

RapaGUI 2.1

Rapid Cross-Platform GUI Development On All Islands

Andreas Falkenhahn

Inhaltsverzeichnis

1	Allgemeine Informationen	1
1.1	Einführung	1
1.2	Lizenzvereinbarung	2
1.3	Anforderungen	3
2	Über RapaGUI	5
2.1	Geschichte	5
2.2	Kompatibilität/API Änderungen	5
2.3	Zukunft	6
2.4	Häufig gestellte Fragen	6
2.5	Danksagungen	7
3	Konzeptioneller Überblick	9
3.1	Anwendungsbaum	9
3.2	GUI-Layout	9
3.3	Ausführen von GUIs	11
3.4	Initialisierung von RapaGUI	11
3.5	Umgang mit Objekten	12
3.6	Ereignisse verarbeiten	13
3.7	Benachrichtigungen der Attribute	14
3.8	Dynamische Objekte	15
3.9	Größenveränderung	17
3.10	Anwendbarkeit	20
3.11	Geräteunabhängige Pixel	20
3.12	High-DPI-Unterstützung (hochauflösende Displays)	21
3.13	Tastaturkürzel	22
3.14	Textformatierungs-codes	24
3.15	Implementierung von Hilfetexten	24
3.16	Kontextmenüs	25
3.17	Internationalisierung	26
3.18	Zeichenkodierung	27
3.19	Verbindung zu Hollywood	27
3.20	Bilder-Cache	32
3.21	Plattformabhängige Fähigkeiten	33
3.22	Kompatibilität mit MUI Royale	33
4	Tutorial	37
4.1	Tutorial	37
5	Beispiele	45
5.1	Beispiele	45

6	Befehlsreferenz	47
6.1	moai.CreateApp	47
6.2	moai.CreateDialog	48
6.3	moai.CreateObject	49
6.4	moai.DoMethod	51
6.5	moai.FreeApp	52
6.6	moai.FreeDialog	52
6.7	moai.FreeImage	53
6.8	moai.FreeObject	54
6.9	moai.Get	54
6.10	moai.HaveObject	55
6.11	moai.Notify	55
6.12	moai.Request	56
6.13	moai.Set	57
6.14	moai.UpdateImage	58
7	Accelerator-Klasse (Tastaturkürzel)	61
7.1	Übersicht	61
8	Acceleratoritem-Klasse (Tastaturkürzelement)	63
8.1	Übersicht	63
8.2	Acceleratoritem.Mod	64
8.3	Acceleratoritem.Pressed	64
9	Application-Klasse (Anwendung)	65
9.1	Übersicht	65
9.2	Application.AboutMUI	65
9.3	Application.AboutRapaGUI	65
9.4	Application.AddWindow	65
9.5	Application.ContextMenu	66
9.6	Application.HelpFile	67
9.7	Application.Icon	67
9.8	Application.OpenConfigWindow	67
9.9	Application.RemoveWindow	68
9.10	Application.ShowHelp	68
9.11	Application.ShowHelpNode	69
9.12	Application.Sleep	69
9.13	Application.UseIcons	70
9.14	Application.WindowMenu	70

10	Area-Klasse (Bereich)	71
10.1	Übersicht	71
10.2	Area.ContextMenu	71
10.3	Area.Disabled	72
10.4	Area.FixHeight	73
10.5	Area.FixWidth	73
10.6	Area.FontName	73
10.7	Area.FontSize	74
10.8	Area.FontStyle	74
10.9	Area.Height	75
10.10	Area.Hide	75
10.11	Area.Left	75
10.12	Area.NoAutoKey	76
10.13	Area.Redraw	76
10.14	Area.Tooltip	76
10.15	Area.Top	77
10.16	Area.Weight	77
10.17	Area.Width	78
11	Busybar-Klasse (Beschäftigung)	79
11.1	Übersicht	79
11.2	Busybar.Move	79
11.3	Busybar.Reset	79
12	Button-Klasse (Schaltflächen)	81
12.1	Übersicht	81
12.2	Button.Hint	81
12.3	Button.Icon	81
12.4	Button.IconPos	82
12.5	Button.IconScale	82
12.6	Button.IconType	83
12.7	Button.Pressed	84
12.8	Button.Selected	84
12.9	Button.Text	84
12.10	Button.Toggle	85
13	Checkbox-Klasse (Auswahlkästchen)	87
13.1	Übersicht	87
13.2	Checkbox.Right	87
13.3	Checkbox.Selected	87

14	Choice-Klasse (Auswahl)	89
14.1	Übersicht	89
14.2	Choice.Active	89
14.3	Choice.Clear	89
14.4	Choice.Count	90
14.5	Choice.GetEntry	90
14.6	Choice.Insert	91
14.7	Choice.Remove	91
14.8	Choice.Rename	92
15	Combobox-Klasse (Textlisten)	93
15.1	Übersicht	93
15.2	Combobox.Acknowledge	93
15.3	Combobox.Clear	93
15.4	Combobox.Close	94
15.5	Combobox.Count	94
15.6	Combobox.GetEntry	94
15.7	Combobox.Insert	95
15.8	Combobox.Open	95
15.9	Combobox.Popup	96
15.10	Combobox.Remove	96
15.11	Combobox.Rename	96
15.12	Combobox.Selected	97
15.13	Combobox.Value	97
16	Dialog-Klasse	99
16.1	Übersicht	99
16.2	Dialog.EndModal	100
16.3	Dialog.ShowModal	101
17	Finddialog-Klasse (Systemsuchdialog)	103
17.1	Übersicht	103
17.2	Finddialog.Down	103
17.3	Finddialog.Find	103
17.4	Finddialog.FindNext	104
17.5	Finddialog.FindString	104
17.6	Finddialog.MatchCase	104
17.7	Finddialog.NoMatchCase	105
17.8	Finddialog.NoUpDown	105
17.9	Finddialog.NoWholeWord	105
17.10	Finddialog.Replace	106
17.11	Finddialog.ReplaceAll	106
17.12	Finddialog.ReplaceMode	106
17.13	Finddialog.ReplaceString	107
17.14	Finddialog.WholeWord	107

18	Group-Klasse (Gruppen)	109
18.1	Übersicht	109
18.2	Group.Append	110
18.3	Group.Color	110
18.4	Group.Columns	111
18.5	Group.ExitChange	111
18.6	Group.Frame	112
18.7	Group.FrameTitle	112
18.8	Group.HAlign	112
18.9	Group.Hide	113
18.10	Group.HorizSpacing	113
18.11	Group.Icon	114
18.12	Group.IconScale	114
18.13	Group.IconType	115
18.14	Group.InitChange	115
18.15	Group.Insert	116
18.16	Group.Padding	117
18.17	Group.Paint	117
18.18	Group.Prepend	118
18.19	Group.Remove	119
18.20	Group.SameSize	120
18.21	Group.Spacing	120
18.22	Group.Title	120
18.23	Group.VAlign	121
18.24	Group.VertSpacing	121
18.25	Group.Weight	121
19	HLine-Klasse (HorizontalLinie)	123
19.1	Übersicht	123
20	Hollywood-Klasse	125
20.1	Übersicht	125
20.2	Hollywood.Display	125
20.3	Hollywood.DropFile	126
20.4	Hollywood.DropTarget	126
21	HSpace-Klasse (HorizontalAbstand)	127
21.1	Übersicht	127
21.2	HSpace.Width	127

22	HSplitter-Klasse	129
22.1	Übersicht	129
22.2	HSplitter.Border	129
22.3	HSplitter.Gravity	129
22.4	HSplitter.MinPaneSize	130
22.5	HSplitter.Position	130
22.6	HSplitter.Split	130
22.7	HSplitter.Unsplit	131
23	HTMLview-Klasse (HTMLAnsicht)	133
23.1	Übersicht	133
23.2	HTMLview.CanGoBack	133
23.3	HTMLview.CanGoForward	133
23.4	HTMLview.ClearHistory	134
23.5	HTMLview.Contents	134
23.6	HTMLview.File	134
23.7	HTMLview.GoBack	135
23.8	HTMLview.GoForward	135
23.9	HTMLview.Reload	135
23.10	HTMLview.Search	135
23.11	HTMLview.Title	136
23.12	HTMLview.URL	136
24	Hyperlink-Klasse	139
24.1	Übersicht	139
24.2	Hyperlink.Label	139
24.3	Hyperlink.URL	139
25	Image-Klasse (Bild)	141
25.1	Übersicht	141
25.2	Image.Brush	141
25.3	Image.BrushScale	141
25.4	Image.BrushType	142
26	Label-Klasse (Beschriftung)	143
26.1	Übersicht	143
26.2	Label.Align	143
26.3	Label.Text	144

27	Listview-Klasse (Listenansicht)	145
27.1	Übersicht	145
27.2	Listview.AbortEditing	146
27.3	Listview.Active	146
27.4	Listview.Alternate	147
27.5	Listview.Clear	148
27.6	Listview.ClickColumn	148
27.7	Listview.Columns	148
27.8	Listview.CompareItems	148
27.9	Listview.DefClickColumn	149
27.10	Listview.DoubleClick	149
27.11	Listview.DropFile	150
27.12	Listview.DropTarget	150
27.13	Listview.Edit	151
27.14	Listview.Entries	151
27.15	Listview.Exchange	152
27.16	Listview.First	152
27.17	Listview.ForceMode	153
27.18	Listview.GetColumnID	153
27.19	Listview.GetDisabled	154
27.20	Listview.GetEntry	154
27.21	Listview.GetSelection	155
27.22	Listview.GetState	155
27.23	Listview.HRrules	156
27.24	Listview.Insert	156
27.25	Listview.InsertColumn	157
27.26	Listview.ItemStyle	159
27.27	Listview.Jump	159
27.28	Listview.LongClick	160
27.29	Listview.Move	160
27.30	Listview.MultiSelect	161
27.31	Listview.Quiet	161
27.32	Listview.Remove	161
27.33	Listview.RemoveColumn	162
27.34	Listview.Rename	162
27.35	Listview.RowHeight	163
27.36	Listview.Select	164
27.37	Listview.SetDisabled	165
27.38	Listview.SetState	165
27.39	Listview.Sort	165
27.40	Listview.SortColumn	166
27.41	Listview.StartEditing	166
27.42	Listview.SystemTheme	167
27.43	Listview.TitleClick	167
27.44	Listview.ValueChange	168
27.45	Listview.Visible	168
27.46	Listview.VRules	168

28	Listviewcolumn-Klasse	
	(Listenansichtsspalten)	171
28.1	Übersicht	171
28.2	Listviewcolumn.Align	171
28.3	Listviewcolumn.Checkbox	171
28.4	Listviewcolumn.Editable	172
28.5	Listviewcolumn.Hide	173
28.6	Listviewcolumn.Icon	173
28.7	Listviewcolumn.IconScale	174
28.8	Listviewcolumn.IconType	174
28.9	Listviewcolumn.Sortable	175
28.10	Listviewcolumn.SortOrder	175
28.11	Listviewcolumn.Title	176
28.12	Listviewcolumn.Width	176
29	Listviewitem-Klasse	
	(Listenansichts-Element)	177
29.1	Übersicht	177
29.2	Listviewitem.Icon	177
30	Menu-Klasse (Menü)	179
30.1	Übersicht	179
30.2	Menu.Append	179
30.3	Menu.Disabled	180
30.4	Menu.Insert	180
30.5	Menu.NoAutoKey	181
30.6	Menu.Prepend	181
30.7	Menu.Remove	181
30.8	Menu.Title	182
30.9	Menu.Type	182
31	Menubar-Klasse (Menüleiste)	185
31.1	Übersicht	185
31.2	Menubar.Append	186
31.3	Menubar.Insert	186
31.4	Menubar.Prepend	187
31.5	Menubar.Remove	187
32	MenuItem-Klasse (Menüelement)	189
32.1	Übersicht	189
32.2	MenuItem.Disabled	189
32.3	MenuItem.Help	189
32.4	MenuItem.NoAutoKey	190
32.5	MenuItem.Selected	190
32.6	MenuItem.Shortcut	191

32.7	MenuItem.Title	193
32.8	MenuItem.Type	193
33	MOAI-Klasse	195
33.1	Übersicht	195
33.2	MOAI.Class	195
33.3	MOAI.I18N	195
33.4	MOAI.ID	195
33.5	MOAI.NoNotify	196
33.6	MOAI.Notify	196
33.7	MOAI.NotifyData	197
33.8	MOAI.UserData	197
34	Pageview-Klasse (Seitenansicht)	199
34.1	Übersicht	199
34.2	Pageview.Active	199
34.3	Pageview.Append	200
34.4	Pageview.GetPageID	201
34.5	Pageview.Insert	201
34.6	Pageview.Mode	202
34.7	Pageview.Multiline	202
34.8	Pageview.Pages	203
34.9	Pageview.PlainBG	203
34.10	Pageview.Position	203
34.11	Pageview.Prepend	204
34.12	Pageview.Remove	205
35	Popcolor-Klasse (Farbdialog)	207
35.1	Übersicht	207
35.2	Popcolor.RGB	207
35.3	Popcolor.Title	207
36	Popfile-Klasse (Dateialog)	209
36.1	Übersicht	209
36.2	Popfile.File	209
36.3	Popfile.Pattern	209
36.4	Popfile.SaveMode	210
36.5	Popfile.Title	210
37	Popfont-Klasse (Schriftdialog)	211
37.1	Übersicht	211
37.2	Popfont.Font	211
37.3	Popfont.MaxSize	211
37.4	Popfont.MinSize	212
37.5	Popfont.Title	212

38	Poppath-Klasse (Verzeichnisdialog)	213
38.1	Übersicht	213
38.2	Poppath.Path	213
38.3	Poppath.Title	213
39	Progressbar-Klasse (Fortschrittsbalken) ...	215
39.1	Übersicht	215
39.2	Progressbar.Horiz	215
39.3	Progressbar.Level	215
39.4	Progressbar.Max	216
40	Radio-Klasse	217
40.1	Übersicht	217
40.2	Radio.Active	217
40.3	Radio.Columns	217
40.4	Radio.GetItem	218
40.5	Radio.SetItem	218
40.6	Radio.Title	219
41	Rectangle-Klasse (Rechteck)	221
41.1	Übersicht	221
42	Scrollbar-Klasse (Bildlaufleiste)	223
42.1	Übersicht	223
42.2	Scrollbar.AutoScale	223
42.3	Scrollbar.Horiz	223
42.4	Scrollbar.Level	224
42.5	Scrollbar.Range	224
42.6	Scrollbar.StepSize	224
42.7	Scrollbar.Target	225
42.8	Scrollbar.UseWinBorder	225
42.9	Scrollbar.Visible	226
43	Scrollcanvas-Klasse (Bildlaufleinwand)	227
43.1	Übersicht	227
43.2	Scrollcanvas.AutoBars	227
43.3	Scrollcanvas.AutoScale	228
43.4	Scrollcanvas.Paint	228
43.5	Scrollcanvas.Scroll	230
43.6	Scrollcanvas.StepSize	230
43.7	Scrollcanvas.UseLeftBorder	230
43.8	Scrollcanvas.UseWinBorder	231
43.9	Scrollcanvas.VirtHeight	231
43.10	Scrollcanvas.VirtWidth	232

44	Scrollgroup-Klasse (Bildlaufgruppe)	233
44.1	Übersicht	233
44.2	Scrollgroup.AutoBars	233
44.3	Scrollgroup.Horiz	233
44.4	Scrollgroup.UseWinBorder	234
45	Slider-Klasse (Schieberegler)	235
45.1	Übersicht	235
45.2	Slider.Drag	235
45.3	Slider.Horiz	235
45.4	Slider.Level	236
45.5	Slider.Max	236
45.6	Slider.Min	236
45.7	Slider.Quiet	237
45.8	Slider.Release	237
45.9	Slider.Reverse	237
45.10	Slider.StepSize	238
46	Statusbar-Klasse (Statusleiste)	239
46.1	Übersicht	239
47	Statusbaritem-Klasse (Statusleistenelement) ..	241
47.1	Übersicht	241
47.2	Statusbaritem.Text	241
47.3	Statusbaritem.Width	241
48	Text-Klasse	243
48.1	Übersicht	243
48.2	Text.Align	243
48.3	Text.Frame	243
48.4	Text.Text	244
49	Texteditor-Klasse	245
49.1	Übersicht	245
49.2	Texteditor.Align	245
49.3	Texteditor.AreaMarked	245
49.4	Texteditor.Bold	246
49.5	Texteditor.Clear	246
49.6	Texteditor.Color	247
49.7	Texteditor.Copy	247
49.8	Texteditor.CursorPos	247
49.9	Texteditor.Cut	248
49.10	Texteditor.GetLineLength	248
49.11	Texteditor.GetPosition	248
49.12	Texteditor.GetSelection	249

49.13	Texteditor.GetText	249
49.14	Texteditor.GetXY	250
49.15	Texteditor.HasChanged	250
49.16	Texteditor.Hint	250
49.17	Texteditor.Insert	251
49.18	Texteditor.Italic	251
49.19	Texteditor.Mark	252
49.20	Texteditor.MarkAll	252
49.21	Texteditor.MarkNone	252
49.22	Texteditor.NoWrap	253
49.23	Texteditor.Paste	253
49.24	Texteditor.ReadOnly	253
49.25	Texteditor.Redo	254
49.26	Texteditor.RedoAvailable	254
49.27	Texteditor.ScrollToLine	254
49.28	Texteditor.SetBold	255
49.29	Texteditor.SetColor	255
49.30	Texteditor.SetItalic	256
49.31	Texteditor.SetUnderline	256
49.32	Texteditor.Styled	257
49.33	Texteditor.Text	257
49.34	Texteditor.Underline	257
49.35	Texteditor.Undo	258
49.36	Texteditor.UndoAvailable	258
50	Textentry-Klasse (Texteingabe)	259
50.1	Übersicht	259
50.2	Textentry.Accept	259
50.3	Textentry.Acknowledge	259
50.4	Textentry.AdvanceOnCR	260
50.5	Textentry.Copy	260
50.6	Textentry.CursorPos	260
50.7	Textentry.Cut	261
50.8	Textentry.GetSelection	261
50.9	Textentry.Hint	261
50.10	Textentry.Insert	262
50.11	Textentry.Mark	262
50.12	Textentry.MarkAll	263
50.13	Textentry.MarkNone	263
50.14	Textentry.MaxLen	263
50.15	Textentry.Password	264
50.16	Textentry.Paste	264
50.17	Textentry.ReadOnly	264
50.18	Textentry.Redo	265
50.19	Textentry.Reject	265
50.20	Textentry.Text	265
50.21	Textentry.Undo	266

51	TextView-Klasse (Textanzeige)	267
51.1	Übersicht	267
51.2	TextView.Align	267
51.3	TextView.Append	267
51.4	TextView.Styled	268
51.5	TextView.Text	268
52	Toolbar-Klasse (Symbolleiste)	269
52.1	Übersicht	269
52.2	Toolbar.Horiz	269
52.3	Toolbar.ViewMode	270
53	Toolbarbutton-Klasse (Symbolleistenschaltfläche)	271
53.1	Übersicht	271
53.2	Toolbarbutton.Disabled	271
53.3	Toolbarbutton.Help	271
53.4	Toolbarbutton.Icon	272
53.5	Toolbarbutton.IconScale	272
53.6	Toolbarbutton.IconType	273
53.7	Toolbarbutton.Pressed	273
53.8	Toolbarbutton.Selected	274
53.9	Toolbarbutton.Tooltip	274
53.10	Toolbarbutton.Type	274
54	Treeview-Klasse (Baumansicht)	277
54.1	Übersicht	277
54.2	Treeview.AbortEditing	279
54.3	Treeview.Active	279
54.4	Treeview.Alternate	280
54.5	Treeview.Clear	280
54.6	Treeview.ClickColumn	280
54.7	Treeview.Close	281
54.8	Treeview.DoubleClick	281
54.9	Treeview.DropFile	281
54.10	Treeview.DropTarget	282
54.11	Treeview.EditableNodes	282
54.12	Treeview.ForceMode	283
54.13	Treeview.GetEntry	284
54.14	Treeview.HRules	285
54.15	Treeview.InsertLeaf	286
54.16	Treeview.InsertNode	287
54.17	Treeview.Open	288
54.18	Treeview.Remove	289
54.19	Treeview.StartEditing	289
54.20	Treeview.ValueChange	290
54.21	Treeview.VRules	291

55	Treeviewcolumn-Klasse	
	(Baumansichtspalten)	293
55.1	Übersicht	293
55.2	Treeviewcolumn.Align	293
55.3	Treeviewcolumn.Checkbox	293
55.4	Treeviewcolumn.Editable	294
55.5	Treeviewcolumn.Hide	295
55.6	Treeviewcolumn.Icon	295
55.7	Treeviewcolumn.IconScale	296
55.8	Treeviewcolumn.IconType	296
55.9	Treeviewcolumn.Title	297
55.10	Treeviewcolumn.Width	297
56	Treeviewleaf-Klasse (Baumansichtblätter)	299
56.1	Übersicht	299
56.2	Treeviewleaf.Edit	299
56.3	Treeviewleaf.GetDisabled	300
56.4	Treeviewleaf.GetIcon	300
56.5	Treeviewleaf.GetItem	300
56.6	Treeviewleaf.GetState	301
56.7	Treeviewleaf.SetDisabled	301
56.8	Treeviewleaf.SetIcon	302
56.9	Treeviewleaf.SetItem	302
56.10	Treeviewleaf.SetState	302
56.11	Treeviewleaf.UID	303
57	Treeviewleafitem-Klasse	
	(Baumansichtblätterelement)	305
57.1	Übersicht	305
57.2	Treeviewleafitem.Icon	305
58	Treeviewnode-Klasse (Baumansichtknoten)	307
58.1	Übersicht	307
58.2	Treeviewnode.Edit	307
58.3	Treeviewnode.Icon	308
58.4	Treeviewnode.Name	308
58.5	Treeviewnode.UID	308
59	VLine-Klasse (VertikalLinie)	311
59.1	Übersicht	311
60	VSpace-Klasse (VertikalAbstand)	313
60.1	Übersicht	313
60.2	VSpace.Height	313

61	VSplitter-Klasse	315
61.1	Übersicht	315
61.2	VSplitter.Border	315
61.3	VSplitter.Gravity	315
61.4	VSplitter.MinPaneSize	316
61.5	VSplitter.Position	316
61.6	VSplitter.Split	316
61.7	VSplitter.Unsplit	317
62	Window-Klasse (Fenster)	319
62.1	Übersicht	319
62.2	Window.Accelerator	319
62.3	Window.Activate	320
62.4	Window.ActiveObject	320
62.5	Window.Borderless	320
62.6	Window.CloseGadget	321
62.7	Window.CloseRequest	321
62.8	Window.DefaultObject	321
62.9	Window.DragBar	322
62.10	Window.Height	322
62.11	Window.HideFromTaskbar	322
62.12	Window.Left	323
62.13	Window.Margin	323
62.14	Window.MaximizeGadget	324
62.15	Window.Menubar	324
62.16	Window.MinimizeGadget	324
62.17	Window.NoCyclerMenu	325
62.18	Window.Open	325
62.19	Window.Orientation	325
62.20	Window.Parent	326
62.21	Window.PubScreen	326
62.22	Window.Remember	327
62.23	Window.ScreenTitle	327
62.24	Window.SingleMenu	328
62.25	Window.SizeChange	328
62.26	Window.SizeGadget	328
62.27	Window.StayOnTop	329
62.28	Window.Subtitle	329
62.29	Window.Title	329
62.30	Window.Toolwindow	330
62.31	Window.Top	330
62.32	Window.UseBottomBorderScroller	331
62.33	Window.UseLeftBorderScroller	331
62.34	Window.UseRightBorderScroller	331
62.35	Window.Width	332

Anhang A	Lizenzen	333
A.1	wxWidgets license	333
A.2	MUI license	333
A.3	Expat license	334
A.4	LGPL license	334
Index		343

1 Allgemeine Informationen

1.1 Einführung

RapaGUI ist ein Plugin für Hollywood, mit dem Sie ganz einfach GUIs mit Hollywood erstellen können. Sie müssen lediglich die Gestaltung der GUI in einer XML-Datei definieren, die von RapaGUI automatisch in eine vollwertige GUI umgewandelt wird. Einfacher geht es nicht!

RapaGUI's beste Eigenschaft ist definitiv das plattformübergreifende Design. RapaGUI wurde wie Hollywood als plattformunabhängiges Programm konzipiert. Daher läuft es auf einer Vielzahl von Plattformen, aber dennoch verwendet es OS-native Widgets (das ist ein Kofferwort aus "window" und "gadget" und ist ein sichtbares Bedienelement) auf allen unterstützten Plattformen, um Ihren Programmen ein authentisches Aussehen zu verleihen. Derzeit ist RapaGUI in nativen Versionen für Windows, Linux (GTK2/GTK3), macOS, AmigaOS, MorphOS, AROS und sogar für Android. Mit RapaGUI müssen Sie Ihre Programme nur einmal schreiben und sie laufen automatisch auch auf vielen anderen Plattformen! Dies ermöglicht es Ihnen, Ihre Zeit auf die wirklich wichtigen Dinge, d.h. Programmabläufe zu verwenden, statt Anpassungen für viele verschiedene Plattformen schreiben zu müssen.

RapaGUI verwendet ein objektorientiertes Design, das aus über 40 MOAI-Klassen (Magic Omnigui Architecture Interface) besteht. Diese MOAI-Klassen bilden das Herzstück von RapaGUI. Alle von RapaGUI unterstützten GUI-Elemente (Fenster, Widgets, Menüleisten...) sind einfach Objekte, die von diesen MOAI-Klassen abgeleitet sind. Indem diese Klassen die verschiedenen nativen OS-GUI-APIs in plattformunabhängige MOAI-Klassen umwandeln, reduzieren sie die vielen Designs der verschiedenen OS-GUI-APIs in nur ein einziges MOAI API-Design, welches von RapaGUI in Stein gemeißelt wurde!

RapaGUI unterstützt alle Widgets, die Sie zur Erstellung moderner GUI-Programmen benötigen, einschließlich mehrspaltige Listenansicht, Baumansicht, Registerkarten, Symbolleiste, Texteditor, Menüleisten, HTML-Ansichten und vieles mehr. Das Highlight von RapaGUI ist jedoch sicherlich die eingebaute Hollywood-MOAI-Klasse. Diese Klasse ermöglicht die dynamische Einbettung kompletter Hollywood-Displays in GUIs, mit denen Hollywoods leistungsstarke Multimedia-Funktionalität mit RapaGUIs GUI-Fähigkeiten in einem einzigen leistungsstarken Programm kombiniert werden kann.

RapaGUI kommt mit einer umfangreichen Dokumentation in verschiedenen Formaten wie PDF, HTML, AmigaGuide und CHM daher, die die Grundlagen der GUI-Programmierung detailliert beschreibt und eine komfortable MOAI-Funktion und Klassenreferenz bietet. Eine Schritt-für-Schritt-Anleitung, die Sie zu Ihrem ersten RapaGUI-Programm führt, ist ebenfalls enthalten. Darüber hinaus sind viele Beispielskripte im RapaGUI-Archiv enthalten, darunter auch fortgeschrittene Skripte wie ein kompletter Videoplayer, die die Leistungsfähigkeit von Hollywood und RapaGUI bei der Zusammenarbeit unter Beweis stellen.

