextmenü für eine TextArea deklarieren, dann wird das Original überschrieben, wie die beiden folgenden Bilder zeigen:

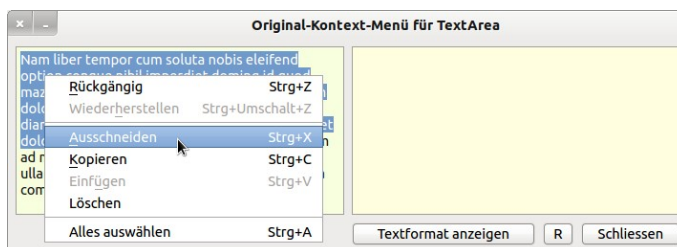


Abbildung 13.4.1: Kontextmenü TextArea – Original

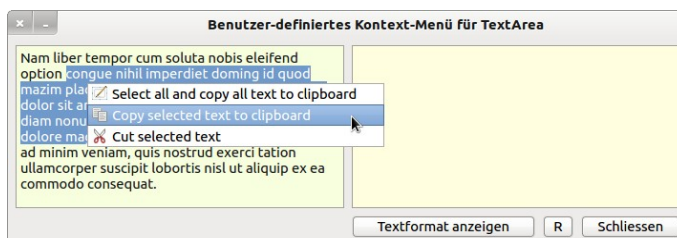


Abbildung 13.4.2: Kontextmenü TextArea – benutzerdefiniert

Sie können Kontextmenüs von ausgewählten Komponenten nicht nur mit eigenen Menüs überschreiben, sondern das Kontextmenü auch generell abschalten. Dazu wird folgender Code eingesetzt – hier für die (fiktive) Komponente TextArea3:

```
Public Sub TextArea3_Menu()
    Stop Event
End
```

13.4.1 Projekt

Im vorgestellten Projekt wird das originale Kontextmenü der zwei Komponenten 'TextArea' jeweils durch ein benutzerdefiniertes, einfaches Kontextmenü ersetzt. Gleichzeitig wird der Einsatz der Klasse *Clipboard* demonstriert, indem Text aus der TextArea1 kopiert oder ausgeschnitten wird, um dann in die TextArea2 eingefügt zu werden. Außerdem können Sie sich nach einem Kopiervorgang den Typ (Textformat) des eingefügten Textes ansehen. Ein heftiger Klick auf den Reset-Knopf (R) stellt den Startzustand des Programms wieder her.

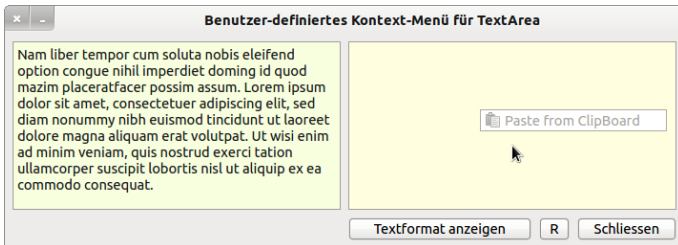


Abbildung 13.4.1.1: Das ClipBoard ist leer

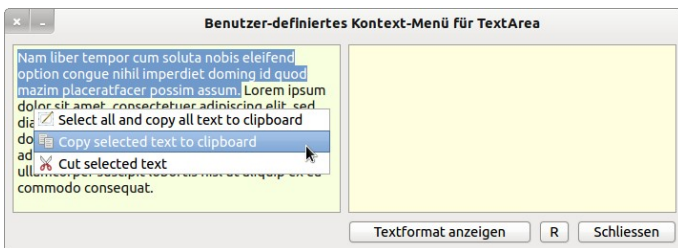


Abbildung 13.4.1.2: Markierter Text wird in das ClipBoard kopiert – Kontextmenü1

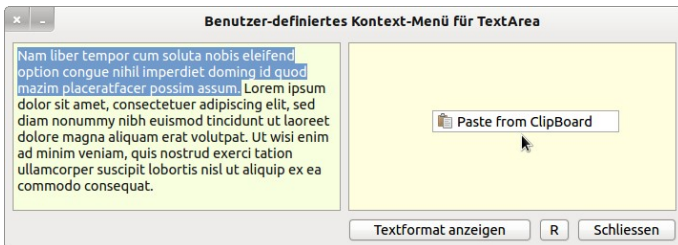


Abbildung 13.4.1.3: Inhalt des ClipBoards kann in die TextArea2 eingefügt werden – Kontextmenü2