All dies macht RapaGUI zum ultimativen plattformübergreifenden GUI-Werkzeug, das sorgfältig für Sie als Segler der sieben GUI-Seen entwickelt wurde! Nur RapaGUI ermöglicht eine schnelle plattformübergreifende GUI-Entwicklung auf allen Inseln - es ist die ultimative Verschmelzung der verschiedenen OS-GUI-Werkzeugen zu einem MOAI-Werkzeug, das für die Ewigkeit und darüber hinaus in Stein gemeißelt ist.

1.2 Lizenzvereinbarung

RapaGUI ist © Copyright 2015-2021 bei Andreas Falkenhahn (im folgenden "der Autor" genannt). Alle Rechte vorbehalten.

Das Programm wird zur Verfügung gestellt "wie es ist" und der Autor kann für keinerlei Schäden, welcher Natur sie auch immer sein mögen, verantwortlich gemacht werden. Sie benutzen dieses Programm völlig auf eigene Gefahr und eigenes Risiko. Der Autor gibt keinerlei Garantien in Verbindung mit der Benutzung dieses Programmes, nicht einmal die Garantie der Funktionstüchtigkeit.

Dieses Programm kann frei weitergegeben werden solange die folgenden drei Bedingungen erfüllt sind:

1. Es dürfen keine Änderungen am Programm vorgenommen werden.
2. Das Programm darf nicht verkauft werden.
3. Wenn Sie das Programm auf einer Coverdisk veröffentlichen möchten, müssen Sie erst um Erlaubnis fragen.

Dieses Programm benutzt wxWidgets Copyright (C) 1998-2005 Julian Smart, Robert Roebing u.a. Siehe [Abschnitt A.1 \[wxWidgets-Lizenz\]](#), Seite 333, für Details.

Dieses Programm benutzt das Magic User Interface (MUI), welches (C) Copyright 1992-97 by Stefan Stuntz ist. Siehe [Abschnitt A.2 \[MUI-Lizenz\]](#), Seite 333, für Details.

Dieses Programm benutzt Expat (C) Copyright 1998, 1999, 2000 Thai Open Source Software Center Ltd und Clark Cooper. (C) Copyright 2001, 2002, 2003, 2004, 2005, 2006 Expat-Autoren. Siehe [Abschnitt A.3 \[Expat-Lizenz\]](#), Seite 334, für Details.

Dieses Programm benutzt TextEditor.mcc von Allan Odgaard und dem TextEditor.mcc Open Source Team. Siehe [Abschnitt A.4 \[LGPL-Lizenz\]](#), Seite 334, für Details.

Dieses Programm benutzt HTMLview.mcc von Allan Odgaard und dem HTMLview.mcc Open Source Team. Siehe [Abschnitt A.4 \[LGPL-Lizenz\]](#), Seite 334, für Details.

Dieses Programm benutzt TheBar.mcc von Alfonso Ranieri und dem TheBar.mcc Open Source Team. Siehe [Abschnitt A.4 \[LGPL-Lizenz\]](#), Seite 334, für Details.

Dieses Programm benutzt codesets.library von Alfonso Ranieri und dem codesets.library Open Source Team. Siehe [Abschnitt A.4 \[LGPL-Lizenz\]](#), Seite 334, für Details.

Das Handbuch von MUI Royale basiert auf dem MUI 3.8 Software Development Kit © Copyright 1992-97 by Stefan Stuntz. Einige weitergehende Informationen in diesem Handbuch stammen von den TextEditor.mcc und TheBar.mcc Entwicklerunterlagen.

Alle Warenzeichen sind Eigentum ihrer jeweiligen Firmen.

FÜR DIESES PROGRAMM GIBT ES KEINE GARANTIE, SOWEIT ES DIE ANZUWENDENDEN GESETZE ZULASSEN. SOFERN ANDERSWO NICHTS GEGENTEILIGES GESCHRIEBEN STEHT STELLEN DER AUTOR UND/ODER DRITTE DAS PROGRAMM "SO WIE ES IST" ZUR VERFÜGUNG, OHNE IRGEND-EINE GARANTIE, WEDER DIREKT NOCH INDIREKT. DIES BEINHÄLTET, IST ABER NICHT DARAUF BESCHRÄNKT, VERKÄUFLICHKEIT UND EIGNUNG FÜR EINEN BESTIMMTEN VERWENDUNGSZWECK. DAS VOLLSTÄNDIGE RISIKO DER QUALITÄT UND AUSFÜHRBARKEIT DES PROGRAMMS LIEGT

BEIM ANWENDER. SOLLTE SICH DAS PROGRAMM ALS DEFEKT HERAUSSTELLEN, LIEGEN ALLE KOSTEN FÜR SERVICE, INSTANDSETZUNG ODER NACHBESSERUNG BEIM ANWENDER.

KEIN COPYRIGHT-INHABER ODER DRITTER, DER DAS PROGRAMM WIE OBEN ERLAUBT WEITERVERKAUFT, KANN FÜR SCHÄDEN IRGENDWELCHER ART HAFTBAR GEMACHT WERDEN (DIES BEINHÄLTET, IST ABER NICHT BESCHRÄNKT AUF, DATENVERLUST INFOLGE UNFÄHIGKEIT DES PROGRAMMS, MIT ANDEREN PROGRAMMEN ZUSAMMENZUARBEITEN), SELBST WENN EIN SOLCHER INHABER ODER DRITTER AUF DIE MÖGLICHKEIT EINES SOLCHEN SCHADENS HINGEWIESEN WURDE, AUSSER ES BESTEHT EINE SCHRIFTLICHE EINWILLIGUNG ODER WIRD VOM GESETZ VERLANGT.

1.3 Anforderungen

RapaGUI benötigt mindestens Hollywood 9.0. Je nach Plattform ergeben sich folgende zusätzliche Anforderungen:

Windows Version:

- mindestens Windows XP

macOS Version:

- Intel Macs: mindestens macOS 10.10
- PowerPC Macs: mindestens macOS 10.5

Linux version:

- erfordert GTK+ 2 oder GTK+ 3, je nachdem, welche Version Sie von RapaGUI verwenden (für die meisten Architekturen ist RapaGUI in einem Build für GTK+ 2 und GTK+ 3 verfügbar)
- optional: WebKitGTK+ wird für die HTMLview-Klasse benötigt.

AmigaOS version:

- erfordert MUI 3.8 oder besser; die Verwendung von MUI 4 (oder besser) wird jedoch dringend empfohlen!
- 68020+ oder PowerPC-Prozessor
- CyberGraphX oder Picasso96
- codesets.library für die UTF-8-Unterstützung
- Toolbar-Klasse erfordert TheBar.mcc
- extEditor-Klasse erfordert TextEditor.mcc
- HTMLview-Klasse erfordert HTMLview.mcc

2 Über RapaGUI

2.1 Geschichte

Für ein vollständiges Änderungsprotokoll von RapaGUI beachten Sie bitte die auf englisch verfasste Datei `history.txt`.

2.2 Kompatibilität/API Änderungen

RapaGUI 2.0 API-Änderungen

Leider war es notwendig, in RapaGUI 2.0 ein paar Designänderungen vorzunehmen. Dies könnte einige Anpassungen in Ihren Skripten erfordern, um sie mit RapaGUI 2.0 kompatibel zu machen. Bitte lesen Sie sich die folgenden Hinweise durch, um zu erfahren, ob Ihr Skript betroffen ist.

- Ereignisse werden jetzt immer behandelt, wenn sie auftreten. Nehmen wir zum Beispiel an, Sie haben eine Benachrichtigung für das Attribut `Radio.Active` installiert und gehen dann folgendermaßen vor:

```
moai.Set("radio", "active", 5) ; löst Ereignis-Callback aus!
p_DoSomething()
```

Mit RapaGUI 1.x würde die Callback-Funktion für `Radio.Active` nicht vor dem Befehl `WaitEvent()` aufgerufen werden, d.h. `p_DoSomething()` würde immer vor der Callback-Funktion für `Radio.Active` aufgerufen. In RapaGUI 2.0 ist dies jetzt anders. Der Ereignis-Handler wird immer dann aufgerufen, wenn das Ereignis eintritt. Das bedeutet, dass `p_DoSomething()` nach der Callback-Funktion für `Radio.Active` aufgerufen wird, da der Aufruf von `moai.Set()` die Ausführung des Ereignis-Handlers sofort nach dem Ändern des aktiven Radioelements erzwingt.

- `WaitEvent()` wird nie zurückkehren, da auf den meisten Plattformen ein Aufruf von `WaitEvent()` ein einfaches Ticket ist, das die Hauptereignisschleife Ihres Programmes startet und nicht mehr zurückkehrt. Nur auf AmigaOS und kompatiblen Geräten kommt `WaitEvent()` immer noch von Zeit zu Zeit zurück, aber Sie sollten sich nicht auf dieses Verhalten verlassen, da es sich auf anderen Plattformen anders verhält. Wenn Sie möchten, dass `WaitEvent()` benutzerdefinierte Callbacks ausführt, verwenden Sie einfach den neuen Befehl `RunCallback()` von Hollywood 9.0.
- Alle RapaGUI-Projekte müssen jetzt `WaitEvent()` verwenden. Es ist nicht mehr zulässig, eine benutzerdefinierte Ereignisverwaltung mit `CheckEvent()` oder `CheckEvents()` zu implementieren. Das Steuern normaler Hollywood-Skripte über RapaGUI funktioniert nur, wenn auch `WaitEvent()` verwendet wird.
- `Dialog.ShowModal` unterstützt keine Dialogbefehle mehr. Wenn Sie einen Dialog öffnen möchten, der Ihr Skript nicht blockiert, müssen Sie jetzt `Window.Open` für den Dialog auf `True` setzen (und möglicherweise auch `Application.Sleep` auf `True` setzen) und dann können Sie das eigentliche Verhalten des Dialogs durch wiederholtes Aufrufen eines Verwaltungsbefehls implementieren, entweder mit `SetTimeout()` oder dem neuen Befehl `RunCallback()` von Hollywood 9.0. Sehen Sie sich das aktualisierte Dialog-Beispiel (mitgeliefertes Hollywood-Script) an, um eine Referenzimplementierung eines

nicht blockierenden Dialogs zu sehen, der diese Technik verwendet (wählen Sie "Test progress bar dialog" aus dem Menü).

- Da modale Ereignisschleifen von RapaGUI nicht mehr unterstützt werden, deaktiviert das Plugin nun alle Hollywood-Befehle, die versuchen, eine modale Ereignisschleife zu starten, z.B. `WaitLeftMouse()`, `WaitSampleEnd()`, `InKeyStr()` etc. Diese können nicht mehr zusammen mit RapaGUI verwendet werden.
- RapaGUI benötigt jetzt Hollywood 9. RapaGUI 2.0 verwendet viele Schnittstellen, die nur in Hollywood 9 verfügbar sind, sodass Sie Version 2.0 nicht mehr mit älteren Hollywood-Versionen verwenden können.
- Beim Setzen von `Texteditor.Text` löst RapaGUI keine `Texteditor.HasChanged`-Benachrichtigung mehr aus. Stattdessen wird `Texteditor.HasChanged` einfach auf `False` gesetzt, ohne ein Ereignis auszulösen.
- `Toolbar.ViewMode` ist nicht mehr standardmäßig `TextGfx`, sondern auf `Gfx` eingestellt.
- Wenn eine Listenansicht (`Listview`) sortierbare Spalten hat, werden jetzt immer alle Elemente sortiert. In einer Listenansicht mit sortierbaren Spalten ist es nicht mehr möglich, Einträge an beliebigen Positionen einzufügen. Stattdessen werden die Einträge immer sortiert. Die Sortierung wird durch die Spalte bestimmt, die den Sortierfokus hat. Siehe [Abschnitt 27.40 \[Listview.SortColumn\]](#), [Seite 166](#), für Details.

2.3 Zukunft

Hier sind einige Punkte, die auf meiner Aufgabenliste stehen:

- RapaGUI auf iOS portieren
- Unterstützung für Scintilla für eine erweiterte plattformübergreifende Texteditor-Komponente
- Unterstützung für erweiterte Benutzeroberflächen mit wxAUI (Docking etc.)
- Unterstützung für Ziehen und Ablegen
- Unterstützung für mehr Widgets

2.4 Häufig gestellte Fragen

Dieser Abschnitt behandelt einige häufig gestellte Fragen. Bitte lesen Sie diese zuerst, bevor Sie auf der Mailingliste oder im Forum nachfragen, da Ihr Problem möglicherweise hier behandelt wurde.

F: Meine GUI ist nicht größenveränderbar. Was mache ich falsch?

A: Es gibt einige Punkte, die Sie beachten müssen, damit die Fähigkeit zum Ändern der Größe korrekt funktioniert. Wenn es ein Widget in Ihrer GUI gibt, welches eine feste Größe hat, müssen Sie es mit Hilfe von größenveränderbaren `<rectangle>`-Objekten (Rechteck-Objekte) an den Seiten auffüllen. Dann wird Ihre GUI wieder größenveränderbar sein. Stellen Sie sich zum Beispiel vor, Sie haben ein 64x64 `<image>`-Objekt (Bild-Objekt) in einer horizontalen Gruppe in Ihrem Fenster. RapaGUI wird nicht in der Lage sein, die Größe dieses Fensters zu ändern, es sei denn, Sie fügen Rechteck-Objekte an den Seiten Ihres Bild-Objekts hinzu, weil RapaGUI ein Objekt finden muss, das in der Größe verändert werden

kann. So müssen Sie darauf achten, dass nicht größenveränderbare Objekte in Ihrer Benutzeroberfläche immer mit größenveränderbaren Objekten aufgefüllt werden. Übrigens, seien Sie vorsichtig mit dem Tag `<label>`: Widget der Label-Klasse (Beschriftungen) sind eigentlich nicht größenveränderbar, weil größenveränderbare Beschriftungen ziemlich unangenehm aussehen!

F: RapaGUI läuft nicht auf meiner Linux-Distribution, obwohl ich GTK+ installiert habe. Was könnte der Grund dafür sein?

A: Stellen Sie zunächst sicher, dass Sie die richtige RapaGUI-Version verwenden. RapaGUI für Linux ist in Versionen für GTK+ 2 und GTK+ 3 verfügbar. Normalerweise ist auf Linux-Distributionen nur eine der beiden GTK-Versionen installiert, sodass Sie einen RapaGUI-Build verwenden müssen, das zu Ihrer installierten GTK-Version passt. Möglicherweise fehlen Ihnen auch einige freigegebene Objekte von Drittanbietern, die von RapaGUI benötigt werden. Um zu sehen, welche Objekte von Drittanbietern für RapaGUI benötigt werden, führen Sie den folgenden Befehl in einer Konsole aus: `readelf -d rapagui.hwp`

F: Bei der Ausführung auf einem High-DPI-Monitor erscheint mein Programm unter Windows unscharf. Warum?

A: Stellen Sie sicher, dass Sie den Tag `DPIAware` in Hollywoods Präprozessor-Anweisung `@OPTIONS` auf `True` gesetzt ist. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), Seite 21, für Details.

F: Das Treeview-Widget (Textanzeige) von RapaGUI funktioniert nicht auf Android. Warum?

A: Android hat kein natives Treeview-Widget, weshalb RapaGUI keine Treeviews auf Android unterstützen kann.

2.5 Danksagungen

RapaGUI wurde von Andreas Falkenhahn geschrieben. Das Design wurde von meinem MUI Royale Plugin inspiriert, welches nur auf AmigaOS und Kompatibeln läuft und Ende 2012 erstmals veröffentlicht wurde. Erste Experimente mit einem wxWidgets-basierten GUI-Toolkit für Hollywood wurden bereits 2013 gestartet. Ursprünglich wollte ich nur ein Wrapper-Plugin entwickeln, welches es Hollywood-Skripten erlaubt, wxWidgets zu verwenden. Aber dann wurde mir klar, dass die wxWidgets-API ziemlich kompliziert in der Anwendung sein kann und ich dachte, dass ein MUI Royale-basierter Ansatz viel benutzerfreundlicher und bequemer für GUI-Skripting sein würde. Durch die Nachahmung der MUI-Programmierparadigmen in RapaGUI konnte ich auch Versionen für AmigaOS und Kompatible veröffentlichen, was RapaGUI zum ersten plattformübergreifenden GUI-Toolkit macht, welches das native MUI-Toolkit des Amiga unterstützt. Damit hat RapaGUI das Potenzial, den Traum vieler Amiga-Anwender wahr werden zu lassen: Ein plattformübergreifendes GUI-Toolkit, das native Widgets auf allen Plattformen verwendet! RapaGUI hat diese Mission endlich erfüllt. Vielen Dank an das wxWidgets-Team und Stefan Stuntz für ihre wunderbaren GUI-Toolkits. Ein besonderer Dank geht an Vadim Zeitlin und Eric Jensen für ihre wertvolle Hilfe zu wxWidgets sowie Thore Böckelmann für seine sofortigen MUI-Fixes und

seine Offenheit für nützliche MUI-Erweiterungen, die er ebenfalls sehr schnell implementiert hat.

Ein weiterer Dank geht an Alfonso Ranieri für TheBar.mcc und codesets.library, Allan Odgaard für TextEditor.mcc, HTMLview.mcc und den Open Source-Teams TheBar.mcc, HTMLview.mcc sowie TextEditor.mcc für die Pflege dieser Klassen und das Beheben alter Fehler.

Der nächste besondere Dank geht an Dominic Widmer und Helmut Haake für die Übersetzung des Handbuchs ins Deutsche. Fehler oder Verbesserungsvorschläge bzgl. des deutschen Hollywood-Handbuchs bitte an das Übersetzungsteam richten, welches unter handbuch@gmx.ch erreicht werden kann.

Wenn Sie mich kontaktieren möchten, senden Sie bitte eine E-Mail an [andreas@airsoftsoftwair . de](mailto:andreas@airsoftsoftwair.de) oder nutzen Sie das Kontaktformular unter <http://www.hollywood-mal.com>.

3 Konzeptioneller Überblick

3.1 Anwendungsbaum

Jede RapaGUI-Anwendung ist im Grunde genommen ein Baum, der viele MOAI-Objekte enthält, aus denen sich die Anwendung zusammensetzt. Typischerweise sind MOAI-Objekte nur Widgets wie Schaltflächen, Listenansichten, Textlisten etc, können aber auch separate Fenster oder abstrakte Objekte wie Gruppenobjekte sein, die zur Berechnung des GUI-Layouts verwendet werden.

Das Wurzelement jedes Anwendungsbaum muss immer eine Instanz von der Applications-Klasse sein. Außerhalb des Applikations-Objekts darf kein MOAI-Objekt existieren. Applikations-Objekte werden durch den Aufruf von `moai.CreateApp()` erzeugt und es kann in jedem Programm nur ein Applikations-Objekt geben. Das Applikations-Objekt ist für die Abfertigung aller Ereignisse und Nachrichten zuständig, die Ihr Programm benötigt. Siehe [Abschnitt 9.1 \[Application-Klasse\], Seite 65](#), für Details.

Die wichtigsten Elemente des Applikations-Objekt sind die separaten im Vordergrund befindlichen Fenster Ihrer Anwendung. Dies können normale Top-Level-Fenster oder modale Dialoge sein, die den Rest Ihrer Anwendung blockieren, während sie geöffnet sind. Um diese Objekte in XML zu erzeugen, müssen Sie sie nur mit den Tags `<window>` und `<dialog>` erzeugen. Siehe [Abschnitt 62.1 \[Window-Klasse\], Seite 319](#), für Details. Siehe [Abschnitt 16.1 \[Dialog-Klasse\], Seite 99](#), für Details.

Jedes Fenster oder jeder Dialog muss immer genau ein Wurzelobjekt haben, das von der Group-Klasse abgeleitet sein muss. Mit der Group-Klasse können Sie ein oder mehrere Widgets in einem horizontalen, vertikalen oder Raster-basierten Layout kombinieren. Dies ist eine der wichtigsten Klassen, auf die mit den Tags `<vgroup>`, `<hgroup>` und `<colgroup>` aus XML zugegriffen werden kann. Wenn sich die Fenstergröße ändert, wird das an alle Gruppen automatisch weitergeleitet, was bedeutet, dass Sie sich nicht um die komplexe Aufgabe kümmern müssen, Ihr GUI-Layout auf der Grundlage der aktuellen Fenstergröße neu zu berechnen. RapaGUI macht das alles automatisch für Sie. Siehe [Abschnitt 18.1 \[Group-Klasse\], Seite 109](#), für Details.

Eine weitere sehr wichtige Klasse ist die Area-Klasse (Bereichs-Klasse). Alle Widgets sind untergeordnete Elemente der Area-Klasse, weil Area-Klasse im Grunde nur einen rechteckigen Bereich innerhalb des GUI-Layouts beschreibt, auf dem Widgets-abhängige Grafiken gezeichnet werden. Dies bedeutet, dass Sie alle Attribute und Methoden dieser Klasse für alle MOAI-Objekte, die Widgets sind, verwenden können. Sie können zum Beispiel die Dimensionen aller Ihrer Widgets festlegen, indem Sie nur die Attribute `Area.Width` und `Area.Height` verwenden. Dies ist jedoch normalerweise nicht notwendig, da RapaGUI automatisch die Dimensionen für alle Widgets auswählt. Wenn Sie mit der Wahl von RapaGUI nicht zufrieden sind, können Sie sie jedoch mit diesen Attributen außer Kraft setzen. Siehe [Abschnitt 10.1 \[Area-Klasse\], Seite 71](#), für Details.

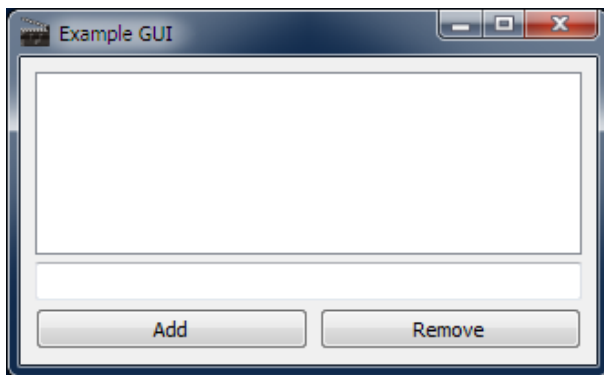
3.2 GUI-Layout

Wie Sie bereits oben gesehen haben, werden in RapaGUI GUIs vollständig mit der XML-Sprache definiert. Auf diese Weise können Sie schnell GUI-Layouts erstellen, indem Sie

einfach eine Baumhierarchie aus mehreren Fenstern, Gruppen und Widgets zusammensetzen. Hier ist ein Beispiel, wie ein Applikations-Baum in der XML-Sprache aussehen könnte:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<application>
  <window title="Example GUI">
    <vgroup>
      <listview>
        <column/>
      </listview>
      <textentry/>
      <hgroup>
        <button id="add">Add</button>
        <button id="rem">Remove</button>
      </hgroup>
    </vgroup>
  </window>
</application>
```

Und hier sehen Sie, wie diese GUI unter Windows 7 aussieht:



Wie Sie im XML sehen können, werden keine Positionskoordinaten oder -größen angegeben. Dies alles wird von RapaGUI automatisch berechnet, um ein sauberes Erscheinungsbild auf allen Systemen zu gewährleisten. Denken Sie daran, dass RapaGUI-Applikationen auf einer Vielzahl von Systemen laufen können: Auf Windows, GTK, macOS und sogar auf AmigaOS. Durch Weglassen von fest programmierten Fenster- und Widget-größen sowie -positionen kann RapaGUI automatisch die besten Größen und Positionen in Abhängigkeit von der aktuellen Bildschirm- und Schriftgröße auswählen. Natürlich können Sie RapaGUI auch zwingen, fest codierte Größen zu verwenden, indem Sie die Attribute `Area.Width` und `Area.Height` verwenden. Um eine Listenansichts-Größe von mindestens 600x400 geräteunabhängige Pixel zu erzwingen, könnte man einfach schreiben:

```
<listview width="600" height="400">
  <column/>
</listview>
```

Das Einzige, was Sie nicht ändern können, ist die Position des Widgets. Da die Position der Widgets stark von der aktuellen Schriftgröße der Benutzeroberfläche abhängt, macht es wirklich keinen Sinn, hartcodierte Widget-Positionen in einem plattformübergreifenden

GUI-Werkzeug zuzulassen, da es massive Unterschiede zwischen den verschiedenen Betriebssystemen geben wird. RapaGUI erlaubt es Ihnen also nicht, Widget-Positionen fest zu programmieren.

3.3 Ausführen von GUIs

Alles, was Sie tun müssen, um mit XML definierten GUIs zu erstellen und auszuführen, ist ein paar Zeilen Hollywood-Code zu schreiben. Nehmen wir an, Sie haben die XML-Beispieldefinition von oben in einer Datei mit dem Namen `GUI.xml` gespeichert. Zum Erstellen und Anzeigen dieser GUI aus einem Hollywood-Skript müssen Sie ein Hollywood-Skript mit den folgenden Zeilen schreiben:

```
@REQUIRE "RapaGUI"
moai.CreateApp(FileToString("GUI.xml"))
Repeat
    WaitEvent
Forever
```

Das ist alles! Wenn Sie das Hollywood-Skript von oben ausführen, wird Ihre GUI automatisch erstellt und angezeigt, unabhängig davon, ob Ihr Skript unter Windows, Linux, macOS oder AmigaOS läuft.

3.4 Initialisierung von RapaGUI

Alles, was Sie tun müssen, damit Ihr Skript RapaGUI anstelle von Hollywoods integriertem Grafik-Modul verwendet, ist die folgende Zeile am Anfang Ihres Skripts hinzuzufügen:

```
@REQUIRE "rapagui"
```

Wenn Sie Hollywood von einer Konsole aus verwenden, können Sie Ihr Skript alternativ auch so starten:

```
Hollywood test.hws -requireplugins rapagui
```

RapaGUI akzeptiert die folgenden Argumente in seinem `@REQUIRE`-Aufruf:

ScaleGUI:

Wenn RapaGUI auf einem Monitor mit hohem DPI-Wert läuft, werden alle Rastergrafiken automatisch skaliert, um sie an die DPI-Einstellungen des Monitors anzupassen. Wenn Sie dies nicht möchten, setzen Sie diesen Tag auf `False`. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), Seite 21, für Details. Wenn dieser Tag auf `True` gesetzt ist (Voreinstellung), können Sie mit dem Tag `InterpolateGUI` (siehe unten) steuern, ob beim Skalieren bilineare Interpolation verwendet werden soll oder nicht. Voreingestellt ist `True`. (V2.0)

InterpolateGUI:

Wenn `ScaleGUI` auf `True` gesetzt ist (Voreinstellung), kann mit dem Tag `InterpolateGUI` angegeben werden, ob beim Skalieren bilineare Interpolation verwendet werden soll oder nicht. Die Voreinstellung ist `True`. (V2.0)

ScaleHollywood:

Wenn RapaGUI auf einem Monitor mit hoher DPI ausgeführt wird, werden alle Hollywood-Widgets automatisch skaliert, um sich den DPI-Einstellungen des Monitors anzupassen. Wenn Sie dies nicht möchten, setzen Sie diesen Tag auf

`False`. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), Seite 21, für Details. Wenn dieser Tag auf `True` gesetzt ist (Voreinstellung), können Sie mit dem Tag `InterpolateHollywood` (siehe unten) steuern, ob beim Skalieren bilineare Interpolation verwendet werden soll oder nicht. Die Voreinstellung ist `True`. (V2.0)

`InterpolateHollywood`:

Wenn `ScaleHollywood` auf `True` gesetzt ist (Voreinstellung), kann der Tag `InterpolateHollywood` verwendet werden, um anzugeben, ob beim Skalieren bilineare Interpolation verwendet werden soll oder nicht. Beachten Sie, dass im Gegensatz zum Tag `InterpolateGUI` der Tag `InterpolateHollywood` auf `False` voreingestellt ist, was bedeutet, dass standardmäßig keine bilineare Interpolation für Hollywood-Widgets verwendet wird. (V2.0)

`HideDisplays`:

Standardmäßig blendet RapaGUI alle Hollywood-Displays beim Start aus. Wenn Sie dies nicht möchten, setzen Sie diesen Tag auf `True`. Dies kann nützlich sein, wenn RapaGUI im Hollywood-Emulationsmodus verwendet wird, d.h. ohne die GUI-Funktionalität von RapaGUI zu verwenden, sondern nur, um einige Funktionen zu erhalten, die Hollywood auf einigen Plattformen standardmäßig nicht unterstützt (z.B. Menüs unter Linux oder Kommunikation zwischen Prozessen unter Linux und macOS). (V2.0)

Hier ist ein Beispiel für die Übergabe von Argumenten an die Präprozessor-Anweisung `@REQUIRE`:

```
@REQUIRE "rapagui", {ScaleGUI = False}
```

Alternativ können Sie auch das Konsolenargument `-requiretags` verwenden, um diese Argumente zu übergeben. Weitere Informationen finden Sie im Hollywood-Handbuch.

3.5 Umgang mit Objekten

Wie Sie bereits gesehen haben, verwendet RapaGUI XML-Dateien, um MOAI-Objekte zu erstellen. Beim Anlegen von Objekten können Sie alle mit dem Buchstaben `I` gekennzeichneten Attribute im Abschnitt *Anwendbarkeit* der zugehörigen Attributdokumentation verwenden. Um beispielsweise ein Texteingabe-Objekt mit einer maximalen Länge von 80 Zeichen zu erstellen, verwenden Sie einfach das Attribut `TextEntry.MaxLength` und nehmen es in Ihre XML-Definition auf.

```
<textentry id="mystring" maxlen="80"/>
```

Sobald Ihr Objekt fertig ist, können Sie mit der Interaktion beginnen, indem Sie seine Attribute festlegen oder abrufen oder Sie ihre Methoden ausführen. Zu diesem Zweck ist es wichtig, dass Sie Ihrem Objekt einen Identifikator (ID, Identität, Kennung) mit dem Attribut `id` geben, damit die Befehle `moai.Set()`, `moai.Get()` und `moai.DoMethod()` Ihr Objekt finden kann. Im obigen Code haben wir unserem `TextEntry`-Widget (Texteingabe) die ID `mystring` zugewiesen. Hier ist ein Beispiel, wie wir jetzt mit diesem Widget interagieren können:

```
moai.Set("mystring", "text", "look")
s$ = moai.Get("mystring", "text")
DebugPrint("Always " .. s$ .. " on the bright side of life.")
```

Wie bereits oben erwähnt, sind alle Attribute und Methoden in der Dokumentation von RapaGUI vollständig beschrieben.

Das nächste, was Sie erledigen müssen, ist die Verarbeitung von Ereignissen, die von Ihrer GUI ausgelöst werden. Dies wird im nächsten Abschnitt erklärt.

3.6 Ereignisse verarbeiten

RapaGUI-Ereignisse werden genauso verarbeitet wie normale Hollywood-Ereignisse. Das bedeutet, dass Sie lediglich eine Callback-Funktion an den Befehl `InstallEventHandler()` von Hollywood übergeben müssen, und jedes Mal, wenn ein RapaGUI-Ereignis auftritt, wird Ihre Callback-Funktion aufgerufen.

Hier ist ein Beispiel, wie Sie eine Callback-Funktion für ein RapaGUI-Ereignis installieren:

```
InstallEventHandler({RapaGUI = p_EventHandler})
```

Immer wenn ein Ereignis eintritt, ruft RapaGUI die Funktion `p_EventHandler()` mit einer Tabelle als Parameter auf. In dieser Tabelle werden die folgenden Felder initialisiert:

Action: Enthält "RapaGUI".

Class: Enthält den Namen der MOAI-Klasse, von der dieses Ereignis stammt, z.B. "Choice".

Attribute:

Enthält den Namen des Klassenattributs, welches das Ereignis ausgelöst hat, z.B. "Active".

ID: Enthält die ID des MOAI-Objekts, welches dieses Ereignis ausgelöst hat, z.B. "meineauswahl".

TriggerValue:

Enthält den aktuellen Wert des Attributs, welches dieses Ereignis ausgelöst hat. Sie können dies auch herausfinden, indem Sie den Befehl `moai.Get()` auf dem Objekt anwenden, aber es ist bequemer, den aktuellen Wert direkt mit ihrer Callback-Funktion auslesen.

MOAIUserData:

Wenn dem Objekt bestimmte Benutzerdaten zugewiesen wurden, wird es in diesem Tag an die Callback-Funktion übergeben. Siehe [Abschnitt 33.8 \[MOAI.UserData\]](#), Seite 197, für Details.

NotifyData:

Wenn dem Objekt bestimmte Benachrichtigungsdaten zugewiesen wurden, wird es in diesem Tag an die Callback-Funktion übergeben.

Darüber hinaus ist es auch notwendig, RapaGUI mitzuteilen, welche Ereignisse Sie erhalten möchten. Nur Ereignisse, die Sie explizit anfordern, werden an Ihre Callback-Funktion übergeben, um die Verwaltungsdaten (Overhead) zu minimieren. Eine Ausnahme bilden Schaltflächen, Symbolleistenschaltflächen und Menüeinträge. Sie werden an Ihre Callback-Funktion weitergeleitet, auch wenn Sie sie nicht angefordert haben. Der Grund für diese Auswahl von Verwaltungsdaten liegt darin, dass normalerweise alle diese Ereignisse individuell behandelt werden müssen, da der Benutzer erwartet, dass auf Knopfdruck etwas passiert.

Alle RapaGUI-Ereignisse sind mit Attributen von MOAI-Objekten gekoppelt. Der nächste Abschnitt beschreibt das im Detail.

3.7 Benachrichtigungen der Attribute

Der Umgang mit Ereignissen von RapaGUI basiert auf den Werten bestimmter Attribute der verschiedenen MOAI-Klassen. Sie können die Werte aller Attribute überwachen, die eine Anwendbarkeit von N haben. Wenn Sie eine Benachrichtigung für einen Attributwert einrichten, führt RapaGUI Ihre Callback-Funktion aus, wenn sich dieser Attributwert ändert.

Sie können z.B. eine Benachrichtigung für das Attribut `Listview.Active` einrichten, welches das aktive Element eines Listview-Widgets (Listenansicht) ist. In diesem Fall wird Ihre Callback-Funktion immer dann aufgerufen, wenn sich das aktive Element des jeweiligen Listview-Widget ändert. Um eine Überwachung auf einem bestimmten Attribut zu installieren, müssen Sie beim Erstellen Ihres Objekts eine Benachrichtigung für dieses Attribut einrichten. Dies geschieht direkt in der XML-Datei unter Verwendung des Attributs `notify`, das von allen MOAI-Klassen akzeptiert wird:

```
<listview id="lv" notify="active">
  <column>
    <item>One</item>
    <item>Two</item>
    <item>Three</item>
  </column>
</listview>
```

Der obige Code führt Ihre Callback-Funktion immer dann aus, wenn sich das aktive Element des Listview-Objekts mit der ID `lv` ändert, weil Sie den Wert des Attributs `Listview.Active` überwachen.

Wenn Sie mehrere verschiedene Benachrichtigungen vom demselben MOAI-Objekt einrichten möchten, müssen Sie diese durch Semikolons (;) trennen, z.B:

```
<listview id="lv" notify="active; doubleclick">
  ...
</listview>
```

Wie Sie sehen können, installiert der obige Code die Überwachung für die beiden Attributen `Listview.Active` und `Listview.DoubleClick`, so dass Ihre Callback-Funktion auch dann aufgerufen wird, wenn der Benutzer auf einen Listeneintrag doppelklickt.

Alternativ können Sie Benachrichtigungen zur Laufzeit auch mit dem Befehl `moai.Notify()` aus Ihrem Code einrichten oder entfernen.

Um das gerade aktive Listenelement in Ihrer Callback-Funktion auszulesen, können Sie den folgenden Code verwenden:

```
Function p_EventHandler(msg)
  Switch msg.action
  Case "RapaGUI":
    Switch msg.attribute
    Case "Active":
      Switch msg.id
```



```

        Case "lv":
            DebugPrint("Active listview item:", msg.triggervalue)
        EndSwitch
    EndSwitch
EndSwitch
EndFunction

```

Beachten Sie jedoch, dass Attributwerte auch manuell geändert werden können. Beispielsweise könnte Ihr Programm ein bestimmtes Element der Listenansicht manuell aktivieren wollen, indem es z.B. folgenden Befehl ausführt:

```
moai.Set("lv", "active", 5) ; aktiviert Element Nummer 6
```

Da der obige Code auch den Wert des Attributs `Listview.Active` ändert, wird auch Ihre Callback-Funktion durch diesen Aufruf ausgelöst. Wenn Sie dies nicht wollen, können Sie das spezielle Attribut `MOAI.NoNotify` im Befehl `moai.Set()` verwenden. Immer wenn `MOAI.NoNotify` auf `True` gesetzt ist, wird der Wert des Attributs geändert, ohne dass irgendwelche Callback-Funktionen aufgerufen werden. Um also das aktive Element der Listenansicht ohne Callback-Funktionen zu ändern, müssen Sie einfach folgendes schreiben:

```
moai.Set("lv", "active", 5, "nonotify", True)
```

Schließlich gibt es, wie bereits im vorherigen Abschnitt erwähnt, einige Attribute, die RapaGUI immer überwacht. Dies sind: `Button.Pressed`, `Button.Selected`, `Toolbarbutton.Pressed`, `Toolbarbutton.Selected` und `MenuItem.Selected`. Da diese so häufig und weit verbreitet sind und Sie normalerweise immer Attribute wie `Button.Pressed` überwachen möchten, weil das Drücken einer Taste immer eine Reaktion hervorruft, benachrichtigt Sie RapaGUI automatisch. Daher müssen Sie nicht explizit eine Benachrichtigung von diesen Attributen anfordern, d.h. das Schreiben des Folgenden ist überflüssig:

```
<button id="btn" notify="pressed">Click me</button>
```

Stattdessen können Sie einfach schreiben:

```
<button id="btn">Click me</button>
```

Dasselbe gilt für Menüeinträge und Schaltflächen in der Symbolleiste (Toolbar).

3.8 Dynamische Objekte

Die meisten Programme erstellen ihre GUI durch einen einfachen Aufruf von `moai.CreateApp()`, dem eine XML-Definition übergeben wird, welche die komplette Benutzeroberfläche des gesamten Programms enthält. In manchen Situationen kann es jedoch notwendig sein, MOAI-Objekte nach dem Aufruf von `moai.CreateApp()` zu erzeugen und in die bestehende Anwendung einzufügen.

Dies ist möglich, indem Sie den Befehl `moai.CreateObject()` verwenden. Dieser Befehl ermöglicht es Ihnen, ein MOAI-Objekt aus einer XML-Definition zu erstellen, ähnlich wie `moai.CreateApp()`, außer dass `moai.CreateObject()` Ihnen nicht erlaubt ein GUI-Application-Objekt (Anwendung) zu erstellen, da es nur ein GUI-Application-Objekt geben kann.

Zum Beispiel können Sie eine Schaltfläche erstellen, indem Sie `moai.CreateObject()` wie folgt verwenden:

```
moai.CreateObject([[<button id="mybutton">Click me</button>]])
```

Es ist sehr wichtig, eine ID für das MOAI-Objekt zu setzen, welches Sie mit `moai.CreateObject()` erstellen, da der Identifikator später für das Objekt benötigt wird. Nachdem wir die Schaltfläche erstellt haben, können wir sie in ein bestehendes Fensterlayout einfügen. Nehmen wir an, unser Fensterlayout sieht momentan so aus:

```
<window>
  <hgroup id="mygroup">
    <button id="ok">OK</button>
    <button id="cancel">Cancel</button>
  </hgroup>
</window>
```

Wir könnten nun die Methode `Group.Insert` verwenden, um die neu erstellte Schaltfläche nach der Schaltfläche "OK" einzufügen. Dazu müssten wir den folgenden Code verwenden:

```
moai.DoMethod("mygroup", "initchange")
moai.DoMethod("mygroup", "insert", "mybutton", "ok")
moai.DoMethod("mygroup", "exitchange")
```

Sie sehen, dass wir auch die Methoden `Group.InitChange` und `Group.ExitChange` aufrufen. Das ist sehr wichtig. Diese Methoden müssen immer dann ausgeführt werden, wenn Sie die Elemente einer Gruppe ändern, d.h. wenn Sie Elemente entfernen oder hinzufügen. RapaGUI benötigt den Aufruf dieser Methoden, da nach dem Entfernen oder Hinzufügen von Gruppenelementen ein Neuzeichnen des Fensterlayout notwendig wird. Deshalb reicht es nicht aus, einfach `Group.Insert` oder eine andere Methode aufzurufen, die die Anzahl der Gruppenelemente ändert.

Natürlich können wir auch Elemente aus Gruppen entfernen. Dies ist mit `Group.Remove` möglich. Der folgende Code entfernt die Schaltfläche "Abbrechen" aus dem obigen Fensterlayout:

```
moai.DoMethod("mygroup", "initchange")
moai.DoMethod("mygroup", "remove", "cancel")
moai.DoMethod("mygroup", "exitchange")
```

Nach diesem Aufruf ist das MOAI-Objekt mit der ID "cancel" ein losgelöstes Objekt geworden. Das bedeutet, dass Sie es jetzt auch wieder an eine Gruppe anhängen können, indem Sie `Group.Insert` oder eine ähnliche Methode verwenden. Beachten Sie, dass alle Methoden, die MOAI-Objekte in bestehende Layouts einfügen, nur losgelöste Objekte akzeptieren. Es ist nicht möglich, dasselbe MOAI-Objekt in mehrere Gruppen einzufügen. Sobald Sie ein MOAI-Objekt in eine Gruppe (oder einen Menübaum) einfügen, ändert es seinen Zustand von losgelöst in angehängt. Sobald ein MOAI-Objekt im angehängten Zustand ist, kann es nicht mehr an Methoden übergeben werden, die ein losgelöstes Objekt erwarten. Um ein MOAI-Objekt vom angefügten in den gelösten Zustand zu versetzen, müssen Sie eine Methode wie `Group.Remove` verwenden, um ein MOAI-Objekt von einer Gruppe zu trennen.

Als letzte Anmerkung: Der XML-Code, den Sie an `moai.CreateObject()` übergeben, kann auch ein vollständiger Baum sein, z.B. könnten Sie ein vollständiges Fenster mit nur einem Aufruf mit dem Befehl `moai.CreateObject()` erstellen. Das ist alles möglich. `moai.CreateObject()` ist nicht auf das Erzeugen einzelner MOAI-Objekte beschränkt, sondern kann auch mehrere Objekte für Sie erstellen. Wenn Sie jedoch ein Fenster oder

ein Dialog mit `moai.CreateObject()` erstellen, vergessen Sie nicht, es zum Applications-Objekt hinzuzufügen, indem Sie `Application.AddWindow` aufrufen. Fensterobjekte sind standardmäßig im gelösten Zustand, sofern sie keinem Application-Objekt hinzugefügt wurden. Für Dialoge können Sie einfach `moai.CreateDialog()` verwenden, welches `Application.AddWindow` einschließlich aufruft.

3.9 Größenveränderung

Eine der wichtigsten Eigenschaften von RapaGUI ist die Fähigkeit, das gesamte GUI-Layout automatisch neu zu berechnen, wenn die Größe des Fensters mit den Widgets geändert wird. Dies ist jedoch nur möglich, wenn alle Gruppen in einem Fenster in der Größe geändert werden können, da sonst die Größe des Fensters unverändert bleibt und nicht skalierbar ist. Um frei skalierbare Fenster zu erstellen, müssen Sie wissen, welche Widgets in der Größe geändert werden können und welche nicht. Beachten Sie die folgende GUI-Definition:

```
<window>
  <vgroup>
    <button id="btn">Hello World!</button>
  </vgroup>
</window>
```

Dieses Fenster ist nur horizontal skalierbar, da Schaltflächen-Widgets standardmäßig nur horizontal gedehnt/gestaucht werden können. Sie sind standardmäßig nicht vertikal skalierbar, da Schaltflächen, die ihre vertikale Größe ändern, ziemlich hässlich aussehen. Normalerweise ist die Höhe einer Schaltfläche an die Standardschaltflächenhöhe gebunden, die derzeit auf dem Betriebssystem benutzt wird.

Wenn Sie möchten, dass dieses Fenster in der Größe veränderbar ist, haben Sie mehrere Möglichkeiten: Die meistgebrauchte Möglichkeit ist das Einfügen von leerem Raum mit dem Rechteck-Objekt `Rectangle`-Klasse. Diese sind in alle Richtungen veränderbar. Wenn Sie also ein Objekt `<rectangle>` einfügen, wird Ihr Fenster plötzlich in alle Richtungen skalierbar:

```
<window>
  <vgroup>
    <button id="btn">Hello World!</button>
    <rectangle/>
  </vgroup>
</window>
```

Alternativ können Sie auch ein anderes Widget einfügen, das vertikal skalierbar ist, z.B. eine Listenansicht (`Listview`). Oder Sie können die Schaltfläche sogar vertikal skalierbar machen, indem Sie das Attribut `Area.FixHeight` auf `False` setzen:

```
<window>
  <vgroup>
    <button id="btn" fixheight="false">Hello World!</button>
  </vgroup>
</window>
```

Aber das sieht nicht so schön aus, weil der Benutzer jetzt eine Schaltfläche mit einer Höhe von einigen hundert Pixeln haben könnte. Deshalb ist der Ansatz mit `<rectangle>` die

am häufigsten verwendete Methode, um Füllbereiche für nicht veränderbare Objekte einzufügen. Deshalb hier eine Übersicht über die Größe der einzelnen Widgets, die von RapaGUI unterstützt werden:

Busybar-Klasse (Beschäftigung):

Horizontal veränderbar.

Button-Klasse (Schaltflächen):

Horizontal veränderbar.

Checkbox-Klasse (Auswahlkästchen):

Nicht veränderbar.

Choice-Klasse (Auswahl):

Horizontal veränderbar.

Combobox-Klasse (Textlisten):

Horizontal veränderbar.

Hollywood-Klasse:

Nicht veränderbar.

HLine-Klasse (HorizontalLinie):

Horizontal veränderbar.

HSpace-Klasse (HorizontalAbstand):

Vertikal veränderbar.

HTMLview-Klasse (HTMLAnsicht):

In alle Richtungen veränderbar.

Image-Klasse (Bild):

Nicht veränderbar.

Label-Klasse (Beschriftung):

Nicht veränderbar.

Listview-Klasse (Listenansicht):

In alle Richtungen veränderbar.

Pageview-Klasse (Seitenansicht):

In alle Richtungen veränderbar.

Popcolor-Klasse (Farbdialog):

Horizontal veränderbar.

Popfile-Klasse (Dateidialog):

Horizontal veränderbar.

Popfont-Klasse (Schriftdialog):

Horizontal veränderbar.

Poppath-Klasse (Verzeichnisdiallog):

Horizontal veränderbar.

Progressbar-Klasse (Fortschrittsbalken):

Horizontal skalierbar für horizontale Fortschrittsbalken. Vertikal skalierbar für vertikale Fortschrittsbalken.

Radio-Klasse:

Nicht veränderbar.

Rectangle-Klasse (Rechteck):

In alle Richtungen veränderbar.

Scrollbar-Klasse (Bildlaufleisten):

Horizontal skalierbar für horizontale Bildlaufleisten. Vertikal skalierbar für vertikale Bildlaufleisten.

Scrollcanvas-Klasse (Bildlaufleinwand):

In alle Richtungen veränderbar.

Scrollgroup-Klasse (Bildlaufgruppen):

In alle Richtungen veränderbar.

Slider-Klasse (Schieberegler):

Horizontal skalierbar für horizontale Schieberegler. Vertikal skalierbar für vertikale Schieberegler.

Text-Klasse:

Horizontal veränderbar.

Texteditor-Klasse:

In alle Richtungen veränderbar.

Textentry-Klasse (Texteingabe):

Horizontal veränderbar.

TextView-Klasse (Textanzeige):

In alle Richtungen veränderbar.

TreeView-Klasse (Baumansicht):

In alle Richtungen veränderbar.

VLine-Klasse (Vertikallinie):

Vertikal veränderbar.

VSpace-Klasse (Vertikalabstand):

Horizontal veränderbar.

Natürlich können Sie diese Voreinstellungen ändern, indem Sie die Attribute `Area.FixWidth` und `Area.FixHeight` entsprechend setzen, um bestimmte Größeneinstellungen zu aktivieren oder zu deaktivieren. Aber das ist oft nicht empfehlenswert, da es ziemlich hässlich aussieht, wenn z.B. ein Texteingabe-Widget plötzlich vertikal skalierbar ist. Dies würde den Benutzer unnötig verwirren, da er am Ende ein Texteingabe-Widget mit einer Höhe von 300 Pixeln haben könnte, welches aber nur die Eingabe einer Textzeile erlaubt. Daher ist es die beste Idee, die Voreinstellungen des Betriebssystems für die Standard-Widgets zu respektieren.

Natürlich gibt es auch Widgets, bei denen es völlig akzeptabel ist, die Voreinstellungen von oben zu überschreiben. Zum Beispiel sind Widgets, die von der `Hollywood`-Klasse abgeleitet sind, standardmäßig nicht skalierbar. Aber natürlich sollten Sie `Area.FixWidth` und `Area.FixHeight` verwenden, um die Größe des Widgets an Ihre persönlichen Bedürfnisse anzupassen.

3.10 Anwendbarkeit

In der Dokumentation jedes Objektattributs finden Sie Informationen über die Anwendbarkeit dieses Attributs. Die Attribut-Anwendbarkeit wird in Form einer Kombination der vier Buchstaben I, S, G und N beschrieben. Dies teilt Ihnen die verschiedenen Möglichkeiten mit, in denen das Attribut verwendet werden kann.

Hier ist eine Erklärung der verschiedenen Möglichkeiten der Anwendbarkeit:

- I Das Attribut kann beim Anlegen des Objekts in der XML-Datei verwendet werden. (Initialisierungszeit)
- S Das Attribut kann zur Laufzeit mit `moai.Set()` verwendet werden.
- G Das Attribut kann mit `moai.Get()` zur Laufzeit verwendet werden.
- N Benachrichtigungen zu diesem Attribut sind entweder über das Attribut "Notify" in der XML-Definition oder durch Aufruf von `moai.Notify()` zur Laufzeit möglich.

Wenn zum Beispiel ein Attribut eine Anwendbarkeit von nur "I" hat, dann kann dieses Attribut nur während der Initialisierung des Objekts verwendet und kann später nicht mit `moai.Set()` geändert werden. Wenn ein Attribut eine Anwendbarkeit von nur "S" hat, ist es andererseits nicht möglich, das Attribut bereits zur Initialisierungszeit in der XML-Datei anzugeben. Attribute, die eine Anwendbarkeit von "ISGN" haben, können mit allen Möglichkeiten verwendet werden.

3.11 Geräteunabhängige Pixel

Alle Positions- und Größenangaben in RapaGUI müssen in geräteunabhängigen Pixeln angegeben werden. Dies hat den Vorteil, dass sie unabhängig von der DPI-Einstellung des aktuellen Monitors sind, sodass Ihr Programm beispielsweise auf Monitoren mit hohem DPI-Wert immer noch korrekt aussieht.

Betrachten Sie die folgende Deklaration eines Fensters:

```
<window width="640" height="480">
  ...
</window>
```

Dadurch wird ein Fenster deklariert, dessen Breite 640dpi und dessen Höhe 480dpi beträgt. Wenn der aktuelle Monitor keine DPI-Skalierung verwendet (z.B. 96dpi unter Windows, 72dpi unter macOS), sind die geräteunabhängigen Pixel identisch mit den physischen Pixeln, so dass die Größe des Fensters 640x480 Pixel beträgt. Wenn der Monitor jedoch für die DPI-Skalierung konfiguriert ist, werden die geräteunabhängigen Pixel automatisch an die DPI-Einstellung des Monitors angepasst. Auf einem 192-dpi-Monitor (d.h. 200% Skalierung unter Windows) betragen die physischen Abmessungen des Fensters auf dem Bildschirm beispielsweise 1280x960 Pixel, da RapaGUI die geräteunabhängigen Pixel automatisch an die DPI-Einstellung des aktuellen Monitors anpasst.

Beachten Sie, dass es im Allgemeinen eine gute Idee ist, bestimmte Positionen und Größen nicht fest zu kodieren und RapaGUI diese automatisch aus Ihrem Layout ermitteln zu lassen. Dies garantiert ein Höchstmaß an Kompatibilität zwischen verschiedenen Betriebssystemen und deren spezifischen Fenstermanagern, Schriftarten, Designs und Dekorationen.

Lesen Sie auch unbedingt das Kapitel über High-DPI-Unterstützung, um mehr über andere Punkte zu erfahren, die Sie beim Entwerfen eines Programms beachten sollten, das mit High-DPI-Monitoren kompatibel sein sollte. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), [Seite 21](#), für Details.

3.12 High-DPI-Unterstützung (hochauflösende Displays)

RapaGUI bietet volle Unterstützung für High-DPI-Displays. Normalerweise erledigt RapaGUI alles automatisch für Sie, so dass Ihr Programm unabhängig von der DPI-Einstellung des Monitors gleich aussehen sollte. Dies wird erreicht, indem alle Positions- und Größenwerte in geräteunabhängige Pixel berechnet werden, wodurch RapaGUI-Programme gut auf verschiedene Monitor-DPI-Einstellungen skaliert werden können. Darüber hinaus skaliert RapaGUI automatisch alle Rastergrafiken, z.B. Schaltflächenbilder, die mit `Button.Icon` eingestellt wurden, um sie an die DPI-Einstellung des Monitors anzupassen.

Beachten Sie, dass die automatische Skalierung von Rastergrafiken durch RapaGUI dazu führen kann, dass Bilder auf Monitoren mit hohen DPI-Werten unscharf werden. Sie können dies ändern, indem Sie entweder mehrere Bilder für verschiedene DPI-Einstellungen bereitstellen, anstatt nur ein einziges Bild zu verwenden, oder indem Sie Vektorgrafiken verwenden, die ohne Qualitätsverluste auf jede Größe skaliert werden können.

Angenommen, Sie möchten Ihrer Schaltfläche ein 32x32px-Icon hinzufügen, indem Sie das Attribut `Button.Icon` verwenden. Wenn Sie `Button.Icon` auf einen Rasterpinsel setzen und der Monitor des Benutzers eine DPI-Skalierung von 200% aktiviert hat, skaliert RapaGUI das Bild auf 64x64px, was unscharf aussieht. Sie können also entweder `Button.Icon` auf den Identifikator eines Vektorpinsels setzen oder ein Hollywood-Icon erstellen, 32x32px und 64x64px (und möglicherweise mehr) Bilder hinzufügen und dann dieses Icon an `Button.Icon` übergeben.

Beachten Sie, dass Sie in diesem Fall auch das Attribut `Button.IconType` auf `Icon` setzen müssen, um RapaGUI mitzuteilen, dass Sie den Identifikator eines Icons anstelle eines Pinsels übergeben haben, was der Standardtyp für `Button.Icon` ist. In diesem Fall verwendet RapaGUI das 32x32px-Bild im Icon, falls auf dem Monitor keine DPI-Skalierung aktiviert ist oder das 64x64px-Bild, wenn der Monitor eine DPI-Skalierung von 200% aktiviert hat. Wenn die DPI-Skalierung zwischen 100% und 200% liegt, skaliert RapaGUI das Bild innerhalb des Icons, dessen Größe der gewünschten Zielgröße am nächsten kommt.

Standardmäßig erwarten alle Widgets, die Bilder unterstützen, dass Sie Hollywood-Pinsel übergeben. Wenn Sie stattdessen immer Hollywood-Icons übergeben möchten, können Sie auch das globale Attribut `Application.UseIcons` auf `True` setzen. Dadurch wird die Voreinstellung aller Tags, die Bilder von Hollywood-Pinseln akzeptieren, in Hollywood-Icons geändert.

Wie oben beschrieben, skaliert RapaGUI bei der Übergabe von Hollywood-Pinseln an RapaGUI, die Rasterdaten enthalten, diese Rasterpinsel automatisch, um sie an die DPI-Einstellungen des aktuellen Monitors anzupassen. Wenn Sie das nicht möchten, können Sie den Tag `ScaleGUI` auf `False` setzen, wenn Sie das RapaGUI-Plugin mit der Präprozessor-Anweisung `@REQUIRE` deklarieren. Siehe [Abschnitt 3.4 \[Initialisierung von RapaGUI\]](#), [Seite 11](#), für Details. Die bilineare Interpolation während der Skalierung kann ausgeschaltet werden, indem der Tag `InterpolateGUI` bei `@REQUIRE` auf `False` gesetzt wird. Alternativ

kann die automatische Bildskalierung auch für jedes einzelne Widget konfiguriert werden. Bei Schaltflächenbildern können Sie beispielsweise die automatische Bildskalierung steuern, indem Sie den Tag `Button.IconScale` setzen.

Außerdem werden alle Hollywood-Widgets automatisch skaliert, um sie an die DPI-Einstellungen des aktuellen Monitors anzupassen. Wenn Sie das nicht möchten, können Sie den Tag `ScaleHollywood` auf `False` setzen, wenn Sie das RapaGUI-Plugin mit `@REQUIRE` deklarieren. Siehe [Abschnitt 3.4 \[Initialisierung von RapaGUI\], Seite 11](#), für Details. Hier kann die bilineare Interpolation während der Skalierung aktiviert werden, indem der Tag `InterpolateHollywood` bei `@REQUIRE` auf `True` gesetzt wird.

Mit der `Scrollcanvas`-Klasse sind die Abmessungen, die Sie in `Scrollcanvas.VirtWidth` und `Scrollcanvas.VirtHeight` angeben, standardmäßig in geräteunabhängigen Pixeln und RapaGUI wendet automatisch den Skalierungsfaktor des Systems auf den vom Befehl `Scrollcanvas.Paint` gezeichneten Inhalt an. Wenn Sie eine fein abgestimmte Kontrolle wünschen, können Sie das Attribut `Scrollcanvas.AutoScale` auf `False` setzen. In diesem Fall werden `Scrollcanvas.VirtWidth` und `Scrollcanvas.VirtHeight` als physikalische Pixel interpretiert und es erfolgt keine automatische Skalierung, damit Ihre Malfunktion hochauflösende Grafiken ohne Qualitätsverlust durch Skalierung zeichnen kann.

Denken Sie schließlich daran, dass Sie den DPI-bewahr Modus unter Windows explizit aktivieren müssen, wenn Ihre Programme nativ High-DPI-Modi unterstützen soll. Dies kann durch Setzen des Tags `DPIAware` in der Präprozessor-Anweisung `@OPTIONS` auf `True` erfolgen, z.B:

```
@OPTIONS {DPIAware = True}
```

Wenn Sie dies nicht tun, ist Ihr Programm nicht DPI-fähig, was bedeutet, dass Windows es in einem speziellen Kompatibilitätsmodus ausführt. In diesem Kompatibilitätsmodus skaliert Windows Ihr Programm entsprechend der DPI-Einstellung des Monitors, dies führt jedoch zu einem unscharfen Erscheinungsbild. Aus diesem Grund wird empfohlen, wie oben bereits aufgeführt, `DPIAware` auf `True` zu setzen.

Beachten Sie, dass Sie unter Linux die GTK+ 3 Version von RapaGUI verwenden sollten, wenn Sie High-DPI-Modi unterstützen möchten, da GTK+ 2 High-DPI-Modi nicht sehr gut unterstützt.

3.13 Tastaturkürzel

RapaGUI ermöglicht es Ihnen, Tastenkombinationen ganz leicht einzurichten, indem Sie einfach ein Unterstreichungszeichen ("_") vor dem Zeichen verwenden, das Sie als Tastenkombination einrichten möchten. Sie sollten Tastaturkürzel immer unterstützen, weil viele Benutzer sie lieber als die Maus verwenden. Besonders wenn es um Aktionen geht, die Dutzende Male wiederholt werden müssen, ist es viel einfacher, die Tastatur zu verwenden. Mit der Maus müssen Sie ständig zu bestimmten Untermenüs navigieren, die tief in der Menühierarchie versteckt sind. Die folgenden Klassen unterstützen Tastaturkürzel, die durch den Unterstrich definiert werden:

- Button-Klasse (Schaltflächen)
- Checkbox-Klasse (Auswahlkästchen)
- Label-Klasse (Beschriftungen)
- Menu-Klasse

- MenuItem-Klasse
- Radio-Klasse
- Slider-Klasse
- Text-Klasse
- Toolbarbutton-Klasse (Symbolleistenschaltflächen)

Hier ist ein Beispiel, wie Sie zwei Tastenkombinationen für Widgets einrichten, die von Button-Klasse abgeleitet sind:

```
<hgroup>
  <button id="ok">_OK</button>
  <button id="cancel">_Cancel</button>
</hgroup>
```

Der obige XML-Code wird "O" und "C" als Tastaturkürzel einrichten. Unter Windows muss der Benutzer mit ALT+O die Schaltfläche "OK" und mit ALT+C die Schaltfläche "Abbrechen" auswählen. Auf anderen Plattformen sind die Kombinationen manchmal unterschiedlich, z.B. unter macOS müssen Sie die Taste COMMAND anstelle der Taste ALT verwenden.

Da die Label-Klasse keine Widgets erstellt, die vom Benutzer gesteuert werden können, aktiviert ein Tastaturkürzel für Beschriftungen einfach das nächste Widget im GUI-Layout, das normalerweise das Widget ist, das durch das Label beschrieben wird. Beispielsweise:

```
<hgroup>
  <label>_Name</label>
  <textentry/>
</hgroup>
```

Wie Sie sehen, wurde im obigen XML-Code "N" als Tastenkombination eingerichtet, indem ein Unterstrich verwendet wurde. Wenn Sie ALT+N drücken, wird das Texteingabe-Widget aktiviert, da Beschriftungen den Fensterfokus nicht übernehmen können.

In dem seltenen Fall, dass Sie den Unterstrich als Beschriftung für eines Ihrer Widgets verwenden und nicht automatisch in ein Tastaturkürzel konvertieren möchten, können Sie einfach das Attribut `Area.NoAutoKey` auf `True` setzen. Wenn dies eingestellt ist, zeigt RapaGUI einfach den Unterstrich an und behandelt ihn nicht als Sonderzeichen für ein Tastaturkürzel. Hinweis: Da Menü- und Menüobjekte nicht von der Area-Klasse abgeleitet sind, müssen Sie `Menu.NoAutoKey` und `MenuItem.NoAutoKey` verwenden, wenn Sie die automatische Verknüpfungserzeugung für Menü- oder Menüobjekte deaktivieren möchten.

Wenn Sie komplexere Tastenkombinationen (z. B. bestimmte Funktions- oder Steuertasten, Tastenkombinationen) verwenden wollen, müssen Sie eine Tastaturkürzel-Tabelle mit der Accelerator-Klasse für Ihr Fenster einrichten. Die Accelerator-Klasse ermöglicht Ihnen die Definition von erweiterten Tastaturkürzeln, die auch unabhängig von Widgets in Ihrem Fenster sind. Wenn Ihre komplexen Tastenkombinationen jedoch mit Menü-Elementen verknüpft sind, müssen Sie nicht die Accelerator-Klasse verwenden, sondern Sie können das Attribut `MenuItem.Shortcut` einfach auf die gewünschte Verknüpfung setzen. Dadurch wird automatisch ein Tastaturkürzel eingerichtet. Siehe [Abschnitt 32.6 \[MenuItem.Shortcut\]](#), Seite 191, für Details. Siehe [Abschnitt 7.1 \[Accelerator-Klasse\]](#), Seite 61, für Details.

3.14 Textformatierungs-codes

Widgets vom Typ Textview-Klasse (Textanzeige) und Texteditor-Klasse unterstützen die Textformatierung, wenn das Attribut `Styled` dieser Klassen auf `True` gesetzt ist. RapaGUI unterstützt die Textformatierung durch spezielle Steuer-codes, die in diesem Abschnitt beschrieben werden.

Formatierungs-codes beginnen immer mit einem Escape-Zeichen, gefolgt von einer Zeichenfolge, die den Formatierungscode beschreibt. In der Dezimalschreibweise entspricht das Escape-Zeichen dem ASCII-Code 27, der in Oktalzahl 33 und in Hexadezimalschreibweise \$1B ist. Bei der Verwendung von Formatierungs-codes ist Vorsicht geboten, da sie in XML-Dateien und Hollywood-Quelldateien unterschiedlich verwendet werden. In XML-Dateien wird die Oktalnotation verwendet, d.h. sie starten eine Escape-Sequenz mit einem Backslash und der Oktalzahl 33 (`'\33'`). In Hollywood-Quell-codes werden Oktalzahlen nach einem umgekehrten Schrägstrich jedoch nicht unterstützt. Hollywood erwartet immer nach einem umgekehrten Schrägstrich den ASCII-Code in Dezimalschreibweise. Deshalb müssen Sie `'\27'` verwenden, um eine Escape-Sequenz im Hollywood-Code zu initiieren.

Um diesen Unterschied etwas besser zu verdeutlichen, lassen Sie uns zwei Beispiele anschauen. Hier ist ein Beispiel für die Erstellung eines fettgedruckten Textobjekts in einer XML-Datei. Fettgedruckter Text wird aktiviert, indem nach dem Escape-Zeichen das Zeichen `'b'` verwendet wird:

```
<textview styled="true" id="mytext">\33bBold text</textview>
```

Sie sehen, dass hier die Oktalschreibweise verwendet wird, da XML-Dateien nach einem umgekehrten Schrägstrich eine Oktalzahl erwarten. In Hollywood ist das anders, denn Hollywood erwartet ein Dezimalzeichen nach einem umgekehrten Schrägstrich. Beim nächsten Beispiel sehen Sie, wie Sie Escape-Codes angeben müssen, wenn Sie sie aus einer Hollywood-Quelldatei verwenden:

```
moai.Set("mytext", "text", "\27bBold text")
```

Sie können sehen, dass der Code derselbe ist, außer dass wir `\27b` anstelle von `\33b` verwenden, da Hollywood nach einem umgekehrten Schrägstrich immer Dezimalzahlen anstelle von Oktalzahlen verwendet.

Die folgenden Formatierungs-codes werden derzeit unterstützt:

`\33u` Setzt den Textstil auf Unterstrichen.

`\33b` Setzt den Textstil auf Fett.

`\33i` Setzt den Textstil auf Kursiv.

`\33n` Setzt den Textstil zurück auf Normal.

`\33P[RRGGBB]`

Ändern Sie die Frontfarbe in die angegebene RGB-Farbe. Die RGB-Farbe muss in Form von sechs Hexadezimalziffern RRGGBB angegeben werden. Auf AmigaOS und Kompatible erfordert dies MUI 4.0.

3.15 Implementierung von Hilfetexten

RapaGUI unterstützt mehrere Möglichkeiten, kontextsensitive Hilfetexte in Ihrem Programm zu implementieren. Zuerst einmal kann jedes Widgets Tooltips haben, die auftauchen, nachdem die Maus für einige Zeit über einem Widget schwebt. Dies geschieht über

das Attribut `Area.Tooltip`. Da `Area`-Klasse (Bereich) die Oberklasse für alle Widgets ist, können Sie dieses Attribut verwenden, um Tooltip-Hilfe zu all Ihren Widgets hinzuzufügen. Hier ist ein Beispiel:

```
<vgroup>
  <listview tooltip="List of loaded video files">
    <column/>
  </listview>
  <button id="btn1" tooltip="Plays a video stream">Play</button>
  <button id="btn2" tooltip="Stops a video stream">Stop</button>
  <button id="btn3" tooltip="Exits the program.">Quit</button>
</vgroup>
```

Darüber hinaus unterstützen Menüeinträge und Symbolleistenflächen die Attribute `MenuItem.Help` und `Toolbarbutton.Help`. Wenn in Ihrem Fenster eine Statusleiste angebracht ist, wird der Text, den Sie in diesen Attributen angeben, automatisch in der Statusleiste angezeigt, wenn sich der Mauszeiger über dem entsprechenden Menüeintrag oder der entsprechenden Schaltfläche der Symbolleiste befindet. Diese Art von visuellem Feedback ist sehr hilfreich, da er sofort in der Statusleiste angezeigt wird und der Benutzer nicht wie bei Tooltips einige Sekunden warten muss.

3.16 Kontextmenüs

RapaGUI unterstützt Kontextmenüs für jedes seiner Widget-Klassen. Kontextmenüs werden den einzelnen Widgets mithilfe des Attributs `Area.ContextMenu` zugewiesen. Da die `Area`-Klasse (Bereich) die Oberklasse für alle Widgets ist, können Sie dieses Attribut für jedes MOAI-Objekt verwenden, das ein Widget erstellt. Wenn Sie `Area.ContextMenu` für ein Widget definiert haben, zeigt RapaGUI automatisch das Kontextmenü an, wenn der Benutzer mit der rechten Maustaste klickt, während sich der Mauszeiger über einem Widget befindet, an das ein Kontextmenü angehängt ist.

`Area.ContextMenu` erwartet ein Objekt, das von der `Menu`-Klasse als Argument abgeleitet ist, so dass Sie zuerst ein solches MOAI-Objekt für Ihr Kontextmenü in XML erstellen müssen. Es ist sehr wichtig zu beachten, dass Sie Ihre Menüs im Bereich `<application>` definieren müssen, da Menüs globale Objekte sind und später nur an Fenster oder Widgets angehängt werden. Aus diesem Grund ist es nicht erlaubt, Menüs in einem Abschnitt `<window>` zu definieren.

Hier ist ein Beispiel, in dem wir ein Kontextmenü zum Ausschneiden, Kopieren und Einfügen in ein Objekt vom Typ `Texteditor`-Klasse einfügen:

```
<menu title="Context menu" id="ctxtmenu">
  <item>Cut</item>
  <item>Copy</item>
  <item>Paste</item>
</menu>

<window>
  ...
  <texteditor contextmenu="ctxtmenu"/>
  ...
```

```
</window>
```

Beachten Sie, dass bei der Verwendung der Menu-Klasse zur Erstellung von Kontextmenüs das Attribut `Menu.Title` nur unter AmigaOS und kompatiblen Betriebssystemen verwendet wird. Kontextmenüs unter Windows, Linux und macOS zeigen keinen Titel an.

Beachten Sie auch, dass Kontextmenü-Ereignisse über den standardmäßigen Mechanismus `MenuItem.Selected` und nicht über eine speziellen Kontextmenü-Callback-Funktion ausgeliefert werden. Da Sie dasselbe Menüobjekt als Kontextmenü für mehrere Widgets verwenden können, benötigen Sie eine Möglichkeit, das Widget herauszufinden, dessen Kontextmenü das Ereignis ausgelöst hat. Um Ihnen diese Informationen zu geben, enthält die Ereignismeldung einen zusätzlichen Eintrag mit dem Namen "Parent", der die ID des Widgets des Kontextmenüs enthält. Auf diese Weise können Sie dasselbe Menüobjekt als Kontextmenü für mehrere übergeordnete Widgets wiederverwenden.

AmigaOS-Benutzer beachten bitte, dass MUI keine Kontextmenü-Objekte in Fenstern freigeben kann. Sie dürfen nicht dasselbe Menüobjekt mit Widgets verwenden, die sich in verschiedenen Fenstern befinden. Ein Menüobjekt für mehrere Widgets im selben Fenster zu verwenden, funktioniert, aber nicht für Widgets in einem anderen Fenster. Alle anderen Plattformen haben diese Einschränkung nicht, nur AmigaOS ist hiervon betroffen.

3.17 Internationalisierung

RapaGUI ermöglicht es Ihnen, Ihren Programmen auf einfache Weise Internationalisierungsunterstützung (`i18n`) hinzuzufügen, da es sich nahtlos in das Katalogsystem von Hollywood integriert. Um mehrere Sprachen in Ihrem Programm zu unterstützen, müssen Sie also lediglich einen Katalog mit allen sprachabhängigen Texten Ihres Programms erstellen und beim Start öffnen, z.B. mit der Präprozessor-Anweisung `@CATALOG`:

```
@CATALOG "MyApp.catalog"
```

Sie können dann einzelne Katalogeinträge ansprechen, indem Sie entweder das Attribut `MOAI.I18N` oder die Direktive `@i18n:` verwenden. Alle Attribute, die ein Zeichenkette-Argument akzeptieren, akzeptieren auch das Attribut `MOAI.I18N`, mit dem Sie einen Katalog-Zeichenketten-Index angeben können, der verwendet werden soll, wenn ein Katalog für die Systemsprache des Benutzers verfügbar ist. Der Zeichenkettenindex kann entweder ein absoluter numerischer Wert oder eine Hollywood-Konstante sein. Um beispielsweise eine mehrsprachige Schaltfläche zu erstellen, können Sie folgendes tun:

```
<button i18n="0">Click me</button>
```

Bei einem solchen XML-Code wird der Text "Click me" nur verwendet, wenn die Systemsprache des Benutzers Englisch ist oder RapaGUI keinen Katalog für die aktuelle Systemsprache finden kann. Andernfalls verwendet RapaGUI die Zeichenfolge bei Index 0 in der Katalogdatei anstelle des Standardtexts "Click me".

Für lokalabhängige Zeichenketten, die in Tag-Attributen übergeben werden, müssen Sie die Direktive `@i18n:` verwenden. Hängen Sie einfach dieses Suffix an die Zeichenketten an, gefolgt von einem numerischen Wert oder einer Hollywood-Konstante und Hollywood verwendet stattdessen die angegebene Katalogzeichenketten, wenn ein Katalog für die Systemsprache des Benutzers verfügbar ist. Zum Beispiel:

```
<window title="My application@i18n:1">...</window>
```

In diesem Beispiel verwendet RapaGUI den Fenstertitel "My application" nur, wenn die Systemsprache des Benutzers Englisch ist oder wenn RapaGUI keinen Katalog für die aktuelle Systemsprache finden kann. Andernfalls verwendet RapaGUI die Zeichenkette an Index 1 in der Katalogdatei anstelle des Titels "My application".

Wie bereits erwähnt, können Sie auch Hollywood-Konstanten anstelle von fest kodierten numerischen Werten verwenden. Die Verwendung von Hollywood-Konstanten ist möglicherweise bequemer, um Ihr XML zu verwalten, da Sie damit problemlos Einträge hinzufügen und entfernen können. Mit Hollywood-Konstanten sieht das XML von oben so aus:

```
<button i18n="#CAT_BUTTON">Click me</button>
<window title="My application@i18n:#CAT_TITLE">...</window>
```

Beachten Sie, dass alle diese Internationalisierungsfunktionen nur für die XML-Datei gelten. Alle Änderungen, die zur Laufzeit vorgenommen werden, müssen manuell bearbeitet werden, daher müssen Sie Hollywoods Befehl `GetCatalogString()` verwenden, um die richtige Katalogzeichenkette zu erhalten, wenn Sie zur Laufzeit lokalabhängige Änderungen an der GUI vornehmen, z.B. durch Aufrufen von Befehlen wie `moai.Set()`.

3.18 Zeichenkodierung

RapaGUI unterstützt zwei Zeichenkodierungen in den XML-Dateien, die zur Beschreibung des GUI-Layouts verwendet werden: `iso-8859-1` und `utf-8`. Wenn Sie die Codierung `utf-8` unter AmigaOS und kompatiblen Betriebssystemen verwenden, benötigt RapaGUI die `codesets.library`, um Zeichenkonvertierungen von UTF-8 in den Standardzeichensatz des Systems durchzuführen.

Amiga-Anwender beachten bitte, dass MUI UTF-8 nicht unterstützt. MUI verwendet immer den Standardzeichensatz des Systems. Wenn Sie also UTF-8-Codierung in Ihren XML-Dateien verwenden, wird RapaGUI versuchen, diese Zeichenfolgen dem Standardzeichensatz des Systems zuzuordnen. Das wird nicht immer gelingen. Wenn beispielsweise der Standardzeichensatz des Systems ISO-8859-1 ist und die UTF-8 XML-Datei einige osteuropäische Zeichen verwendet, die in ISO-8859-1 nicht vorhanden sind, werden sie nicht korrekt dargestellt. Sie werden nur dann korrekt angezeigt, wenn sie auch im Standardzeichensatz des Systems enthalten sind.

3.19 Verbindung zu Hollywood

Ein mächtige Fähigkeit von RapaGUI ist die Möglichkeit, komplette Hollywood-Displays mit der Hollywood-Klasse in Ihre GUIs einzubetten. Immer wenn Sie etwas auf ein Hollywood-Display zeichnen, das mit der Hollywood-Klasse verbunden ist, wird es automatisch auch auf Ihrem GUI-Widget gezeichnet. Sie können sogar das Hollywood-Display ausblenden, so dass der Benutzer nicht einmal merkt, dass Hollywood im Hintergrund läuft. Darüber hinaus werden alle Mausklicks und Tastenanschläge, die innerhalb der Hollywood-Klasse stattfinden, als normale Hollywood-Ereignisse an das entsprechende Hollywood-Display weitergeleitet. So können Sie mit der Hollywood-Klasse fast alle leistungsstarken Funktionen von Hollywood auch in einem GUI-Widget nutzen.

Schauen wir uns ein Beispiel an. Der folgende Code verwendet die Hollywood-Klasse zum Einbetten von einer spielenden Animation der Größe 320x200 in einem GUI-Widget. Dafür

wird die Größe des Hollywood-Displays auf 320x200 geändert und dann in die GUI als ein MOAI-Objekt vom Typ Hollywood-Klasse eingebettet. Hier zuerst die Definition der XML-GUI:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<application>
  <window title="Hollywood bridge">
    <vgroup>
      <hgroup>
        <rectangle/>
        <hollywood display="1"/>
        <rectangle/>
      </hgroup>
      <hgroup>
        <button id="play">Play</button>
        <button id="stop">Stop</button>
      </hgroup>
    </vgroup>
  </window>
</application>
```

Beachten Sie, dass hier MOAI-Objekte der Rectangle-Klasse (Rechteck) verwendet wurde, um das nicht größenveränderliche Hollywood-Objekt aufzufüllen. Dies ist notwendig, damit die GUI in der Größe veränderbar bleibt. Hier ist der Code, der Ihnen zeigt, wie Sie Ihr GUI-Widget mit Hollywood verbinden können:

```
@REQUIRE "RapaGUI"
@ANIM 1, "amy_walks.anim"
@DISPLAY {Width = 320, Height = 200}

Function p_AnimFunc()
  Local numframes = GetAttribute(#ANIM, 1, #ATTRNUMFRAMES)
  curframe = Wrap(curframe + 1, 1, numframes + 1)
  DisplayAnimFrame(1, 0, 0, curframe)
EndFunction

Function p_EventFunc(msg)
  Switch msg.Class
  Case "Button":
    Switch msg.Attribute
    Case "Pressed":
      Switch msg.ID
      Case "play":
        SetInterval(1, p_AnimFunc, 50)
      Case "stop":
        ClearInterval(1)
      EndSwitch
    EndSwitch
  EndSwitch
```

```

EndFunction

InstallEventHandler({RapaGUI = p_EventFunc})
moai.CreateApp(FileToString("GUI.xml"))

Repeat
    WaitEvent
Forever

```

Mit etwas mehr Arbeit können wir die Animation auch mit der Maus bewegen. Der Benutzer kann dann in das Hollywood-Objekt klicken und die Animation mit der Maus umherziehen. Hier ist der Code dafür:

```

@REQUIRE "RapaGUI"
@ANIM 1, "amy_walks.anim"
@DISPLAY {Width = 320, Height = 200}

Function p_AnimFunc()
    Local numframes = GetAttribute(#ANIM, 1, #ATTRNUMFRAMES)
    curframe = Wrap(curframe + 1, 1, numframes + 1)
    SelectBrush(1)
    Cls
    DisplayAnimFrame(1, offx, offy, curframe)
    EndSelect
    DisplayBrush(1, 0, 0)
EndFunction

Function p_MouseFunc()
    If IsLeftMouse() = True
        Local mx, my = MouseX(), MouseY()
        If (grabx = -1) And (graby = -1) Then
            grabx, graby = mx - offx, my - offy
            offx = mx - grabx
            offy = my - graby
        Else
            grabx, graby = -1, -1
        EndIf
    EndFunction

Function p_EventFunc(msg)
    Switch msg.Class
    Case "Button":
        Switch msg.Attribute
        Case "Pressed":
            Switch msg.ID
            Case "play":
                SetInterval(1, p_AnimFunc, 50)
                SetInterval(2, p_MouseFunc, 20)

```

```

        Case "stop":
            ClearInterval(1)
            ClearInterval(2)
        EndSwitch
    EndSwitch
EndSwitch
EndFunction

CreateBrush(1, 320, 200)

InstallEventHandler({RapaGUI = p_EventFunc})
moai.CreateApp(FileToString("GUI.xml"))

Repeat
    WaitEvent
Forever

```

Schließlich ist es auch möglich, das Hollywood-Objekt in der Größe zu verändern, indem man die Attribute `Area.FixWidth` und `Area.FixHeight` auf `False` setzt. Immer wenn der Benutzer die Fenstergröße ändert, erhält Ihr Hollywood-Display ein `SizeWindow`-Ereignis, das Sie mit dem Hollywood-Befehl `InstallEventHandler()` überwachen können. Wenn wir ein in der Größe veränderbares Hollywood-Objekt verwenden, können wir die beiden Objekte `<rectangle>` entfernen, die ausschließlich als Füllraum verwendet werden. Der XML-Code sieht dann so aus:

```

<?xml version="1.0" encoding="iso-8859-1"?>
<application>
    <window title="Hollywood bridge">
        <vgroup>
            <hollywood display="1" fixwidth="false" fixheight="false"/>
            <hgroup>
                <button id="play">Play</button>
                <button id="stop">Stop</button>
            </hgroup>
        </vgroup>
    </window>
</application>

```

Unser Code ist so ziemlich derselbe wie vorher, mit der Ausnahme, dass wir jetzt das Ereignis `SizeWindow` überwachen müssen, um auf die Größenänderungen der GUI zu reagieren. Hier ist der angepasste Code von oben:

```

@REQUIRE "RapaGUI"
@ANIM 1, "amy_walks.anim"
@DISPLAY {Width = 320, Height = 200}

Function p_AnimFunc()
    Local numframes = GetAttribute(#ANIM, 1, #ATTRNUMFRAMES)
    curframe = Wrap(curframe + 1, 1, numframes + 1)
    SelectBrush(1)

```



```

Cls
DisplayAnimFrame(1, offx, offy, curframe,
  {Width = swidth, Height = sheight})
EndSelect
DisplayBrush(1, 0, 0)
EndFunction

Function p_MouseFunc()
  If IsLeftMouse() = True
    Local mx, my = MouseX(), MouseY()
    If (grabx = -1) And (graby = -1) Then
      grabx, graby = mx - offx, my - offy
      offx = mx - grabx
      offy = my - graby
    Else
      grabx, graby = -1, -1
    EndIf
  EndFunction

Function p_EventFunc(msg)
  If msg.Action = "SizeWindow"
    swidth = msg.Width
    sheight = msg.Height
    CreateBrush(1, swidth, sheight)
    Return
  EndIf

  Switch msg.Class
  Case "Button":
    Switch msg.Attribute
    Case "Pressed":
      Switch msg.ID
      Case "play":
        SetInterval(1, p_AnimFunc, 50)
        SetInterval(2, p_MouseFunc, 20)
      Case "stop":
        ClearInterval(1)
        ClearInterval(2)
      EndSwitch
    EndSwitch
  EndSwitch
EndFunction

swidth, sheight = 320, 200
CreateBrush(1, swidth, sheight)

InstallEventHandler({RapaGUI = p_EventFunc, SizeWindow = p_EventFunc})

```

```

moai.CreateApp(FileToString("GUI.xml"))

Repeat
  WaitEvent
Forever

```

Bei diesem Beispiel können Sie sehen, dass die Hollywood-Klasse wirklich mächtig ist und dazu verwendet werden kann, viele innovative GUI-Ideen zu verwirklichen, die nur durch Ihre Kreativität begrenzt sind!