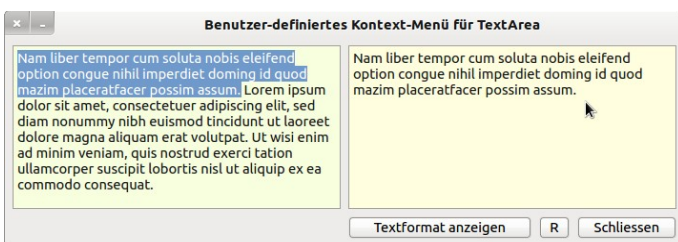


Abbildung 13.4.1.4: Inhalt des ClipBoards wurde in die TextArea2 eingefügt

13.4.2 Projekt-Quelltext

Der Quelltext wird vollständig angegeben und nur in ausgewählten Teilen kommentiert, weil die Kommentare im Quelltext bereits Wesentliches zeigen:

```
[1] ' Gambas class file
[2]
[3] Private sText As String
[4]
```

```

[5] Public Sub Form_Open()
[6]
[7]     FMain.Center
[8]     FMain.Resizable = False
[9]     sText = "Nam liber tempor cum soluta nobis eleifend option congue nihil imperdiet doming "
[10]    sText &= "id quod mazim placeratfacer possim assum. Lorem ipsum dolor sit amet, consectetur "
[11]    sText &= "adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna "
[12]    sText &= "aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation "
[13]    sText &= "ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat."
[14]    TextAreal.Text = sText
[15]    TextAreal.Wrap = True
[16]    TextArea2.Wrap = True
[17]
[18] End ' Form_Open
[19]
[20] Public Sub TextAreal_Menu()
[21]
[22]     Dim mnuContextTextAreal As Menu
[23]     Dim mnuMenuItem As Menu
[24]
[25] ' Es wird ein neues Menü-Objekt für die Textareal erzeugt
[26] mnuContextTextAreal = New Menu(FMain, False)
[27] ' 1. Menü im Menü der Komponente TextAreal
[28] mnuMenuItem = New Menu(mnuContextTextAreal) As "mnuSelectAll"
[29] mnuMenuItem.Text = "Select all and copy all text to clipboard"
[30] mnuMenuItem.Picture = Stock["select"]
[31] ' 2. Menü im Menü
[32] mnuMenuItem = New Menu(mnuContextTextAreal) As "mnuSelectedCopy"
[33] mnuMenuItem.Text = "Copy selected text to clipboard"
[34] mnuMenuItem.Picture = Stock["copy"]
[35] ' 3. Menü im Menü
[36] mnuMenuItem = New Menu(mnuContextTextAreal) As "mnuSelectedCut"
[37] mnuMenuItem.Text = "Cut selected text"
[38] mnuMenuItem.Picture = Stock["cut"]
[39]
[40] mnuContextTextAreal.Popup ' Das Menü wird der TextArea_1 als PopUp-Menü zugewiesen
[41]
[42] ' Ein Nachteil dieser Vorgehensweise, jedem Menüeintrag den Bezeichner mnuMenuItem zu geben, soll
[43] ' nicht verschwiegen werden: Es ist im Programm nicht möglich, einzelne Menü-Eigenschaften
[44] ' zur Laufzeit gezielt zu ändern. Beim Einsatz des Menü-Editors entfällt der Nachteil.
[45]
[46] End ' TextAreal_Menu
[47]
[48] ' Hier werden jetzt die 3 Aktionen programmiert, die ausgelöst werden,
[49] ' wenn das entsprechende Menü im Kontextmenü der TextAreal ausgewählt wurde.
[50]
[51] ' Menü 1 - Komponente TextAreal
[52] Public Sub mnuSelectAll_Click()
[53]     TextAreal.SelectAll()
[54]     TextArea2.Clear
[55]     If TextAreal.Selection.Text <> Null Then
[56]         Clipboard.Copy(TextAreal.Text)
[57]         ' Alternative: TextAreal.Copy()
[58]     Endif ' TextAreal.Selection.Text <> Null
[59] End ' mnuSelectAll_Click
[60]
[61] ' Menü 2 - Komponente TextAreal
[62] Public Sub mnuSelectedCopy_Click()
[63]     If TextAreal.Selection.Text <> Null Then
[64]         TextArea2.Clear
[65]         Clipboard.Copy(TextAreal.Selection.Text)
[66]         ' Alternative: TextAreal.Copy()
[67]     Endif ' TextAreal.Selection.Text <> Null
[68] End ' mnuSelectedCopy_Click
[69]
[70] ' Menü 3 - Komponente TextAreal
[71] Public Sub mnuSelectedCut_Click()
[72]     If TextAreal.Selection.Text <> Null Then
[73]         TextArea2.Clear
[74]         Clipboard.Copy(TextAreal.Selection.Text)
[75]         TextAreal.Cut()
[76]     Endif ' TextAreal.Selection.Text <> Null
[77] End ' mnuSelectedCopy_Click
[78]
[79] Public Sub TextArea2_Menu()
[80]     Dim mnuContextTextArea2 As Menu
[81]     Dim mnuP2T As Menu
[82]
[83] ' Es wird ein neues Menü-Objekt für die Textarea2 erzeugt
[84] mnuContextTextArea2 = New Menu(FMain, False)
[85] ' Ein Menü im Menü2 der Komponente TextArea2 anlegen
[86] mnuP2T = New Menu(mnuContextTextArea2) As "mnuPasteToTextArea2"
[87] mnuP2T.Text = "Paste from ClipBoard"
[88] mnuP2T.Picture = Stock["paste"]