3.20 Bilder-Cache

Viele MOAI-Klassen ermöglichen es Ihnen, Symbole mit den von ihnen erstellten Widgets zu verwenden. Z.B. können Sie ein Symbol zu einem Schaltflächen-Widget hinzufügen, indem Sie einfach das Attribut `Button.Icon` verwenden. Alle MOAI-Klassen, die mit Symbolen umgehen können, erfordern, dass Sie entweder einen Hollywood-Pinsel oder ein Hollywood-Symbol übergeben, das als Symbol für die Klasse verwendet werden soll.

Wichtig ist zu wissen, dass RapaGUI diese Bilder aus Performance- und Wirtschaftlichkeitsgründen zwischenspeichert. Stellen Sie sich eine Listenansicht mit Tausenden von Zeilen und einem Symbol in jeder Zeile vor. Es wäre ein absoluter Performancekiller, wenn RapaGUI diese Icons für jede einzelne Zeile von einem Hollywood-Pinsel/Icon in ein RapaGUI-Bild umwandeln müsste. Deshalb werden die Bilder zwischengespeichert.

Schauen Sie sich dieses Beispiel an:

```
<button id="ok" icon="1">OK</button>
```

In der obigen Definition steht Hollywood-Pinsel Nummer 1 neben dem Text "OK" für die Schaltfläche. RapaGUI wird nun die Bilddaten des Hollywood-Pinsels Nummer 1 intern zwischenspeichern und jedes Mal, wenn es wieder einen Verweis auf den Hollywood-Pinsel Nummer 1 gibt, wird einfach die Kopie aus seinem Cache genommen, anstatt den Originalpinsel. Das bedeutet, dass alle Änderungen, die Sie an Hollywood-Pinsel Nummer 1 nach der obigen Definition vornehmen, aufgrund des Bilder-Caches keinerlei Auswirkungen auf RapaGUI haben werden! Bitte denken Sie daran. Sie müssen entweder einen Pinsel-Identifikator verwenden, der sich noch nicht im Bildcache von RapaGUI befindet, oder Sie müssen den Pinsel zuerst mit dem Befehl `moai.FreeImage()` aus dem Bildcache von RapaGUI entfernen. Für Listview- und Treeview-Widgets (Listenansichts- und Baumansichts-Widgets) ist es auch möglich, stattdessen `moai.UpdateImage()` zu verwenden.

Es gibt nur eine Ausnahme von der Regel: Widgets, die von der Image-Klasse (Bild) abgeleitet sind, speichern keine Bilddaten, obwohl sie auch mit Hollywood-Pinseln/Icons arbeiten. Der Grund dafür ist, dass Bild-Widgets oft ziemlich große Bilder anzeigen, die auch häufig animiert oder geändert werden können. Aus diesem Grund ist es nicht sinnvoll, Bilddaten für Widgets der Image-Klasse zu cachen. In anderen Widgets verwendete Bilder, z.B. Schaltflächen, Symbolleisten, Schaltflächen, Listen-, Baum-, Seitenansichten sind normalerweise sehr klein und es gibt im normalen Fall nur eine feste Anzahl von Symbolen. Obendrein werden sie vielleicht sehr oft benötigt. Man denke nur an eine Listenansicht mit Tausenden von Zeilen und einem Symbol in jeder dieser Zeilen. Deshalb ist hier wirklich ein Bilder-Cache notwendig, um eine gute Performance zu garantieren.

3.21 Plattformabhängige Fähigkeiten

Generell versucht RapaGUI so wenig plattformabhängige Besonderheiten wie möglich einzubauen, damit die gleiche Anwendung auf einer Vielzahl von Plattformen ohne Anpassungen laufen kann. Es gibt jedoch sehr wenige Attribute und Methoden, die nicht auf allen Plattformen verfügbar sind, aber dennoch enthalten sind. Dies weil sie als so wichtig erachtet wurden, dass sie aufgenommen wurden, obwohl sie in gewissem Widerspruch zu RapaGUIs plattformblindem GUI-Toolkit-Ansatz stehen. Zum Beispiel ist das Attribut `Window.HideFromTaskBar` nur unter Windows und GTK+ verfügbar, da es unter AmigaOS und macOS keine Taskleiste gibt. Ein weiteres Beispiel ist `Application.OpenConfigWindow`, das nur unter AmigaOS und kompatiblen Betriebssystemen verfügbar ist.

Plattformspezifische Fähigkeiten werden immer als solche dokumentiert. Sie müssen nur in der Dokumentation nachsehen, ob eine bestimmte Fähigkeit plattformspezifisch ist. Um plattformspezifischen Code zu minimieren, können alle plattformabhängigen Fähigkeiten auch auf allen von RapaGUI unterstützten Plattformen spezifiziert werden. RapaGUI ignoriert dann einfach die Attribute und Methoden, anstatt eine Fehlermeldung zu melden. So könnte man `Window.HideFromTaskBar` auch auf AmigaOS und macOS setzen. RapaGUI meldet keine Fehlermeldung. Der Aufruf wird einfach ignoriert.

3.22 Kompatibilität mit MUI Royale

RapaGUI begann als eine Abzweigung des populären MUI Royale Plugins für Hollywood. Leute, die mit MUI Royale vertraut sind, werden ohne Zweifel sehen, dass RapaGUI das gleiche Design benutzt, so dass die Portierung von Programmen von MUI Royale nach RapaGUI nicht besonders schwierig ist. Obwohl RapaGUI aufgrund der AmigaOS-basierten Terminologie viele semantische Änderungen erfahren hat, sind die Designunterschiede tatsächlich sehr gering. Wenn Sie also Programme von MUI Royale nach RapaGUI portieren, werden Sie sich nur mit Namensänderungen beschäftigen. Die meisten Leute werden wahrscheinlich nicht einmal auf einen der wenigen Design-Unterschiede zwischen MUI Royale und RapaGUI treffen. Dennoch ist hier eine nicht vollständige Liste von Designunterschieden zwischen MUI Royale und RapaGUI, um Ihre Portierungsbemühungen zu vereinfachen:

- MUI Royale verlangte von Ihnen, dass Sie `Window.CloseRequest` für alle Ihre Fenster verwenden, sonst würden sie offen bleiben. RapaGUI schließt automatisch Fenster, die keine Überwachung für `Window.CloseRequest` installiert haben.
- MUI Royale verlangte auch, dass Sie die Ereignisse "HideWindow" und "ShowWindow" überwachen, um die Minimierung von Fenstern zu ermöglichen. RapaGUI übernimmt dies nun automatisch.
- RapaGUI richtet automatisch für Schaltflächen, Symbolleistenschaltflächen und Menüeinträge Ereignisüberwachung ein. MUI Royale verlangte von Ihnen, dass Sie diese explizit anfordern, aber dies führte zu unnötig wortreichen XML-Dateien, in denen Sie so etwas wie `notify="pressed"` für jede einzelne Schaltfläche einfügen mussten, weil Sie unter normalen Umständen über jeden Schaltflächen-Klick benachrichtigt werden möchten. RapaGUI ruft automatisch Ihre Callback-Funktion für alle Schaltflächen- und Symbolleisten-Schaltflächen-Klicks (Toolbarbutton) und Menüeintrag-Ereignisse auf, um den XML-Code besser lesbar zu halten.

- RapaGUI blendet automatisch alle Hollywood-Displays beim Start aus, während dies bei MUI Royale nicht der Fall ist. Bei MUI Royale müssen Sie die Standard-Displays manuell als verborgen definieren.
- MUI Royale ermöglicht es Ihnen, fast überall Textformatierungs-codes zu verwenden. Da es sich hierbei um eine hochgradig MUI-zentrierte Fähigkeit handelt, ist es unmöglich, diese plattformunabhängig zu implementieren. RapaGUI unterstützt Textformatierung nur in den Klassen `Textview` (Textanzeige) sowie `Texteditor` und nur dann, wenn `Textview.Styled` sowie `Texteditor.Styled` auf `True` gesetzt wurden.
- MUI Royale erlaubt es Ihnen auch, Symbole in fast allen Widgets über Textformatierungs-codes einzubinden. Dies ist auch nicht plattformübergreifend realisierbar. Dennoch hat RapaGUI Symbol-Unterstützung in vielen Klassen, aber die Art und Weise, wie dies implementiert wird, ist immer klassenabhängig. Mit RapaGUI können Sie Symbole mit folgenden Klassen verwenden: `Button`-Klasse, `Listview`-Klasse, `Pageview`-Klasse und `Treeview`-Klasse (Schaltflächen, Listenansicht, Seitenansicht und Bauman-sicht). Schauen Sie sich die Dokumentation dieser Klassen an, um mehr darüber zu erfahren, wie man Symbole mit ihnen verwendet.
- MUI Royale verlangt, dass Sie ein von der Klasse `<virtgroup>` abgeleitetes Objekt als Element für `<scrollgroup>`-Objekte verwenden. Mit RapaGUI können Sie ganz normale Gruppen verwenden.
- MUI Royale sendet alle Kontextmenü-Ereignisse immer über das Attribut `Area.ContextMenuTrigger`. RapaGUI sendet Kontextmenü-Ereignisse einfach als normale Menüpunkt-Ereignisse. Aber es enthält ein Feld `Parent` in der Ereignismeldung, um Sie über das Widget zu informieren, dessen Kontextmenü das Ereignis ausgelöst hat.
- RapaGUI's `Checkbox`-Klasse erzeugt ein Auswahlkästchen mit einer Textbeschriftung (Label). MUI Royale's `Checkmark`-Klasse erstellt einfach ein Auswahlkästchen-Bild und Sie müssen die Beschriftung (Label) selbst erstellen.
- RapaGUI hat keine Unterstützung für das Attribut `Listview.InsertPosition`, aber die Methode `Listview.Insert` gibt einfach die Position des neu eingefügten Eintrags zurück.
- In MUI Royale wurden `Texteditor`-Bereiche durch Blockkoordinaten angegeben, die aus vier verschiedenen Werten (`x1`, `y1`, `x2`, `y2`) bestehen. RapaGUI verwendet jetzt nur noch Start- und Stoppkoordinaten, so dass alle Methoden und Attribute, die sich mit Textbereichen befassen, jetzt nur noch zwei Argumente benötigen/zurückgeben.
- RapaGUI's `Texteditor`-Klasse, `Textview`-Klasse (Textanzeige) und `Textentry`-Klasse (Texteingabe) verwenden den Inhalt zwischen den öffnenden und schließenden XML-Tags als initialen Widget-Inhalt, während Sie mit MUI Royale immer ein Attribut verwenden müssen, um den ursprünglichen Inhalt festzulegen.
- RapaGUI verlangt, dass Sie ein übergeordnetes Objekt angeben, wenn Sie Objekte dynamisch mit `moai.CreateObject()` erstellen. Das war bei MUI Royale nicht nötig.
- Da RapaGUI's `Treeview`-Klasse (Bauman-sicht) mehrere Spalten unterstützt, müssen Sie bei der Definition Ihrer Bauman-sicht in XML mindestens einen `<column>` erstellen. Dies war bei MUI Royale nicht nötig, da die `Treeview`-Klasse nur einzelne Spaltenbäume unterstützt.

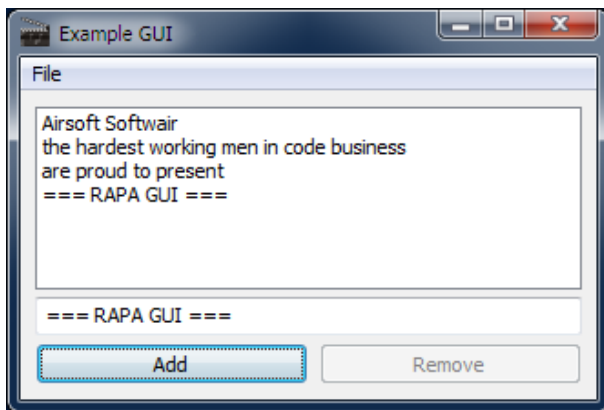
- MUI Royale hat kein Konzept mit Dialogen, da AmigaOS auch keines hat. Modale Dialoge wurden nur mit normalen Fenstern emuliert und alle anderen Fenster wurden in den Ruhezustand versetzt. RapaGUI führt jetzt echte Dialoge ein. Während Sie bei MUI Royale normalerweise alle diese Pseudo-Dialoge beim Start erstellen, wird dies mit RapaGUI nicht mehr empfohlen. Da Fenster auf manchen Betriebssystemen eine recht begrenzte Ressource ist, sollten Sie Dialoge nur dann erstellen, wenn Sie sie benötigen und sie nachher wieder sofort löschen. Siehe [Abschnitt 16.1 \[Dialog-Klasse\]](#), Seite 99, für Details. Wenn Sie nur AmigaOS verwenden möchten, dann können Sie natürlich alle Ihre Fenster mit nur einem einzigen Aufruf von `moai.CreateApp()` erstellen, aber wenn Sie andere Plattformen auch berücksichtigen möchten, sollten Sie die empfohlene Programmierrichtlinie befolgen und Dialoge nur dann erstellen, wenn Sie sie benötigen und diese sofort nach dem Schliessen wieder löschen.
- Objekte der Radio-Klasse werden immer in RapaGUI gerahmt. In MUI Royale werden sie standardmäßig nicht gerahmt.
- Wenn Sie einen etwas komplexere Tastaturkürzel-Zeichenfolge wie `CTRL+V`, `Alt+X` oder `F5` mit `MenuItem.CommandString` verwenden, überwacht diese MUI Royale nicht automatisch. Dies muss manuell durchgeführt werden. Nur Standardabkürzungen in Form einer einzelnen alphabetischen Taste, kombiniert mit der Taste `CMD`, werden von MUI Royale automatisch überwacht. RapaGUI verarbeitet jedoch auch komplexe Verknüpfungen automatisch.
- Wenn `Listview.Active` abgerufen wird und kein Eintrag aktiv ist, gibt MUI Royale die spezielle Zeichenfolge `Off` zurück, während RapaGUI `-1` zurückgibt.

Natürlich gibt es auch viele Attribute und Methoden, die in MUI Royale vorhanden sind, aber in RapaGUI fehlen. Der Grund dafür ist natürlich, dass RapaGUI aufgrund seines plattformübergreifenden Designs den kleinsten gemeinsamen Nenner zwischen Windows, Linux, macOS und AmigaOS darstellt. Deshalb konnten nicht alle Fähigkeiten von MUI Royale auf RapaGUI übertragen werden.

4 Tutorial

4.1 Tutorial

Willkommen zum RapaGUI-Tutorial! Dieses kleine Schritt-für-Schritt-Dokument führt Sie in wenigen Schritten durch den Prozess der Erstellung Ihrer ersten GUI mit RapaGUI. Wir werden eine kleines GUI-Programm erstellen, welches aus einer Listenansicht, einem Texteingabe-Widget und zwei Schaltflächen besteht (Listview, Textentry, Button). Die beiden Schaltflächen sollten so programmiert werden, dass sie das Hinzufügen und Entfernen von Einträgen aus der Listenansicht ermöglichen. Die Daten, die in die Listenansicht eingefügt werden sollen, sind dem Texteingabe-Widget zu entnehmen. So sieht dieses kleine Programm unter Windows aus:



Beginnen wir mit den Grundlagen: In RapaGUI werden GUIs mit Hilfe von XML-Dateien erstellt, die eine Beschreibung einer Reihe von Fenstern enthalten, die eine Vielzahl von GUI-Widgets beinhalten. Hier ist eine minimale GUI-Beschreibung für eine RapaGUI-GUI, die ein Fenster mit einer Listenansicht, einem Texteingabe-Widget und zwei Schaltflächen im XML-Format enthält:

```
<?xml version="1.0" encoding="iso-8859-1"?>
<application>
  <window title="Example GUI">
    <vgroup>
      <listview>
        <column/>
      </listview>
      <textentry/>
      <hgroup>
        <button id="add">Add</button>
        <button id="rem">Remove</button>
      </hgroup>
    </vgroup>
  </window>
</application>
```

Das Objekt `<application>` ist das Haupt-MOAI-Objekt für jede Anwendung und darf nur einmal pro Anwendung verwendet werden. Alle anderen MOAI-Objekte sind untergeordnete Elemente der Application-Klasse. Das Wurzelement jedes `<window>`-Objekts muss von der Group-Klasse (Gruppen) abgeleitet sein, d.h. es muss entweder `<vgroup>`, `<hgroup>` oder `<colgroup>` sein. Beachten Sie auch, dass pro Fenster nur ein Wurzelement vorhanden sein muss.

Um zu sehen, wie unsere obige XML-Definition als GUI aussieht, müssen wir sie in einer Datei namens `GUI.xml` speichern und dann den folgenden Code verwenden, damit RapaGUI sie in eine native GUI für das Betriebssystem konvertiert, auf dem Hollywood läuft:

```
@REQUIRE "RapaGUI"

moai.CreateApp(FileToString("GUI.xml"))

Repeat
    WaitEvent
Forever
```

Als Nächstes sollten Sie einige Informationen über Ihr Programm hinzufügen, indem Sie die Präprozessor-Anweisungen `@APPAUTHOR`, `@APPCOPYRIGHT`, `@APPDESCRIPTION`, `@APTITLE` und `@APPVERSION` verwenden. RapaGUI benötigt diese Informationen für verschiedene Zwecke, z.B. werden die Informationen, die in `@APTITLE` übergeben werden, auch vom MUI-Einstellungsfenster auf AmigaOS und kompatiblen Systemen verwendet. Hier ist eine Beispieldefinition dieser Präprozessor-Anweisungen:

```
@APTITLE "Tutorial"
@APPVERSION "$VER: Tutorial 1.0 (29.12.15)"
@APPCOPYRIGHT "Copyright ©2015, Andreas Falkenhahn"
@APPAUTHOR "Andreas Falkenhahn"
@APPDESCRIPTION "The tutorial app from the RapaGUI guide"
```

Als nächstes müssen wir eine Callback-Funktion mit Hilfe des Hollywood-Befehls `InstallEventHandler()` installieren, da unser Hollywood-Skript jedes Mal informiert werden muss, wenn ein RapaGUI-Ereignis eintritt. Daher müssen wir unseren Code wie folgt modifizieren:

```
@REQUIRE "RapaGUI"

Function p_EventFunc(msg)
    ; contents follow below
EndFunction

InstallEventHandler({RapaGUI = p_EventFunc})
moai.CreateApp(FileToString("GUI.xml"))

Repeat
    WaitEvent
Forever
```

Was wir hier getan haben, ist die Installation der Funktion `p_EventFunc` als Callback-Funktion, die immer dann ausgeführt wird, wenn ein RapaGUI-Ereignis eintritt. Wenn wir

ein solches Ereignis erhalten, müssen wir dann prüfen, welche MOAI-Klasse und welches Attribut es ausgelöst hat. Dies geschieht, indem man die Felder `msg.Class` und `msg.Attribute` der Ereignismeldung überprüft, die unsere Callback-Funktion als ersten Parameter erhält.

Als nächstes wollen wir die Funktionalität hinzufügen, dass der Text im Texteingabe-Widget immer dann als letzter Eintrag in die Listenansicht eingefügt wird, wenn der Benutzer auf die Schaltfläche "Add" (Hinzufügen) klickt. Dazu müssen wir zunächst einen Weg finden, unsere Widgets aus dem Hollywood-Skript zu identifizieren. Dies geschieht durch die Angabe von IDs in der XML-Definition. IDs sind einfache Textzeichenfolgen, die für die Kommunikation mit MOAI-Objekten aus Hollywood-Skripten verwendet werden. Fügen wir also jetzt einige IDs für alle Widgets hinzu, mit denen wir kommunizieren wollen. Wir müssen unsere XML-Definition so modifizieren:

```

...
    <listview id="mylistview">
        <column/>
    </listview>
    <textentry id="mystring"/>
    <hgroup>
        <button id="mybt1">Add</button>
        <button id="mybt2">Remove</button>
    </hgroup>
...

```

Nun, da wir dies erledigt haben, können wir etwas Code zu unserer Callback-Funktion hinzufügen, die den Inhalt des Texteintrag-Widgets erfasst und am Ende der Liste in unserem Listview-Objekt (Listenansicht) hinzufügt. Dies geschieht, indem man zuerst `moai.Get()` für das Attribut `Textentry.Text` aufruft, um den Inhalt des Texteintrag-Widgets zu erhalten und dann die Methode `Listview.Insert` ausführt. Fügen Sie mit `moai.DoMethod()` den Eintrag in die Listenansicht ein. Hier ist der Code, der zu diesem Zweck in die Funktion `p_EventFunc` eingefügt werden muss:

```

Switch msg.Class
...
Case "Button":
    Switch msg.Attribute
    Case "Pressed":
        Switch msg.ID
        Case "mybt1": ; "Add" button was pressed
            Local s$ = moai.Get("mystring", "text")
            moai.DoMethod("mylistview", "insert", "bottom", s$)
        EndSwitch
    EndSwitch
EndSwitch

```

Als nächstes wollen wir die Funktionalität unserer Schaltfläche "Remove" (Entfernen) implementieren. Immer wenn diese Schaltfläche gedrückt wird, soll der aktive Eintrag aus der Listenansicht entfernt werden. Wir können dies tun, indem wir die Methode `Listview.Remove` in der Listenansicht ausführen. Daher müssen wir unseren Code so modifizieren:

```

Switch msg.ID

```

```

...
Case "mybt2":      ; "Remove" button was pressed
    moai.DoMethod("mylistview", "remove", "active")
EndSwitch

```

Nun wollen wir, dass der aktive Eintrag der Listenansicht automatisch im Texteingabe-Element angezeigt wird. Dazu müssen wir eine Benachrichtigung für das Attribut `Listview.Active` einrichten, die immer dann ausgelöst wird, wenn sich der aktive Eintrag der Listenansicht ändert. Daher müssen wir unsere XML-Datei so modifizieren:

```

...
    <listview id="mylistview" notify="active">
        <column/>
    </listview>
...

```

In unserer Callback-Funktion können wir diese Funktionalität ganz einfach implementieren, indem wir die Methode `Listview.GetEntry` ausführen und dann den Inhalt des Texteingabe-Widgets mit dem Attribut `Textentry.Text` setzen. Hier ist der Code dafür:

```

Switch msg.Class
...
Case "Listview":
    Switch msg.Attribute
    Case "Active":
        Local s$ = moai.DoMethod("mylistview", "getentry", "active")
        moai.Set("mystring", "text", s$)
    EndSwitch
EndSwitch

```

Wenn Sie diesen Code ausprobieren, werden Sie feststellen, dass das Attribut `Listview.Active` nicht nur dann ausgelöst wird, wenn der Benutzer mit der Maus einen neuen Listeneintrag auswählt, sondern auch dann, wenn Einträge aus der Listenansicht entfernt werden und somit ein neuer Eintrag aktiv wird.

Als nächstes wollen wir die Schaltfläche "Remove" deaktivieren, wenn kein Eintrag in der Listenansicht aktiv ist. Wir können Widgets deaktivieren, indem wir das Attribut `Area.Disabled` auf `True` setzen. Da es zunächst keine Einträge in der Listenansicht gibt, müssen wir `Area.Disabled` bereits beim Start unseres Programms auf `True` setzen. Also müssen Sie diesen Code eingeben:

```

...
moai.CreateApp(FileToString("GUI.xml"))
moai.Set("mybt2", "disabled", True)
...

```

Nun müssen wir einige Änderungen an unserer Callback-Funktion vornehmen. Wann immer wir die Benachrichtigung mit `Listview.Active` erhalten, müssen wir prüfen, ob sie sich von dem speziellen Wert "Off" unterscheidet. Wenn dies der Fall ist, aktivieren wir die Schaltfläche "Remove". Der spezielle Wert "Off" (-1) wird von `Listview.Active` zurückgegeben, wenn in der Listenansicht kein aktiver Eintrag vorhanden ist. Wir müssen unseren Code folgendermaßen anpassen:

```

Switch msg.Class

```

```

...
Case "Listview":
    Switch msg.Attribute
    Case "Active":
        Local s$ = moai.DoMethod("mylistview", "getentry", "active")
        moai.Set("mystring", "text", s$)
        moai.Set("mybt2", "disabled", IIf(msg.triggervalue = -1,
            True, False))
    EndSwitch
EndSwitch

```

Wir verwenden hier das Feld `msg.TriggerValue`. Dieser enthält immer den aktuellen Wert des Attributs, welches das Ereignis ausgelöst hat, d.h. in unserem Fall den aktuellen Wert des Attributs `Listview.Active`. Wir könnten auch `moai.Get()` zuerst manuell auf `Listview.Active` aufrufen, aber das ist nicht wirklich notwendig, da wir einfach `msg.TriggerValue` verwenden können.

Als Letztes fügen wir ein Menü zu unserer GUI hinzu. Alle RapaGUI-Programme sollten über einen Menüpunkt verfügen, der den Benutzer darüber informiert, dass dieses Programm mit RapaGUI erstellt wurde. Sie können diesen Dialog öffnen, indem Sie die Methode `Application.AboutRapaGUI` ausführen. Um dieses Menü unserem Programm hinzuzufügen, verwenden wir die `Menubar`-Klasse (Menüleisten). Hier ist der XML-Code, den Sie vor Ihrer Fensterdefinition hinzufügen müssen:

```

...
<application>
  <menubar id="mymenubar">
    <menu title="File">
      <item id="menabout">About...</item>
      <item id="menaboutrapagui">About RapaGUI...</item>
      <item/>
      <item id="menquit">Quit</item>
    </menu>
  </menubar>
  ...
</application>

```

Nachdem wir unser Menüleisten-Objekt mit Hilfe des obigen XML-Codes erstellt haben, müssen wir diese Menüleiste an unserem Fenster anhängen. Dies geschieht durch das Setzen des Attributs `Window.Menubar` auf unser Menüleisten-Objekt. Hier ist der XML-Code dafür:

```
<window title="Example GUI" menubar="mymenubar">
```

Nun müssen wir unserer Callback-Funktion Code hinzufügen, der die entsprechende Aktion ausführt, wenn ein Menüpunkt ausgewählt wird. Bevor wir das tun können, müssen wir jedoch unserem Applications-Objekt eine ID zuweisen, damit wir `moai.DoMethod()` verwenden können. Hier sehen Sie, wie der XML-Code angepasst werden muss:

```
<application id="app">
```

Jetzt können wir den Code für unsere Callback-Funktion schreiben, welche die Menüeinträge berücksichtigt:

```
Switch msg.Class
```

```

...
Case "MenuItem":
    Switch msg.Attribute
    Case "Selected":
        Switch msg.id
        Case "menabout":
            moai.Request("Test", "Test program\n" ..
                "© 2015 by Andreas Falkenhahn", "OK")
        Case "menaboutrapagui":
            moai.DoMethod("app", "aboutrapagui")
        Case "menquit":
            End
        EndSwitch
    EndSwitch
EndSwitch

```

Einige letzte Änderungen an unserem Programm könnten darin bestehen, Online-Hilfe zu unseren Widgets hinzuzufügen, indem man die Attribute `Area.Tooltip` und `MenuItem.Help` verwendet, eine Statusleiste und Tastaturkürzel für die Menüeinträge mit dem Unterstrichzeichen hinzufügt. Diese Änderungen werden dem Leser als Übung überlassen.

So sieht unser endgültiges Programm aus. Zuerst die XML-Datei:

```

<?xml version="1.0" encoding="iso-8859-1"?>
<application id="app">
  <menubar id="mymenubar">
    <menu title="File">
      <item id="menabout">About...</item>
      <item id="menaboutrapagui">
        About RapaGUI...</item>
      <item/>
      <item id="menquit">Quit</item>
    </menu>
  </menubar>
  <window title="Example GUI" menubar="mymenubar">
    <vgroup>
      <listview id="mylistview" notify="active">
        <column/>
      </listview>
      <textentry id="mystring"/>
      <hgroup>
        <button id="mybt1">Add</button>
        <button id="mybt2">Remove</button>
      </hgroup>
    </vgroup>
  </window>
</application>

```

Und hier ist der Code für die Programmlogik:

```

@REQUIRE "RapaGUI"

@APPTITLE "Tutorial"
@APPVERSION "$VER: Tutorial 1.0 (29.12.15)"
@APPCOPYRIGHT "Copyright ©2015, Andreas Falkenhahn"
@APPAUTHOR "Andreas Falkenhahn"
@APPDESCRIPTION "The tutorial app from the RapaGUI guide"

Function p_EventFunc(msg)

    Switch msg.Class
    Case "Button":
        Switch msg.Attribute
        Case "Pressed":
            Switch msg.ID
            Case "mybt1": ; "Add" button was pressed
                Local s$ = moai.Get("mystring", "text")
                moai.DoMethod("mylistview", "insert", "bottom", s$)
            Case "mybt2": ; "Remove" button was pressed
                moai.DoMethod("mylistview", "remove", "active")
            EndSwitch
        EndSwitch

    Case "Listview":
        Switch msg.Attribute
        Case "Active":
            Local s$ = moai.DoMethod("mylistview", "getentry", "active")
            moai.Set("mystring", "text", s$)
            moai.Set("mybt2", "disabled", IIf(msg.triggervalue = -1,
                True, False))
        EndSwitch

    Case "MenuItem":
        Switch msg.Attribute
        Case "Selected":
            Switch msg.id
            Case "menabout":
                moai.Request("Test", "Test program\n" ..
                    "© 2015 by Andreas Falkenhahn", "OK")
            Case "menaboutrapagui":
                moai.DoMethod("app", "aboutrapagui")
            Case "menquit":
                End
            EndSwitch
        EndSwitch
    EndSwitch
EndSwitch

```

```
EndFunction

InstallEventHandler({RapaGUI = p_EventFunc})
moai.CreateApp(FileToString("GUI.xml"))
moai.Set("mybt2", "disabled", True)

Repeat
    WaitEvent
Forever
```

Das ist es! Jetzt sollten Sie in der Lage sein, fantastische neue GUI-Programme mit Hollywood und RapaGUI zu erstellen. Vielen Dank für das Lesen dieses Tutorials und genießen Sie die Leistungsfähigkeit moderner GUI-Programmierung mit Hollywood und RapaGUI in Ihren Händen!

5 Beispiele

5.1 Beispiele

RapaGUI enthält eine Reihe von Beispielen, die bestimmte Fähigkeiten demonstrieren und Ihnen einen schnellen Einstieg ermöglichen sollen. Hier ist eine Liste von Beispielen, die mit RapaGUI mitgeliefert werden:

Accelerators

Zeigt, wie man eine Tastaturkürzel-Tabelle für die erweiterte Tastatursteuerung der GUI einrichtet. Siehe [Abschnitt 7.1 \[Accelerator-Klasse\]](#), [Seite 61](#), für Details.

Demo Eine umfangreiche Präsentation der meisten von RapaGUI unterstützten Widgets. Schauen Sie sich dieses Beispiel an, um zu sehen, wie die verschiedenen MOAI-Klassen von RapaGUI als Widgets erscheinen.

Dialogs Zeigt, wie man modale Dialoge mit RapaGUI verwendet. Dieses Beispiel demonstriert sowohl funktionsgesteuerte Dialoge als auch normale modale Dialoge. Siehe [Abschnitt 16.1 \[Dialog-Klasse\]](#), [Seite 99](#), für Details.

DragNDrop

Zeigt, wie man Ziehen und Fallenlassen mit RapaGUI implementiert.

Dynamic1 Demonstriert, wie zur Laufzeit neue Fenster erstellt werden.

Dynamic2 Demonstriert, wie zur Laufzeit neue Schaltflächen erstellt werden.

Dynamic3 Zeigt, wie man zur Laufzeit neue Registerkarten für die Pageview-Klasse (Seitenansicht) erstellt. Siehe [Abschnitt 34.1 \[Pageview-Klasse\]](#), [Seite 199](#), für Details.

Dynamic4 Zeigt, wie man zur Laufzeit Menüs und Menüeinträge erstellt.

HTMLView

Dies ist ein rudimentärer Internet-Browser, der in RapaGUI und Hollywood geschrieben wurde. Er demonstriert die HTMLview-Klasse. (HTMLAnsicht) Siehe [Abschnitt 23.1 \[HTMLview-Klasse\]](#), [Seite 133](#), für Details.

Image Dieses Beispiel zeigt, wie man einen Hollywood-Pinsel mit der Image-Klasse (Bild) in die GUI einbetten kann. Siehe [Abschnitt 25.1 \[Image-Klasse\]](#), [Seite 141](#), für Details.

MultiDisplays

Demonstriert, wie man vier verschiedene Hollywood-Displays in ein einziges RapaGUI-Fenster einbetten kann. Dies zeigt die Magie, die durch die Kombination von Hollywood und RapaGUI möglich ist. Siehe [Abschnitt 20.1 \[Hollywood-Klasse\]](#), [Seite 125](#), für Details.

Pages Zeigt, wie man ein Fenster mit Registergruppen mit dem MOAI-Objekt Pageview-Klasse erstellt. Siehe [Abschnitt 34.1 \[Pageview-Klasse\]](#), [Seite 199](#), für Details.

Scrollbar Demonstriert, wie eine mit der Scrollbar-Klasse erstellte Bildlaufleiste an ein benutzerdefiniertes Widget angehängt wird. Siehe [Abschnitt 42.1 \[Scrollbar-Klasse\]](#), [Seite 223](#), für Details. Alternativ können Sie auch Scrollcanvas-Klasse (Bildlaufleinwand) verwenden, um dasselbe zu erreichen. Siehe unten für ein Beispiel.

Scrollcanvas

Dieses Beispiel demonstriert eine andere Art der Verwendung von Bildlaufleisten und Widgets mit benutzerdefinierten Grafiken in RapaGUI. Diesmal wird die Scrollcanvas-Klasse verwendet, um zu sehen, dass das Gleiche möglich wie mit der Scrollbar-Klasse ist. Siehe [Abschnitt 43.1 \[Scrollcanvas-Klasse\]](#), [Seite 227](#), für Details. Siehe oben für ein Beispiel.

ShowHide Zeigt, wie MOAI-Objekte zur Laufzeit ein- und ausgeblendet werden können, indem das Attribut `Area.Hide` gesetzt wird.

SongPlayer

Dies ist ein Audiodatei-Player, der mit RapaGUI geschrieben wurde.

TextEditor

Ein kleiner Texteditor, der in Hollywood geschrieben wurde. Außerdem wird gezeigt, wie man eine Symbolleiste und eine Statusleiste verwendet.

Treeview Zeigt, wie man in RapaGUI mit der Treeview-Klasse eine mehrspaltige Bauman-sicht erstellt. Siehe [Abschnitt 20.1 \[Hollywood-Klasse\]](#), [Seite 125](#), für Details.

VideoPlayer

Dies ist ein mit RapaGUI geschriebener Video-Player. Er zeigt, wie ein Hollywood-Display mit der Hollywood-Klasse in ein RapaGUI-Fenster eingebettet wird. Siehe [Abschnitt 20.1 \[Hollywood-Klasse\]](#), [Seite 125](#), für Details. Darüber hinaus wird gezeigt, wie Sie Bildschaltflächen, eine Listenansicht mit Bildern und Schieberegler-Objekte verwenden.

6 Befehlsreferenz

6.1 moai.CreateApp

BEZEICHNUNG

moai.CreateApp – erstellt ein Application-Objekt aus einer XML-Quelle

ÜBERSICHT

moai.CreateApp(xml\$)

BESCHREIBUNG

Dieser Befehl erstellt eine GUI aus der XML-Beschreibung, die im Argument `xml$` übergeben wird. Bitte beachten Sie, dass `xml$` eine Zeichenkette sein muss, die die XML-GUI-Definition und nicht einen Dateinamen enthält. Wenn Sie eine XML-GUI-Definition aus einer externen Datei verwenden wollen, müssen Sie diese Datei zuerst in eine Zeichenkette konvertieren, z.B. mit dem Hollywood-Befehl `FileToString()`.

Sobald dieser Befehl die GUI erstellt hat, können Sie auf allen MOAI-Objekten, die Sie in der XML-GUI-Definition beschrieben haben, die Befehle `moai.Set()`, `moai.Get()` und `moai.DoMethod()` anwenden.

Beachten Sie, dass es in einem Programm nur ein GUI-Application-Objekt geben und somit dieser Befehl nur einmal aufgerufen werden kann. Wenn Sie `moai.CreateApp()` ein zweites Mal aufrufen wollen, müssen Sie zuerst die alte GUI mit dem Befehl `moai.FreeApp()` löschen.

Wenn Sie Objekte wie Fenster oder Schaltflächen dynamisch zu Ihrem GUI-Application-Objekt hinzufügen müssen, können Sie dazu den Befehl `moai.CreateObject()` verwenden. Für Dialoge können Sie den Befehl `moai.CreateDialog()` benutzen.

EINGABEN

`xml$` eine Zeichenkette, die eine XML-GUI-Definition enthält

BEISPIEL

```
moai.CreateApp([[
<?xml version="1.0" encoding="iso-8859-1"?>
<application>
  <window title="Test program">
    <vgroup>
      <button id="btn">Hello World!</button>
    </vgroup>
  </window>
</application>
]])
```

```
InstallEventHandler({RapaGUI = Function(msg)
  If msg.attribute = "Pressed" Then DebugPrint("Button pressed!")
  EndFunction})
```

Repeat

```

    WaitEvent
Forever

```

Der obige Code erzeugt eine minimale GUI mit nur einem Fenster und einer einzigen Schaltfläche.

6.2 moai.CreateDialog

BEZEICHNUNG

`moai.CreateDialog` – erstellt ein Dialogobjekt aus einer XML-Quelle

ÜBERSICHT

```
moai.CreateDialog(xml$[, parent$])
```

BESCHREIBUNG

Mit diesem Befehl können Sie dynamisch ein Dialogobjekt aus einer XML-Quelle erzeugen. Das neu erstellte Dialogobjekt wird ebenfalls automatisch an Ihr Programm angehängt, so dass es sofort einsatzbereit ist. Sie sollten auch ein übergeordnetes Fenster für Ihren Dialog mit dem optionalen Argument `parent$` angeben.

`moai.CreateDialog()` ist sehr wichtig, da Sie Dialoge nur dann erstellen sollten, wenn Sie sie brauchen und sie aus dem Speicher löschen, sobald Sie den Dialog beendet haben. Es wird nicht empfohlen, alle Ihre Dialoge beim Start mit `moai.CreateApp()` zu erstellen und sie ständig im Speicher zu behalten. Stattdessen sollten Sie einen Dialog nur dann erstellen, wenn Sie ihn benötigen und ihn dann löschen, sobald Sie ihn beendet haben. Der Grund dafür ist, dass Fenster auf einigen von RapaGUI unterstützten Betriebssystemen eine begrenzte Ressource ist. Zum Beispiel gibt es unter Windows ein Limit von ungefähr 10.000 Fenstern pro Prozess. Das klingt vielleicht ausreichend genug, aber bedenken Sie, dass unter Windows jedes GUI-Widget für das Betriebssystem ein "Fenster" ist. Z.B. jede Beschriftung, jede Schaltfläche, jeder Rahmen, jedes Auswahlkästchen, jede Gruppe usw. ist ein Fenster in Ihrer Anwendung (Application), so dass Sie darauf achten sollten, dass Sie Ihre Dialoge nur mit `moai.CreateDialog()` erstellen und anschließend direkt wieder löschen.

In der Praxis ist es ratsam, für jeden Ihrer Dialoge eine eigene XML-Datei zu erstellen und mit `moai.CreateDialog()` diese XML-Datei zur Laufzeit in einen Dialog zu konvertieren. Sobald der Benutzer ihn schließt, lässt man ihn automatisch durch RapaGUI wieder aus dem Speicher löschen.

Technisch gesehen handelt es sich bei diesem Befehl lediglich um eine Komfortfunktion, die intern `moai.CreateObject()` aufruft und dann das neu erstellte Objekt durch Aufruf von `Application.AddWindow` dem Applications-Objekt hinzufügt. `moai.CreateDialog()` kombiniert diese beiden Schritte einfach in einem.

Siehe [Abschnitt 16.1 \[Dialog-Klasse\]](#), Seite 99, für mehr Informationen über Dialoge.

EINGABEN

`xml$` eine Zeichenkette, welche eine XML-MOAI-Dialogbeschreibung enthält
`parent$` optional: übergeordnetes Fenster für das Dialogobjekt; Details siehe oben

BEISPIEL

```
moai.CreateDialog([[
```

```

<dialog id="dlg" title="Question">
  <vgroup>
    <text>What is your name?</text>
    <textentry/>
    <hgroup>
      <button id="ok">OK</button>
      <button id="cancel">Cancel</button>
    </hgroup>
  </vgroup>
</window>
]]

```

```
moai.DoMethod("dlg", "showmodal")
```

Der obige Code erzeugt einen neuen Dialog und zeigt ihn an.

6.3 moai.CreateObject

BEZEICHNUNG

moai.CreateObject – erstellt ein MOAI-Objekt aus einer XML-Quelle

ÜBERSICHT

```
moai.CreateObject(xml$[, parent$])
```

BESCHREIBUNG

Dieser Befehl kann verwendet werden, um ein MOAI-Objekt dynamisch aus einer XML-Quelle zu erstellen. Wenn `moai.CreateObject()` beendet ist, wird das neu erstellte MOAI-Objekt nicht an ein übergeordnetes Objekt angefügt und befindet sich losgelöst (isoliert) von Ihrem Application-Objekt, das mit `moai.CreateApp()` erstellt wurde. Um diesen losgelösten Zustand aufzuheben, müssen Sie das Objekt zuerst an ein übergeordnetes Objekt anhängen, bei dem es sich entweder um ein Gruppen-, ein Menü- oder ein Application-Objekt handeln kann. Wenn das neu zugewiesene MOAI-Objekt ein Fensterobjekt ist, müssen Sie es an das Application-Objekt anfügen, indem Sie die Methode `Application.AddWindow` verwenden. Um Menüobjekte anzuhängen, müssen Sie die Methoden `Menubar.Prepend`, `Menubar.Append`, `Menubar.Insert`, `Menu.Prepend`, `Menu.Append` oder `Menu.Insert` verwenden. Alle anderen Objekte können mit den Gruppenmethoden `Group.Prepend`, `Group.Append` und `Group.Insert` angefügt werden.

Für die meisten MOAI-Klassen müssen Sie im zweiten Parameter `parent$` ein übergeordnetes Objekt angeben. RapaGUI wird das neue Objekt nicht an das übergeordnete Objekt anhängen, aber es muss immer noch das übergeordnete Objekt kennen, um bestimmte Einstellungen festlegen zu können. Die einzigen MOAI-Klassen, für die Sie kein übergeordnetes Objekt angeben müssen, sind die folgenden:

- Window-Klasse (Fenster)
- Dialog-Klasse
- Menubar-Klasse (Menüleisten)
- Menu-Klasse

- MenuItem-Klasse (Menüelement)

Bei allen anderen MOAI-Klassen müssen Sie den Identifikator des übergeordneten Objekts angeben.

Normalerweise müssen Sie nur den Identifikator des Fensters übergeben, an das Sie das MOAI-Objekt als übergeordnetes Objekt anhängen möchten. Es gibt eine Ausnahme: Wenn Sie planen, das MOAI-Objekt einer Gruppe mit einem Rahmen hinzuzufügen, d.h. eine Instanz von der Group-Klasse (Gruppen) mit `Group.Frame` auf `True` gesetzt, dann müssen Sie dieses Gruppenobjekt als übergeordnetes Objekt übergeben. Beachten Sie, dass dies nur für Gruppen mit einem Rahmen gilt. Wenn Sie vorhaben, das MOAI-Objekt einer normalen Gruppe hinzuzufügen, die keinen Rahmen hat, müssen Sie nur den Identifikator des Fensters an `moai.CreateObject()` übergeben. Beachten Sie jedoch, dass für den Fall, dass die normale Gruppe selbst nur ein Element ist, das irgendwo in der Hierarchie einer gerahmten Gruppe angehört, Sie den Identifikator dieser gerahmten Gruppe übergeben müssen. Wenn Sie Objekte zu einer Gruppe der Scrollgroup-Klasse hinzufügen, müssen Sie zudem auch den Identifikator dieser Scrollgroup als übergeordnetes Objekt übergeben.

Um es abstrakt auszudrücken: Das übergeordnete Objekt muss auf das nächste Objekt in der Layout-Hierarchie gesetzt werden, das eine visuelle Darstellung hat. Normale Gruppen sind nur Layout-Werkzeuge. Sie existieren nicht als Widget. Gerahmte Gruppen hingegen sind zwar Layout-Werkzeuge, aber sie sind auch Widgets, da sie eine visuelle Darstellung haben. Wenn Ihr Objekt also irgendwo in eine gerahmte Gruppe eingebettet werden soll, muss die gerahmte Gruppe als übergeordnetes Objekt übergeben werden. Wenn es in eine normale Gruppe eingebettet ist, dann muss das Fenster der obersten Ebene das übergeordnete Fenster sein. Vergessen Sie nicht, in Hierarchien zu denken: Auch wenn Sie Ihr Objekt an eine normale Gruppe anhängen wollen, kann es sein, dass die normale Gruppe selbst in eine gerahmte Gruppe eingebettet ist, was bedeutet, dass Sie die gerahmte Gruppe trotzdem als übergeordnete Gruppe übergeben müssen. Dies trifft auch auf Objekte der Scrollgroup-Klasse zu, da diese auch eine visuelle Darstellung haben.

Im Gegensatz zu `moai.CreateApp()` können Sie `moai.CreateObject()` beliebig oft aufrufen, da es kein Application-Objekt für Sie erstellt, sondern nur losgelöste MOAI-Objekte, von denen Sie beliebig viele haben können.

Es ist wichtig, dass Sie in der XML-Definition eine ID für Ihr MOAI-Objekt angeben, da Sie diese ID verwenden müssen, um auf dieses Objekt zu verweisen, wenn Sie es zu einem Application-, Menü- oder Gruppenobjekt hinzufügen möchten.

Sobald dieser Befehl beendet ist, können Sie mit dem neu erstellten MOAI-Objekt (und allen seinen untergeordneten) die Befehle `moai.Set()`, `moai.Get()` und `moai.DoMethod()` benutzen.

Losgelöste MOAI-Objekte können mit dem Befehl `moai.FreeObject()` gelöscht werden. Dies müssen Sie aber nur in bestimmten Fällen aufrufen, z.B. wenn Sie es mit vielen dynamisch zugewiesenen MOAI-Objekten zu tun haben und Sie etwas Speicher und Ressourcen sparen wollen.

EINGABEN

`xml$` eine Zeichenkette, die eine XML MOAI-Objektbeschreibung enthält

`parent$` optional: gewünschtes übergeordnetes Objekt für das Objekt; Details siehe oben

BEISPIEL

```
moai.CreateObject([[
<window id="newwindow" title="A new window">
  <vgroup>
    <button id="btn">Hello World!</button>
  </vgroup>
</window>
]])
```

```
moai.DoMethod("app", "addwindow", "newwindow")
moai.Set("newwindow", "open", True)
```

Der obige Code erzeugt ein neues Fenster, fügt es dem bestehenden Application-Objekt hinzu und öffnet es.

```
moai.CreateObject([[
<button id="newbutton">Dynamically created button!</button>
]], "mywindow")
```

```
moai.DoMethod("mygroup", "initchange")
moai.DoMethod("mygroup", "append", "newbutton")
moai.DoMethod("mygroup", "exitchange", false)
```

Der obige Code erzeugt dynamisch ein neues Schaltflächen-Objekt und fügt es als letztes Element der Gruppe mit der ID "mygroup" hinzu.

6.4 moai.DoMethod

BEZEICHNUNG

`moai.DoMethod` – wendet eine Methode auf ein MOAI-Objekt an

ÜBERSICHT

```
r = moai.DoMethod(id$, method$, ...)
```

BESCHREIBUNG

Dieser Befehl kann verwendet werden, um eine Methode für das angegebene MOAI-Objekt auszuführen. Sie müssen den Identifikator des MOAI-Objekts im ersten Argument `id$` und den Namen der Methode im zweiten Argument `method$` übergeben. Bei Methoden- und Objekt-IDs wird nicht zwischen Groß- und Kleinschreibung unterschieden. Es spielt also keine Rolle, ob Sie Groß- oder Kleinbuchstaben verwenden.

Welche Methoden Sie mit diesem Befehl verwenden können, hängt von der Klasse des angegebenen MOAI-Objekts ab. Schauen Sie sich die Klassenreferenz an, um zu sehen, welche Methoden von den verschiedenen MOAI-Klassen unterstützt werden.

Auch die Argumente, die Sie diesem Befehl nach dem Methodennamen übergeben müssen, hängen von der Methode ab. Sie sind für jede Methode unterschiedlich.

Dasselbe gilt für die Rückgabewerte. Einige Methoden geben Werte zurück, andere nicht. Bitte beachten Sie die Klassenreferenz, um zu sehen, welche Argumente Ihre Methode benötigt und ob es Rückgabewerte für sie gibt.

EINGABEN

`id$` Identifikator des MOAI-Objekt, auf dem die Methode angewendet wird
`method$` Methodenname als Zeichenkette
`...` weitere Argumente sind methodenabhängig (Details siehe Klassenreferenz)

RÜCKGABEWERTE

`r` der Rückgabewert hängt vom Methodentyp ab

BEISPIEL

```
moai.DoMethod("my_listview", "insert", "bottom", "Last entry")
```

Der obige Code fügt einen neuen Eintrag mit dem Namen "Last entry" am unteren Rand der Listenansicht unter Verwendung des Identifikators "my_listview" hinzu. Dies geschieht durch die Ausführen der Methode `Listview.Insert` auf dem Objekt `Listview`.

6.5 moai.FreeApp

BEZEICHNUNG

`moai.FreeApp` – löscht das gesamte Application-Objekt

ÜBERSICHT

```
moai.FreeApp()
```

BESCHREIBUNG

Verwenden Sie diesen Befehl, um das gesamte Application-Objekt zu löschen, das mit dem letzten Befehl `moai.CreateApp()` erstellt wurde. `moai.FreeApp()` löscht das gesamte Application-Objekt und alle daran angehängten Elementen. Nachdem dieser Befehl beendet wurde, können Sie, wenn Sie möchten, eine neues Application-Objekt mit dem Befehl `moai.CreateApp()` erstellen.

Beachten Sie, dass `moai.FreeApp()` keine MOAI-Objekte löscht, die sich momentan in einem losgelösten Zustand befinden. Diese müssen mit `moai.FreeObject()` gelöscht werden.

EINGABEN

keine

6.6 moai.FreeDialog

BEZEICHNUNG

`moai.FreeDialog` – löscht ein Dialogobjekt

ÜBERSICHT

```
moai.FreeDialog(id$)
```

BESCHREIBUNG

Dieser Befehl kann verwendet werden, um ein mit `moai.CreateDialog()` erzeugtes Dialogobjekt zu löschen. Normalerweise ist es nicht notwendig, diesen Befehl aufzurufen, da Dialoge automatisch durch `Dialog.EndModal` gelöscht werden oder wenn der Dialogbefehl beendet wird. Wenn Sie eine fein abgestimmte Kontrolle über die Löschung von Dialogen benötigen, können Sie diesen Befehl jedoch verwenden.

`moai.FreeDialog()` löst den Dialog zunächst vom Application-Objekt und löscht es dann. Technisch gesehen ist dieser Befehl also nur ein Komfortbefehl, der das Dialogobjekt intern aus dem Application-Objekt entfernt, indem sie `Application.RemoveWindow` und dann `moai.FreeObject()` aufruft, um das Dialogobjekt zu löschen. `moai.FreeDialog()` kombiniert diese beiden Schritte einfach in einem.

Siehe [Abschnitt 16.1 \[Dialog-Klasse\]](#), Seite 99, für mehr Informationen über Dialoge.