```

```

[89]
[90] If Clipboard.Type = Clipboard.Text Then
[91]     mnuContextTextArea2.Popupup ' Das Menü2 wird der TextArea_2 als PopUp-Menü zugewiesen (aktiv)
[92] Else
[93]     mnuP2T.Enabled = False
[94]     mnuContextTextArea2.Popupup ' Das Menü2 wird der TextArea_2 als PopUp-Menü zugewiesen (nicht aktiv)
[95] Endif ' Clipboard.Type = Clipboard.Text
[96]
[97] End ' TextArea2_Menu
[98]
[99] Public Sub mnuPasteToTextArea2_Click()
[100] ' Hier wird die eine Aktion programmiert, die ausgelöst wird,
[101] ' wenn das eine Menü im Kontextmenü der TextArea2 ausgewählt wurde.
[102]     TextArea2.Insert(Clipboard.Paste("text/plain"))
[103] End ' mnuPasteToTextArea2_Click
[104]
[105] Public Sub btnFormats_Click()
[106]     Dim sElement As String
[107]
[108]     TextArea2.Clear
[109]     For Each sElement In Clipboard.Formats
[110]         TextArea2.Insert(sElement & gb.NewLine)
[111]     Next ' sElement
[112]
[113] End ' btnFormats_Click
[114]
[115] Public Sub btnReset_Click()
[116]     Clipboard.Clear
[117]     TextArea2.Clear
[118]     TextArea1.Text = sText
[119]     TextArea1.SetFocus
[120] End ' Reset
[121]
[122] Public Sub btnClose_Click()
[123]     If Not Clipboard.Type = Clipboard.None Then Clipboard.Clear
[124]     FMain.Close
[125] End ' btnClose

```

Kommentar:

- In den Zeilen 20 bis 46 wird ein Menü1 (Zeile20) mit 3 Menüs angelegt und in der Zeile 40 der Komponente TextArea1 als Kontextmenü zugewiesen.
- In den Zeilen 79 bis 97 erfolgt Analoges – hier jedoch nur mit einem Menü – für die Komponente TextArea2. In Abhängigkeit vom Inhalt des ClipBoards wird das Menü aktiv oder nicht aktiv gesetzt.
- Der in der TextArea1 selektierte, kopierte Text oder ausgeschnittene Text wird in die TextArea2 im vorgegebenen Textformat 'text/plain' eingefügt – Zeile 102.