EINGABEN

`id$` Identifikator des Dialogobjekts, welches gelöscht wird

6.7 moai.FreeImage

BEZEICHNUNG

`moai.FreeImage` – löscht das Bild im Bilder-Cache (V1.2)

ÜBERSICHT

```
moai.FreeImage(id[, isicon])
```

BESCHREIBUNG

Dieser Befehl kann verwendet werden, um ein Bild im internen Bilder-Cache von RapaGUI zu löschen. In `id` müssen Sie den Identifikator des freizugebenden Bildes übergeben. Wenn das optionale Argument `isicon` auf `True` gesetzt ist, muss `id` der Identifikator eines zu löschenden Hollywood-Icon sein. Andernfalls muss `id` der Identifikator eines zu löschenden Hollywood-Pinsel sein. Alternativ können Sie auch `-1` in `id` übergeben, um alle Bilder im internen Cache von RapaGUI zu löschen.

Sie müssen sicherstellen, dass das angegebene Bild nicht mehr von irgendwelchen Widgets in Ihrer GUI verwendet wird, bevor Sie diesen Befehl aufrufen. Beachten Sie, dass es unter normalen Bedingungen nicht notwendig ist, diesen Befehl aufzurufen, da normalerweise alle Bilder automatisch von RapaGUI aus dem Speicher gelöscht werden. Unter bestimmten Umständen kann es jedoch sinnvoll sein, diesen Befehl aufzurufen.

RapaGUI speichert alle Bilder, die Sie in Ihrer GUI verwenden. Wenn Sie also ein zweites Mal versuchen, ein bestimmtes Bild in Ihrer GUI zu verwenden, wird es aus Performancegründen einfach aus dem internen Bilder-Cache von RapaGUI geladen. Dies kann zu unerwünschtem Verhalten führen, falls Sie die Grafik Ihres Bildes zwischenzeitlich aktualisiert haben und RapaGUI die aktualisierten Grafiken verwenden soll. In diesem Fall müssen Sie zuerst das Bild im internen Bilder-Cache von RapaGUI mit diesem Befehl löschen. Wenn Sie das Bild dann erneut an RapaGUI übergeben, wird es aus dem aktuellen Inhalt des Bildes neu erstellt und neu zwischengespeichert. Für Listview- und Treeview-Widgets (Listenansichts- und Baumansichts-Widgets) können Sie auch `moai.UpdateImage()` verwenden, um die Grafikdaten eines Bildes zu aktualisieren.

Siehe [Abschnitt 3.20 \[Bilder-Cache\]](#), Seite 32, für Details.

EINGABEN

<code>id</code>	Identifikator des Bildes, das aus dem Bilder-Cache gelöscht werden soll oder -1 für das Löschen aller Bilder
<code>isicon</code>	optional: <code>True</code> , wenn <code>id</code> den Identifikator eines Icons enthält, <code>False</code> , wenn er den Identifikator eines Pinsels enthält (voreingestellt ist <code>False</code>) (V2.0)

6.8 moai.FreeObject

BEZEICHNUNG

`moai.FreeObject` – löscht ein losgelöstes MOAI-Objekt

ÜBERSICHT

`moai.FreeObject(id$)`

BESCHREIBUNG

Dieser Befehl kann verwendet werden, um ein losgelöstes MOAI-Objekt zu löschen, das entweder von `moai.CreateObject()` oder `moai.CreateApp()` erstellt wurde. Das MOAI-Objekt, das Sie hier angeben, darf nicht länger an ein Application-, Gruppen- oder Menüobjekt angehängt sein, da angehängte MOAI-Objekte mit ihren übergeordneten Objekten freigegeben werden. Stellen Sie daher sicher, dass Sie diesen Befehl nur für MOAI-Objekte verwenden, die von ihren übergeordneten Objekten losgelöst wurden und nicht mehr an ein übergeordnetes Objekt gebunden sind.

Um MOAI-Objekte von ihren jeweiligen übergeordneten Objekten loszulösen, müssen Sie eine der folgenden Methoden verwenden: `Application.RemoveWindow`, `Menubar.Remove`, `Menu.Remove` oder `Group.Remove`.

Wenn RapaGUI beendet wird, werden alle losgelösten MOAI-Objekte automatisch aus dem Speicher gelöscht, so dass Sie diesen Befehl normalerweise überhaupt nicht aufrufen müssen, wenn Ihr Programm nicht ständig MOAI-Objekte zur Laufzeit hinzufügt und entfernt.

EINGABEN

<code>id\$</code>	Identifikator des MOAI-Objekts, welches gelöscht werden soll
-------------------	--

6.9 moai.Get

BEZEICHNUNG

`moai.Get` – gibt den Wert eines MOAI-Objekt-Attributs zurück

ÜBERSICHT

`r = moai.Get(id$, attr$)`

BESCHREIBUNG

Mit diesem Befehl kann der aktuelle Wert des Attributs `attr$` im in `id$` angegebenen MOAI-Objekt ermittelt werden. Attributnamen und Objekt-IDs sind unabhängig von Groß- und Kleinschreibung.

Die Attribute, die Sie mit diesem Befehl verwenden können, hängen von der Klasse des angegebenen MOAI-Objekts ab. Schauen Sie sich die Klassenreferenz an, um zu sehen, welche Attribute von den verschiedenen MOAI-Klassen unterstützt werden. Um ein Attribut mit diesem Befehl zu verwenden, muss es eine Anwendbarkeit von "G" haben. Attribute von Area-Klasse (Bereich) und MOAI-Klasse können für fast alle anderen Klassen verwendet werden, da die Area- und MOAI-Klassen für die meisten anderen Klassen als Oberklasse fungieren.

EINGABEN

`id$` Identifikator des abzufragenden MOAI-Objekts
`attr$` Attribut, dessen Wert abgerufen werden soll

RÜCKGABEWERTE

`r` aktueller Attributwert

BEISPIEL

```
DebugPrint(moai.Get("my_listview", "active"))
```

Der obige Code gibt den Index des aktuell aktiven Eintrags in der Listenansicht zurück, der den Identifikator "my_listview" hat, indem er das Attribut `Listview.Active` abfragt.

6.10 moai.HaveObject

BEZEICHNUNG

`moai.HaveObject` – prüft, ob ein MOAI-Objekt vorhanden ist

ÜBERSICHT

```
r = moai.HaveObject(id$)
```

BESCHREIBUNG

Dieser Befehl prüft lediglich, ob das MOAI-Objekt im angegebenen Attribut `id$` existiert oder nicht. Falls vorhanden, wird `True` zurückgegeben, ansonsten `False`.

EINGABEN

`id$` Identifikator des zu prüfenden MOAI-Objekts

RÜCKGABEWERTE

`r` `True` oder `False`, je nachdem, ob das Objekt existiert oder nicht.

6.11 moai.Notify

BEZEICHNUNG

`moai.Notify` – fügt eine Benachrichtigung einem MOAI-Objekt hinzu oder enternt sie

ÜBERSICHT

```
moai.Notify(id$, attr$, enable)
```

BESCHREIBUNG

Dieser Befehl kann verwendet werden, um eine Benachrichtigung für das Attribut `attr$` in dem in `id$` angegebenen MOAI-Objekt hinzuzufügen oder zu entfernen. Sie müssen den Attributnamen und ein boolesches Flag übergeben, das angibt, ob Sie Benachrichtigungen für dieses Attribut aktivieren oder deaktivieren möchten. Attributnamen und Objekt-IDs unterscheiden nicht zwischen Groß- und Kleinschreibung, d.h. es spielt keine Rolle, ob Sie für sie Groß- oder Kleinbuchstaben verwenden.

Welche Attribute Sie mit diesem Befehl verwenden können, hängt von der Klasse des angegebenen MOAI-Objekts ab. Welche Attribute von den verschiedenen MOAI-Klassen unterstützt werden, sehen Sie in der Klassenreferenz. Um ein Attribut mit diesem Befehl verwenden zu können, muss es eine Anwendbarkeit von "N" haben. Attribute von Area-Klasse (Bereich) und MOAI-Klasse können auf fast allen anderen Klassen verwendet werden, da die Area- und MOAI-Klassen als Oberklassen für die meisten anderen Klassen dienen.

Sobald Sie eine Benachrichtigung über ein bestimmtes Objektattribut eingerichtet haben, können Sie diese Ereignisse überwachen, indem Sie eine RapaGUI-Callback-Funktion mit dem Hollywood-Befehl `InstallEventHandler()` installieren. Siehe [Abschnitt 3.7 \[Benachrichtigungen der Attribute\]](#), Seite 14, für Details.

Beachten Sie, dass Benachrichtigungen auch in der XML-GUI-Definition mithilfe des Attributs `Notify` eingerichtet werden können. Siehe [Abschnitt 3.7 \[Benachrichtigungen der Attribute\]](#), Seite 14, für Details.

EINGABEN

<code>id\$</code>	ID des MOAI-Objekts
<code>attr\$</code>	Attribut, bei dem Sie eine Benachrichtigung hinzufügen oder entfernen wollen
<code>enable</code>	<code>True</code> , um eine Benachrichtigung hinzuzufügen oder <code>False</code> , um eine Benachrichtigung aus diesem Objekt zu entfernen

BEISPIEL

```
moai.Notify("my_listview", "active", True)
```

Der obige Code installiert eine Benachrichtigung, die ausgelöst wird, wenn sich das Attribut `Listview.Active` in der Listenansicht mit dem Identifikator "my_listview" ändert.

6.12 moai.Request

BEZEICHNUNG

`moai.Request` – öffnet ein Auswahldialogfenster

ÜBERSICHT

```
r = moai.Request(title$, body$, buts$[, icon$])
```

BESCHREIBUNG

Dieser Befehl öffnet einen Standard-Auswahldialogfenster, das die in `body$` angegebene Nachricht anzeigt und dem Benutzer außerdem erlaubt, eine Auswahl unter Verwendung einer der durch `buts$` spezifizierten Schaltflächen zu treffen. Optional können Sie auch

ein Symbol in `icon$` angeben, das im Dialogfenster angezeigt werden soll. Wenn Sie in `title$` eine leere Zeichenfolge angeben, wird automatisch der in `@APPTITLE` angegebene Titel verwendet.

Die in `butts$` angegebenen Schaltflächen trennen Sie mit einem "|". Der Rückgabewert `r` gibt an, welche Taste der Benutzer gedrückt hat. Bitte beachten Sie, dass die rechte Schaltfläche immer den Wert `False` (0) hat, da er typischerweise als "Abbrechen"-Schaltfläche verwendet wird. Wenn Sie zum Beispiel drei Schaltflächen "Eins|Zwei|Drei" gewählt haben, hat die Schaltfläche "Drei" den Rückgabewert 0, "Zwei" gibt 2 und "Eins" gibt 1 zurück.

Das optionale Argument `icon$` kann auf einen der folgenden vordefinierten Werte gesetzt werden:

"None": Kein Symbol
 "Information": Ein Informationszeichen
 "Error": Ein Fehlerzeichen
 "Warning": Ein Warnzeichen
 "Question": Ein Fragezeichen

EINGABEN

`title$` Title für das Dialogfenster; übergeben Sie eine leere Zeichenfolge (""), wird der `@APPTITLE` verwendet
`body$` Text, der als Nachricht im Dialogfenster erscheint
`butts$` eine oder mehrere Schaltflächen, die der Benutzer drücken kann
`icon$` optional: Symbol, welches im Dialogfenster angezeigt werden kann (Voreinstellung ist "Information")

RÜCKGABEWERTE

`r` die Schaltfläche, die durch den Benutzer gedrückt wurde

BEISPIEL

```
moai.Request("RapaGUI", "Hello!\n\n" ..
  "Do you like RapaGUI!", "Yes|No")
```

Der obige Code demonstriert die Verwendung des Befehls `moai.Request()`.

6.13 moai.Set

BEZEICHNUNG

`moai.Set` – setzt den Wert eines MOAI-Objekt-Attributs

ÜBERSICHT

```
moai.Set(id$, attr1$, val1$, ...)
```

BESCHREIBUNG

Mit diesem Befehl kann der aktuelle Wert eines oder mehrerer Attribute in dem in `id$` angegebenen MOAI-Objekt gesetzt werden. Für jedes zu ändernde Attribut müssen Sie den Attributnamen und den gewünschten neuen Wert übergeben. Sie können diese Attribut/Wertpaare beliebig oft wiederholen, um mehrere Attribute mit nur einem einzigen Aufruf von `moai.Set()` zu ändern. Attributnamen und Objekt-IDs sind unabhängig von Groß- und Kleinschreibung.

Welche Attribute Sie mit diesem Befehl verwenden können, hängt von der Klasse des angegebenen MOAI-Objekts ab. Welche Attribute von den verschiedenen MOAI-Klassen unterstützt werden, sehen Sie in der Klassenreferenz. Um ein Attribut mit diesem Befehl verwenden zu können, muss es eine Anwendbarkeit von "S" haben. Attribute der Area-Klasse (Bereich) und MOAI-Klasse können auf fast allen anderen Klassen verwendet werden, da die Area- und MOAI-Klassen als Oberklassen für die meisten anderen Klassen dienen.

Wenn Sie mit diesem Befehl eine Benachrichtigung für das Attribut erstellt haben, das Sie ändern möchten, wird die Benachrichtigung ausgelöst, sobald Sie `moai.Set()` für dieses Attribut aufrufen. Wenn Sie dieses Verhalten nicht möchten, können Sie das Attribut `MOAI.NoNotify` verwenden, um zu verhindern, dass eine Benachrichtigung ausgegeben wird.

EINGABEN

<code>id\$</code>	Identifikator des zu ändernden MOAI-Objekts
<code>attr1\$</code>	Attribut, dessen Wert geändert werden soll
<code>val1\$</code>	neuer Wert für das Attribut
<code>...</code>	Sie können Attribut/Wertpaare beliebig oft wiederholen

BEISPIEL

```
moai.Set("my_listview", "active", 15)
```

Der obige Code setzt die Eintragsnummer 15 als derzeit aktiven Eintrag in der Listenansicht mit dem Identifikator "my_listview", indem das Attribut `Listview.Active` gesetzt wird.

```
moai.Set("my_listview", "nonotify", True, "active", 15)
```

Dieser Code macht dasselbe wie der obige Code, verhindert aber, dass Benachrichtigungen ausgegeben werden, indem das Attribut `MOAI.NoNotify` auf `True` gesetzt wird. Dies ist nützlich, wenn Sie zwischen Benutzerselektionen in der Listenansicht und programmgesteuerten Selektionen mit `moai.Set()` unterscheiden müssen.

6.14 moai.UpdateImage

BEZEICHNUNG

`moai.UpdateImage` – aktualisiert das zwischengespeicherte Bild (V2.0)

ÜBERSICHT

```
moai.UpdateImage(id[, isicon])
```

BESCHREIBUNG

Dieser Befehl kann verwendet werden, um die Grafikdaten eines Bildes im internen Cache von RapaGUI zu aktualisieren. In `id` müssen Sie den Identifikator des zu aktualisierenden Bildes übergeben. Der optionale Parameter `isicon` gibt an, ob das zwischengespeicherte Bild ein Hollywood-Pinsel oder ein Hollywood-Icon ist. Die neuen Grafikdaten für das Bild werden auch aus dem Hollywood-Pinsel oder -Icon entnommen, das durch `id` angegeben wird.

Beachten Sie, dass dieser Befehl derzeit nur mit Listview- und Treeview-Widgets (Listenansichts- und Baumansichts-Widgets) kompatibel ist. Sie können ihn nicht verwenden, um die Grafiken von Icons in anderen Widgets zu aktualisieren, z.B. Icons in einem Schaltflächen-Widget. Nach dem Aktualisieren der Grafikdaten eines Icons in einem Listview- oder Treeview-Widget müssen Sie außerdem sicherstellen, dass alle Zeilen, die dieses Icon verwenden, aktualisiert werden. Dies können Sie beispielsweise mit den Methoden `Listview.Rename` oder `Treeviewleaf.SetItem` erreichen.

Siehe [Abschnitt 3.20 \[Bilder-Cache\]](#), [Seite 32](#), für Details.

EINGABEN

<code>id</code>	Identifikator des zu aktualisierenden Bildes
<code>isicon</code>	optional: <code>True</code> , wenn <code>id</code> den Identifikator eines Icons enthält, <code>False</code> , wenn er den Identifikator eines Pinsels enthält (voreingestellt ist <code>False</code>)

7 Accelerator-Klasse (Tastaturkürzel)

7.1 Übersicht

Die Accelerator-Klasse (Tastaturkürzel) kann verwendet werden, um globale Tastaturkürzel für ein Fenster einzurichten. Normale Tastaturkürzel, die mit dem Unterstrichzeichen definiert wurden, werden von RapaGUI automatisch überwacht. Siehe [Abschnitt 3.13 \[Tastaturkürzel\]](#), [Seite 22](#), für Details. Manchmal möchten Sie jedoch komplexere Tastaturkürzel verwenden, z.B. die Escape-Taste, Funktionstasten, bestimmte Steuertasten oder Tastenkombinationen (z.B. CTRL+V zum Einfügen). Dies ist durch die Verwendung der Accelerator-Klasse möglich.

Mit der Accelerator-Klasse können Sie eine Tabelle von Tastaturkürzeln einrichten, die dann einem Fenster zugewiesen werden können, indem Sie das Attribut `Window.Accelerator` verwenden, um diese Tastaturkürzel im jeweiligen Fenster global verfügbar zu machen. Jede Tastaturkürzel-Tabelle muss mindestens ein Element vom Typ Acceleratoritem-Klasse enthalten, das eine einzige Tastaturkürzel definiert.

Hier ist ein Beispiel, wie man eine Tastaturkürzel-Tabelle in XML einrichtet:

```
<accelerator>
  <item id="ac_cut" mod="ctrl">X</item>
  <item id="ac_copy" mod="ctrl">C</item>
  <item id="ac_paste" mod="ctrl">V</item>
  <item id="ac_f1">F1</item>
</accelerator>
```

Die oben angegebene Tastaturkürzel-Tabelle legt vier verschiedene Tastaturkürzel fest: CTRL+X, CTRL+C, CTRL+V und F1. Wenn der Benutzer eine solche Taste oder Tastenkombination drückt, wird die Benachrichtigung `Acceleratoritem.Pressed` ausgelöst und Ihr Programm kann darauf reagieren.

Beachten Sie, dass Sie, wenn Sie beabsichtigen, Ihre Tastaturkürzel-Einträge mit Menüeinträgen zu verknüpfen, überhaupt keine Accelerator-Klasse verwenden müssen. In diesem Fall können Sie einfach das Attribut `MenuItem.Shortcut` verwenden, um Verknüpfungen für Ihre Menüeinträge einzurichten. Accelerator-Klasse ist wirklich nur notwendig, wenn Sie Tastatur-Ereignisse überwachen müssen, ohne irgendwelche Menüeinträge zu haben.

Die Accelerator-Klasse definiert selbst keine Attribute oder Methoden. Siehe [Abschnitt 8.1 \[Acceleratoritem-Klasse\]](#), [Seite 63](#), für alle notwendigen Informationen.

8 Acceleratoritem-Klasse (Tastaturkürzelement)

8.1 Übersicht

Die Acceleratoritem-Klasse (Tastaturkürzelement) wird verwendet, um eine einzelne Tastenkombination als Teil einer Tastaturkürzel-Tabelle für die Accelerator-Klasse zu erstellen. Sie können keine unabhängigen Instanzen der Acceleratoritem-Klasse erstellen. Sie müssen immer in die Accelerator-Klasse eingebettet werden. Siehe [Abschnitt 7.1 \[Accelerator-Klasse\], Seite 61](#), für Details.

Die folgenden Tasten können derzeit beim Erstellen eines Tastaturkürzel-Eintrags mit dem XML-Tag `<item>` angegeben werden:

UP	Pfeil nach oben
DOWN	Pfeil nach unten
RIGHT	Pfeil nach rechts
LEFT	Pfeil nach links
HELP	Help-Taste
DEL	Delete-Taste
BACKSPACE	Rückschritt-Taste
TAB	Tabulator-Taste
RETURN	Return-Taste
ENTER	Enter-Taste
ESC	Escape-Taste
SPACE	Leertaste
F1 - F16	Funktionstasten
INSERT	Einfüge-Taste
HOME	Anfangs-Taste
END	Ende-Taste
PAGEUP	Seite rauf
PAGEDOWN	Seite runter
PRINT	Drucken-Taste
PAUSE	Pause-Taste

Darüber hinaus unterstützt die Acceleratoritem-Klasse die englischen Buchstaben von A bis Z sowie die Zahlen 0 bis 9 als Tastendefinition. Dabei spielt es keine Rolle, ob Sie alle diese Tasten in Groß-, Klein- oder Mischschreibung angeben. Siehe [Abschnitt 7.1 \[Accelerator-Klasse\], Seite 61](#), für ein Beispiel.

8.2 Acceleratoritem.Mod

BEZEICHNUNG

Acceleratoritem.Mod – setzt die Umschalttaste(n)

BESCHREIBUNG

Mit diesem Attribut können Sie den oder die Umschalttasten für dieses Tastaturkürzel-Element angeben. Dies kann eine Kombination der folgenden vordefinierten Werte sein:

None	Verwendet keine Umschalttasten. Dies ist die Voreinstellung.
Ctrl	Fügt die Control-Taste zu den Umschalttasten hinzu.
Alt	Fügt die Alt-Taste zu den Umschalttasten hinzu.
Cmd	Fügt die Steuer-Taste zu den Umschalttasten hinzu. Dies wird nur auf macOS und AmigaOS unterstützt.
Shift	Fügt die Shift-Taste zu den Umschalttasten hinzu.

Wenn Sie mehr als eine Umschalttaste angeben, müssen Sie die einzelnen Tasten durch ein Semikolon (;) trennen, z.B. "**ctrl;shift**".

Bei Verwendung von Umschalttasten wird die Tastenkombination nur ausgelöst, wenn die Umschalttaste(n) und die Haupttaste(n) gedrückt sind. Dies ist nützlich für die Überwachung von Tastenkombinationen wie CTRL+V zum Einfügen von Daten, etc.

TYP

Zeichenkette

ANWENDBARKEIT

I

8.3 Acceleratoritem.Pressed

BEZEICHNUNG

Acceleratoritem.Pressed – benachrichtigt, wenn das Tastaturkürzel gedrückt wird

BESCHREIBUNG

Dieses Attribut wird ausgelöst, wenn der Benutzer die Taste(n) drückt, die durch dieses Tastaturkürzel-Element definiert sind. RapaGUI überwacht automatisch dieses Attribut, so dass Sie nicht explizit eine Benachrichtigung mit dem Attribut MOAI.Notify anfordern müssen.

TYP

Boolesch

ANWENDBARKEIT

N

9 Application-Klasse (Anwendung)

9.1 Übersicht

Die Application-Klasse (Anwendung) ist die Hauptklasse, die ein Programm verwaltet. Daher kann es in jedem Programm nur eine Instanz dieser Klasse geben. Diese Instanz wird durch einen Aufruf von `moai.CreateApp()` erzeugt. Alle Fenster, die ein Programm anzeigt, sind Elemente der Application-Klasse. Jede GUI-Definition muss also mit der Application-Klasse beginnen.

9.2 Application.AboutMUI

BEZEICHNUNG

Application.AboutMUI – zeigt das Über-MUI-Fenster an

ÜBERSICHT

```
moai.DoMethod(id, "AboutMUI")
```

PLATTFORMEN

Nur AmigaOS und kompatible Betriebssysteme

BESCHREIBUNG

Zeigt das Über-MUI-Fenster an. Aus der MUI-Styleguide ist zu entnehmen, dass alle MUI-Programme einen Menüpunkt namens "Über MUI ..." enthalten sollten.

EINGABEN

id ID des Applications-Objekts

9.3 Application.AboutRapaGUI

BEZEICHNUNG

Application.AboutRapaGUI – zeigt das Über-RapaGUI-Fenster an

ÜBERSICHT

```
moai.DoMethod(id, "AboutRapaGUI")
```

BESCHREIBUNG

Diese Methode zeigt das Über-RapaGUI-Fenster an. Sie sollten den Menüpunkt "Über RapaGUI...." in alle Ihre Programme aufnehmen.

EINGABEN

id ID des Applications-Objekts

9.4 Application.AddWindow

BEZEICHNUNG

Application.AddWindow – fügt ein losgelöstes Fensterobjekt zum Applications-Objekt hinzu

ÜBERSICHT

```
moai.DoMethod(id, "AddWindow", window)
```

BESCHREIBUNG

Diese Methode kann verwendet werden, um dem Applications-Objekt ein losgelöstes Fensterobjekt hinzuzufügen. Nachdem das Fensterobjekt an das Applications-Objekt angehängt wurde, können Sie das Fenster öffnen, indem Sie sein Attribut `Window.Open` setzen.

Losgelöste Fensterobjekte können entweder durch Aufruf des Befehls `moai.CreateObject()` erzeugt werden oder Sie können mit der Methode `Application.RemoveWindow` ein Fensterobjekt explizite von ihrem übergeordneten Objekt loslösen.

Bitte beachten Sie, dass Sie die Methode `Application.AddWindow` nicht für Dialoge aufrufen müssen, die mit dem Befehl `moai.CreateDialog()` erstellt wurden, da `moai.CreateDialog()` bereits intern `Application.AddWindow` aufruft.

EINGABEN

`id` ID des Applications-Objekts
`window` ID des hinzuzufügenden Fensterobjekts

BEISPIEL

Siehe [Abschnitt 6.3 \[moai.CreateObject\]](#), Seite 49.

9.5 Application.ContextMenu

BEZEICHNUNG

`Application.ContextMenu` – setzt das Kontextmenü für das Programm (V2.0)

PLATTFORMEN

Windows, macOS, Linux

BESCHREIBUNG

Wenn Ihr Programm mit Hollywoods Befehl `SetTrayIcon()` ein Icon in der Taskleiste installiert hat, können Sie dieses Attribut verwenden, um dem Taskleisten-Icon des Programmes ein Kontextmenü hinzuzufügen. Setzen Sie dieses Attribut einfach auf einen Identifikator eines MOAI-Objekts, welches von der Menu-Klasse abgeleitet ist, und das Icon Ihres Programmes in der Taskleiste erhält ein Kontextmenü. Immer wenn der Benutzer mit der rechten Maustaste auf das Icon in der Taskleiste drückt, wird das Kontextmenü angezeigt.

Siehe [Abschnitt 3.16 \[Kontextmenüs\]](#), Seite 25, für ein Beispiel.

TYP

MOAI-Objekt

ANWENDBARKEIT

I

9.6 Application.HelpFile

BEZEICHNUNG

Application.HelpFile – setzt/ermittelt die Hilfedatei für das Programm (V2.0)

BESCHREIBUNG

Legen Sie eine Hilfedatei für das Programm fest. Unter Windows sollte dies auf eine CHM-Datei, auf einem Amiga auf eine AmigaGuide-Datei und auf allen anderen Plattformen auf ein Verzeichnis verweisen, das alle HTML-Dateien enthält, die zum Handbuch des Programmes gehören (die Hauptdatei muss `index.html`, die Indexdatei muss `findex.html` heißen).

Nachdem Sie eine Hilfedatei definiert haben, können Sie diese mit `Application.ShowHelp` und `Application.ShowHelpNode` öffnen.

Dies wird derzeit auf Android nicht unterstützt.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

9.7 Application.Icon

BEZEICHNUNG

Application.Icon – setzt das Programm-Icon

PLATTFORMEN

Nur AmigaOS und kompatible Betriebssysteme

BESCHREIBUNG

Mit diesem Tag können Sie das Icon festlegen, das auf dem Bildschirm der Workbench angezeigt wird, wenn das Programm ikonisiert wird. Sie müssen diesen Tag auf den Dateinamen einer Amiga-Icon-Datei `.info` setzen.

Normalerweise sollten Sie die Icons für Ihre Anwendung mit der Präprozessor-Anweisung `@APPICON` von Hollywood setzen. Dieser Tag ist nur hier, um Amiga-Programme auf der Workbench zu unterstützen.

TYP

Zeichenkette

ANWENDBARKEIT

I

9.8 Application.OpenConfigWindow

BEZEICHNUNG

Application.OpenConfigWindow – zeigt das MUI-Einstellungsfenster

ÜBERSICHT

```
moai.DoMethod(id, "OpenConfigWindow")
```

PLATTFORMEN

Nur AmigaOS und kompatible Betriebssysteme

BESCHREIBUNG

Mit dieser Methode können Sie das Einstellungsfenster von MUI für das Programm öffnen. MUI unterstützt individuelle Benutzeroberflächeneinstellungen für jedes Programm, weshalb Sie jedem MUI-Programm einen Menüpunkt wie "Einstellungen/MUI...." hinzufügen sollten, damit der Benutzer diese Einstellungen vornehmen kann.

EINGABEN

`id` ID des Applications-Objekts

9.9 Application.RemoveWindow**BEZEICHNUNG**

`Application.RemoveWindow` – löst ein Fensterobjekt vom Applications-Objekt

ÜBERSICHT

```
moai.DoMethod(id, "RemoveWindow", window)
```

BESCHREIBUNG

Diese Methode entfernt das angegebene Fensterobjekt aus dem Applications-Objekt und versetzt das Fensterobjekt in den Zustand "losgelöst". Fensterobjekte im losgelösten Zustand können entweder durch Ausführen der Methode `Application.AddWindow` wieder angefügt oder durch den Befehl `moai.FreeObject()` aus dem Speicher gelöscht werden. Bitte beachten Sie, dass Sie `Application.RemoveWindow` nicht für Dialoge aufrufen müssen, die mit `moai.FreeDialog()` freigegeben wurden, da `moai.FreeDialog()` bereits intern `Application.RemoveWindow` aufruft.

EINGABEN

`id` ID des Applications-Objekts

`window` ID des zu entfernenden Fensterobjekts

9.10 Application.ShowHelp**BEZEICHNUNG**

`Application.ShowHelp` – zeigt die Hilfedatei des Programmes an (V2.0)

ÜBERSICHT

```
moai.DoMethod(id, "ShowHelp", what$)
```

BESCHREIBUNG

Wenn Sie mit dem Attribut `Application.HelpFile` eine Hilfedatei für Ihr Programm definiert haben, können Sie diese Hilfedatei mit dieser Methode anzeigen. Sie müssen den Teil der Hilfedatei, der angezeigt werden soll, im Parameter `what$` übergeben. Dies kann einer der folgenden speziellen Werte sein:

Contents Zeigt das Inhaltsverzeichnis der Hilfedatei an.

Index	Zeigt den Index der Hilfedatei an.
Search	Zeigt die Suchseite der Hilfedatei an. Dies wird derzeit nur unter Windows unterstützt.

Um einen bestimmten Knoten der Hilfedatei anzuzeigen, verwenden Sie stattdessen die Methode `Application.ShowHelpNode`.

EINGABEN

id	ID des Application-Objekts
what\$	Teil der Hilfedatei, der angezeigt werden soll (siehe oben für mögliche Werte)

9.11 Application.ShowHelpNode**BEZEICHNUNG**

`Application.ShowHelpNode` – zeigt einen bestimmten Knoten in der Hilfedatei an (V2.0)

ÜBERSICHT

```
moai.DoMethod(id, "ShowHelpNode", node$)
```

BESCHREIBUNG

Wenn Sie mit dem Attribut `Application.HelpFile` eine Hilfedatei für Ihr Programm definiert haben, können Sie mit dieser Methode den durch `node$` angegebenen Knoten in dieser Hilfedatei anzeigen.

Um eines der Standardteile der Hilfedatei (wie das Inhaltsverzeichnis oder den Index) anstelle eines bestimmten Knotens anzuzeigen, verwenden Sie stattdessen die Methode `Application.ShowHelp`.

EINGABEN

id	ID des Application-Objekts
node\$	Knoten, der angezeigt werden soll

9.12 Application.Sleep**BEZEICHNUNG**

`Application.Sleep` – versetzt das Programm in den Ruhezustand

BESCHREIBUNG

Dieses Attribut versetzt das Programm in den Ruhezustand. Alle geöffneten Fenster werden deaktiviert und sie akzeptieren keine Benutzereingaben mehr. Dies kann nützlich sein, wenn Ihr Programm gerade Dateien lädt oder speichert, wodurch Benutzereingaben nicht möglich sind.

TYP

Boolesch

ANWENDBARKEIT

S

9.13 Application.UseIcons

BEZEICHNUNG

Application.UseIcons – setzt den Standardbildtyp auf Icon (V2.0)

BESCHREIBUNG

Dieses Attribut kann verwendet werden, um den Standardbildtyp für Attribute wie `Button.Icon` oder `Toolbarbutton.Icon` global auf Icon statt Pinsel zu ändern. Auch wenn die Namen dieser Attribute etwas anderes vermuten lassen, erwarten sie aus historischen Gründen standardmäßig einen Hollywood-Pinsel. Die Verwendung von Hollywood-Icons ist jedoch oft bequemer, da sie eine größere Flexibilität bieten, insbesondere wenn es um die Unterstützung hoher DPIs geht. Deshalb wird die Verwendung von Hollywood-Icons anstelle von Pinseln heutzutage von vielen GUI-Designern bevorzugt. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), [Seite 21](#), für Details.

Alternativ können Sie den Standardbildtyp auch pro Widget von Pinsel zu Icon ändern, indem Sie Attribute wie `Button.IconType` oder `Toolbarbutton.IconType` verwenden, aber global mit `Application.UseIcons` könnte bequemer sein.

TYP

Boolesch

ANWENDBARKEIT

I

9.14 Application.WindowMenu

BEZEICHNUNG

Application.WindowMenu – legt den Fenstermenütitel fest (V2.0)

PLATTFORMEN

macOS

BESCHREIBUNG

Legen Sie den Titel des Standardfenstermenüs von macOS auf einen benutzerdefinierten fest. Dies ist nützlich, wenn Sie Ihre Programme lokalisieren.

TYP

Zeichenkette

ANWENDBARKEIT

I

10 Area-Klasse (Bereich)

10.1 Übersicht

Area-Klasse (Bereich) ist die Oberklasse jeder MOAI-Klasse, die Objekte erzeugt, die eine visuelle Darstellung innerhalb eines Fensters haben. Als solches erlaubt es Ihnen, verschiedene generische Attribute wie Sichtbarkeit von Objekten (ein- oder ausgeblendet), Status (aktiviert oder deaktiviert), Tooltips und Kontextmenüs zu verwalten. Elemente, die von der Area-Klasse abgeleitet sind, werden gewöhnlich Widgets genannt (ein Kofferwort aus "window" und "gadget"). Die Area-Klasse kann Ihnen auch Informationen über die Position und Größe eines Widgets sowie andere Details liefern, die zu seiner visuellen Darstellung innerhalb eines Fensters gehören.

Beachten Sie, dass Objekte, die von der Group-Klasse (Group) abgeleitet werden, keine Elemente der Area-Klasse sind, da die Group-Klasse nur Gruppen von Widgets erstellt, aber kein Widget selbst ist, sondern nur ein Layout-Werkzeug.

10.2 Area.ContextMenu

BEZEICHNUNG

Area.ContextMenu – setzt das Kontextmenü für ein Objekt

BESCHREIBUNG

Setzen Sie dieses Attribut auf den Identifikator eines MOAI-Objekts, welches von der Menu-Klasse abgeleitet ist und somit erhält das entsprechende Objekt ein Kontextmenü. Wenn der Benutzer die rechte Maustaste auf das jeweilige Objekt drückt, wird das Kontextmenü angezeigt.

Beachten Sie, dass Kontextmenü-Ereignisse dann über den Standardmechanismus `MenuItem.Selected` und nicht über einen speziellen Kontextmenü-Ereignishandler geliefert werden. Da Sie dasselbe Menüobjekt für mehrere Widgets verwenden können, benötigen Sie eine Möglichkeit, das Widget herauszufinden, dessen Kontextmenü das Ereignis ausgelöst hat. Um Ihnen diese Informationen zu liefern, enthält die Ereignismeldung einen zusätzlichen Eintrag namens `Parent`, der die ID des Widgets enthält. Dadurch können Sie dasselbe Menüobjekt als Kontextmenü für mehrere übergeordnete Widgets verwenden.

Ab RapaGUI 2.0 können Sie auch eine Benachrichtigung über das Attribut einrichten. In diesem Fall wird Ihr Ereignis-Handler aufgerufen, unmittelbar bevor RapaGUI das Kontextmenü anzeigt. Sie können dann entweder `False` für kein Kontextmenü zurückgeben oder `moai.Set()` auf `Area.ContextMenu` verwenden, um das Kontextmenü festzulegen, das angezeigt werden soll. Dies ist sehr nützlich, um das Kontextmenü anzupassen, je nachdem, wo die rechte Maustaste geklickt wurde. Die Position wird an Ihren Ereignis-Handler weitergegeben. Falls Sie `Area.ContextMenu` überwachen, werden Ihrem Ereignis-Handler abhängig vom übergeordneten Widget des Kontextmenüs auch einige zusätzliche Informationen übergeben. Hier ist eine Übersicht über diese zusätzlichen Nachrichtfelder:

- X X-Koordinate des Mausklicks.
- Y Y-Koordinate des Mausklicks.

CursorPos	Cursor-Index an der Stelle, an der der Benutzer geklickt hat. Dies wird nur für Widgets des Typs Texteditor-Klasse gesetzt.
Row	Zeile an der Position des Mausklicks oder -1, wenn an der Klickposition keine Zeile vorhanden ist. Dies ist nur für Widgets vom Typ Listview-Klasse gesetzt.
Column	Spalte an der Position des Mausklicks oder -1, wenn an der Klickposition keine Spalte vorhanden ist. Dies ist nur für Widgets vom Typ Listview-Klasse gesetzt.
Item	Treeview-Element an der Position des Mausklicks oder -1, wenn sich kein Element an der Klickposition befindet. Dies ist nur für Widgets vom Typ Treeview-Klasse gesetzt.

AmigaOS-Benutzer beachten bitte, dass MUI die Freigabe von Kontextmenü-Objekten über Fenster hinweg nicht erlaubt. Es ist nicht möglich, dasselbe Menüobjekt mit Widgets in verschiedenen Fenstern zu verwenden. Es ist in Ordnung, ein Menüobjekt für mehrere Widgets im selben Fenster zu verwenden, aber nicht für Widgets in einem anderen Fenster. Alle anderen Plattformen haben diese Einschränkung nicht, nur AmigaOS ist hier betroffen.

Um ein Kontextmenü vollständig zu entfernen, übergeben Sie diesem Attribut den speziellen String "(none)".

Siehe [Abschnitt 3.16 \[Kontextmenüs\]](#), Seite 25, für ein Beispiel.

TYP

MOAI-Objekt

ANWENDBARKEIT

ISGN

10.3 Area.Disabled

BEZEICHNUNG

Area.Disabled – setzt oder gibt den Deaktivierstatus zurück

BESCHREIBUNG

Setzen Sie dieses Attribut, um ein Widget zu deaktivieren oder zu aktivieren. Deaktivierte Widgets akzeptieren keine Benutzereingaben mehr und sind optisch als deaktivierte Objekte erkennbar.

TYP

Boolesch

ANWENDBARKEIT

ISG

10.4 Area.FixHeight

BEZEICHNUNG

Area.FixHeight – fixiert die Höhe eines Objektes

BESCHREIBUNG

Setzen Sie dies auf True, um die vertikale Größenanpassung für dieses Objekt zu deaktivieren.

TYP

Boolesch

ANWENDBARKEIT

I

10.5 Area.FixWidth

BEZEICHNUNG

Area.FixWidth – fixiert die Breite eines Objektes

BESCHREIBUNG

Setzen Sie dies auf True, um die horizontale Größenänderung für dieses Objekt zu deaktivieren.

TYP

Boolesch

ANWENDBARKEIT

I

10.6 Area.FontName

BEZEICHNUNG

Area.FontName – setzt/ermittelt die Schriftart für das Objekt (V2.0)

BESCHREIBUNG

Setzt die Schriftart für das Objekt oder ermittelt sie.

Beachten Sie, dass dies auf AmigaOS und kompatiblen Systemen nur bei der Initialisierung eingestellt werden kann. Sie können die Schriftarten für bestehende Objekte nicht ändern. Außerdem müssen Sie auf AmigaOS auch `Area.FontSize` setzen, wenn Sie dieses Attribut setzen.

Beachten Sie auch, dass `Area.FontName` unter AmigaOS nicht mit Widgets vom Typ Label-Klasse verwendet werden kann. Verwenden Sie die Text-Klasse.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

10.7 Area.FontSize

BEZEICHNUNG

Area.FontSize – setzt/ermittelt die Schriftgröße für das Objekt (V2.0)

BESCHREIBUNG

Legen Sie die Schriftgröße für das Objekt fest oder rufen Sie sie ab. Die Schriftgröße muss entweder ein Wert in Punkten (Nicht-Amiga) oder in Pixeln (Amiga) oder eine der folgenden vordefinierten Schriftgrößen sein:

Normal Normale Schriftgröße.

Small Kleine Schriftgröße.

Big Große Schriftgröße.

Beachten Sie, dass dies auf AmigaOS und kompatiblen Systemen nur bei der Initialisierung eingestellt werden kann. Sie können die Schriftarten für bestehende Objekte nicht ändern. Außerdem müssen Sie unter AmigaOS auch **Area.FontName** setzen, wenn Sie dieses Attribut auf eine numerische Größe setzen. Wenn Sie unter AmigaOS eine der vordefinierten Schriftgrößen (siehe oben) übergeben, müssen Sie **Area.FontName** allerdings nicht setzen.

Beachten Sie auch, dass **Area.FontSize** unter AmigaOS nicht mit Widgets vom Typ Label-Klasse verwendet werden kann. Verwenden Sie die Text-Klasse.

TYP

Zahl oder Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

ISG

10.8 Area.FontStyle

BEZEICHNUNG

Area.FontStyle – setzt/ermittelt den Schriftstil für das Objekt (V2.0)

BESCHREIBUNG

Legen Sie den Schriftstil für das Objekt fest oder rufen Sie ihn ab. Dies kann eine Kombination der folgenden vordefinierten Stile sein:

Normal Normaler Schriftstil. Dieser kann nicht mit anderen Stilen kombiniert werden.

Bold Fetter Schriftstil.

Italic Kursiver Schriftstil.

Underlined
Unterstrichener Schriftstil.

Fixed Verwendet eine Schriftart mit fester Breite.

Mehrere Stile können kombiniert werden, indem ein Semikolon (;) als Trennzeichen verwendet wird.

TYP

Zeichenkette (mögliche Werte siehe oben)

ANWENDBARKEIT

ISG

10.9 Area.Height

BEZEICHNUNG

Area.Height – setzt/ermittelt die Höhe eines Objekts

BESCHREIBUNG

Setzen Sie dieses Attribut auf die gewünschte Objekthöhe in geräteunabhängigen Pixeln. Dies ist normalerweise nicht notwendig, da RapaGUI automatisch eine geeignete Größe für seine Objekte wählt. In manchen Fällen kann es jedoch sinnvoll sein, die Objektgröße fein abzustimmen.

Wenn das übergeordnete Fenster des Objekts geöffnet ist, können Sie auch die Höhe des Objekts über dieses Attribut auslesen.

TYP

Zahl

ANWENDBARKEIT

IG

10.10 Area.Hide

BEZEICHNUNG

Area.Hide – blendet ein Objekt ein oder aus

BESCHREIBUNG

Setzen Sie dieses Attribut, um ein Objekt ein- oder auszublenden.

TYP

Boolesch

ANWENDBARKEIT

ISG

10.11 Area.Left

BEZEICHNUNG

Area.Left – gibt die linke Kante des Objekts zurück

BESCHREIBUNG

Dieses Attribut gibt die x-Position eines Objekts innerhalb seines übergeordneten Fensters zurück. Dies gilt nur, wenn das Objekt aktuell sichtbar ist.

TYP

Zahl

ANWENDBARKEIT

G

10.12 Area.NoAutoKey

BEZEICHNUNG

Area.NoAutoKey – deaktiviert die automatische Tastaturkürzel-Generierung

BESCHREIBUNG

Setzen Sie dieses Attribut, um die automatische Tastaturkürzel-Generierung zu deaktivieren. Standardmäßig wird das Zeichen nach einem Unterstrich in einem Label oder einem Menüeintrag als Tastaturkürzel verwendet, mit der Sie über die Tastatur auf das Widget zugreifen können. Wenn Sie dieses Attribut setzen, werden Unterstrichzeichen niemals als Markierung für Tastaturkürzel verwendet und werden als normaler Text bei Beschriftungen (Label) und in Menüeinträgen angezeigt.

Siehe [Abschnitt 3.13 \[Tastaturkürzel\]](#), Seite 22, für Details.

TYP

Boolesch

ANWENDBARKEIT

I

10.13 Area.Redraw

BEZEICHNUNG

Area.Redraw – erzwingt die vollständige Neuzeichnung eines Widget

ÜBERSICHT

```
moai.DoMethod(id, "Redraw")
```

BESCHREIBUNG

Diese Methode zeichnet vollständig das Widget neu. Normalerweise ist dies nicht notwendig, da Widgets selbst entscheiden, wann sie neu gezeichnet werden müssen. Dennoch könnte es zu Testzwecken oder bei Verwendung eines von Scrollcanvas-Klasse (Bildlaufleinwand) abgeleiteten Widget nützlich sein.

EINGABEN

id	ID des Widget
----	---------------

10.14 Area.Tooltip

BEZEICHNUNG

Area.Tooltip – setzt/ermittelt die Zeichenkette des Tooltip

BESCHREIBUNG

Legen Sie eine Zeichenkette fest, die in einem Tooltip angezeigt werden soll, wenn Sie mit der Maus über das Objekt fahren.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

10.15 Area.Top

BEZEICHNUNG

Area.Top – gibt die obere Kante des Objekts zurück

BESCHREIBUNG

Dieses Attribut gibt die y-Position eines Objekts innerhalb seines übergeordneten Fensters zurück. Dies gilt nur, wenn das Objekt aktuell sichtbar ist.

TYP

Zahl

ANWENDBARKEIT

G

10.16 Area.Weight

BEZEICHNUNG

Area.Weight – setzt die Objektgröße

BESCHREIBUNG

Dieses Attribut setzt die Größe eines Objekts innerhalb einer Gruppe von Objekten. Standardmäßig haben alle Objekte einer Gruppe eine Größe von 100. Wenn Sie ein Objekt haben wollen, das doppelt so groß erscheint, müssen Sie ihm eine Größe von 200 geben. Um ein Objekt eineinhalb Mal so groß erscheinen zu lassen, verwenden Sie einfach 150 und so weiter.

Natürlich ist dieses Attribut nur für Objekte sinnvoll, die in der Größe veränderbar sind.

Beachten Sie, dass Sie beim Setzen von `Area.Weight` normalerweise keine spezifischen Abmessungen über `Area.Width` und `Area.Height` festlegen sollten. Die Breite und Höhe sollten stattdessen durch die Einstellung `Area.Weight` bestimmt werden.

TYP

Zahl

ANWENDBARKEIT

I

10.17 Area.Width

BEZEICHNUNG

Area.Width – setzt/ermittelt die Breite des Objekts

BESCHREIBUNG

Setzen Sie dieses Attribut auf die gewünschte Objektbreite in geräteunabhängigen Pixeln. Dies ist normalerweise nicht notwendig, da RapaGUI automatisch eine geeignete Größe für seine Objekte wählt. In manchen Fällen kann es jedoch sinnvoll sein, die Objektgröße fein abzustimmen.

Wenn das übergeordnete Fenster des Objekts geöffnet ist, können Sie auch die Breite des Objekts über dieses Attribut auslesen.

TYP

Zahl

ANWENDBARKEIT

IG

11 Busybar-Klasse (Beschäftigung)

11.1 Übersicht

Die Busybar-Klasse (Beschäftigung) leitet sich von Area-Klasse ab und erstellt ein Widget, welches eine Animation zeigt, womit der Benutzer erkennt, dass das Programm gerade beschäftigt ist. Diese Animation zeigt jedoch nur Aktivität, aber keinen Fortschritt. Wenn Sie den Fortschritt visualisieren möchten, verwenden Sie stattdessen die Progressbar-Klasse (Fortschrittsbalken). Siehe [Abschnitt 39.1 \[Progressbar-Klasse\]](#), Seite 215, für Details.

Sobald Sie ein Objekt der Busybar-Klasse erstellt haben, müssen Sie `Busybar.Move` wiederholt aufrufen, um die Animation zum Leben zu erwecken. Dies kann durch die Einrichtung eines Intervall-Timers mit Hilfe des Hollywood-Befehls `SetInterval()` erreicht werden.

11.2 Busybar.Move

BEZEICHNUNG

Busybar.Move – zeichnet das nächste Animationseinzelbild

ÜBERSICHT

```
moai.DoMethod(id, "Move")
```

BESCHREIBUNG

Sie müssen diese Methode wiederholt aufrufen, um das Widget zu animieren. Eine gute Idee ist es, einen Hollywood-Intervall-Timer mit `SetInterval()` zu installieren und dann diese Methode etwa 10 mal pro Sekunde aufzurufen.

EINGABEN

id ID des Busybar-Objekts

11.3 Busybar.Reset

BEZEICHNUNG

Busybar.Reset – stoppt die Animation

ÜBERSICHT

```
moai.DoMethod(id, "Reset")
```

BESCHREIBUNG

Diese Methode stoppt die Animation und setzt sie auf den meisten Plattformen zurück. Sukzessive Aufrufe von `Busybar.Move` beginnen normalerweise wieder ab dem ersten Einzelbild zu zeichnen.

EINGABEN

id ID des Busybar-Objekts

12 Button-Klasse (Schaltflächen)

12.1 Übersicht

Die Button-Klasse (Schaltflächen) leitet sich von der Area-Klasse ab und ermöglicht es Ihnen, sehr einfach Schaltflächen (Knöpfe, Tasten, Schalter) zu erstellen. Der Inhalt des XML-Tags wird als Schaltflächentext verwendet. Wenn der Schaltflächentext einen Unterstrich enthält, richtet RapaGUI automatisch das auf diesen Unterstrich folgende Zeichen als Tastaturkürzel ein. Wenn Sie dieses Verhalten nicht wünschen, setzen Sie `Area.NoAutoKey` auf `True`. Siehe [Abschnitt 3.13 \[Tastaturkürzel\]](#), [Seite 22](#), für Details.

Beispiel:

```
<button id="ok">OK</button>
```

12.2 Button.Hint

BEZEICHNUNG

`Button.Hint` – setzt den Schaltflächen-Hinweis (V2.0)

PLATTFORMEN

Android

BESCHREIBUNG

Wenn Sie Schaltflächen in Dialogen auf Android verwenden, können Sie einen Hinweiswert angeben, der Android mitteilt, ob die Schaltfläche positiv, negativ oder neutral ist. Dieser Hinweiswert kann mit diesem Attribut festgelegt werden. Sie können ihn auf die folgenden Werte setzen:

- `None` Kein Hinweis. Dies ist die Voreinstellung.
- `Positive` Schaltfläche ist die positive Schaltfläche in einem Dialog (z. B. "Ja").
- `Negative` Schaltfläche ist die negative Schaltfläche in einem Dialog (z. B. "Nein").
- `Neutral` Schaltfläche ist die neutrale Schaltfläche in einem Dialog (z. B. "Abbrechen").

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

I

12.3 Button.Icon

BEZEICHNUNG

`Button.Icon` – setzt das Bild der Schaltfläche

BESCHREIBUNG

Setzen Sie dieses Attribut auf den Identifikator eines Hollywood-Pinsels oder -Icons, um Ihrer Schaltfläche ein Bild hinzuzufügen. Ob dieses Attribut einen Hollywood-Pinsel

oder ein Icon erwartet, hängt davon ab, was Sie im Attribut `Button.IconType` angeben. Standardmäßig erwartet `Button.Icon` einen Hollywood-Pinsel. Sie können die Position des Icons in Bezug auf die Beschriftung der Schaltfläche festlegen, indem Sie das Attribut `Button.IconPos` setzen. Beachten Sie, dass RapaGUI das Bild möglicherweise skaliert, um es an die DPI-Einstellung des aktuellen Monitors anzupassen. Bitte lesen Sie das Kapitel über High-DPI-Unterstützung für weitere Informationen. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), Seite 21, für Details.

Bitte lesen Sie auch den Abschnitt Bilder-Cache von RapaGUI, um mehr über die Unterstützung von Symbolen in RapaGUI zu erfahren. Siehe [Abschnitt 3.20 \[Bilder-Cache\]](#), Seite 32, für Details.

Beachten Sie, dass unter AmigaOS und kompatiblen die Unterstützung von Symbolen nur mit MUI 4.0 oder höher verfügbar ist.

TYP

Zahl

ANWENDBARKEIT

I

12.4 Button.IconPos

BEZEICHNUNG

`Button.IconPos` – definiert die Position des Schaltflächen-Symbols

BESCHREIBUNG

Stellen Sie die gewünschte Position eines mit `Button.Icon` angegebenen Symbols ein. Dies kann einer der folgenden Werte sein:

Left Zeigt das Symbol links neben dem Text der Schaltfläche. Dies ist die Voreinstellung.

Right Zeigt das Symbol rechts neben dem Text der Schaltfläche.

Beachten Sie, dass die Unterstützung von Symbolen auf AmigaOS- und kompatiblen nur mit MUI 4.0 oder besser verfügbar ist.

TYP

Zeichenkette (mögliche Werte siehe oben)

ANWENDBARKEIT

I

12.5 Button.IconScale

BEZEICHNUNG

`Button.IconScale` – konfiguriert die automatische Bildskalierung (V2.0)

BESCHREIBUNG

Wenn `Button.Icon` auf einen Rasterpinsel gesetzt wurde und RapaGUI auf einem Bildschirm mit hohem DPI-Wert läuft, skaliert RapaGUI automatisch die Rastergrafiken des

Pinsels, um sie an die DPI-Einstellung des aktuellen Monitors anzupassen. Wenn Sie dies nicht möchten, setzen Sie dieses Tag auf `False`.

Alternativ können Sie die automatische Bildskalierung auch global deaktivieren, indem Sie bei der Präprozessor-Anweisung `@REQUIRE` mit RapaGUI den Tag `ScaleGUI` auf `False` setzen. Siehe [Abschnitt 3.4 \[Initialisierung von RapaGUI\]](#), Seite 11, für Details.

Bitte lesen Sie auch das Kapitel über die High-DPI-Unterstützung in RapaGUI, um mehr über die Unterstützung von High-DPI-Displays zu erfahren. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), Seite 21, für Details.

TYP

Boolesch

ANWENDBARKEIT

I

12.6 Button.IconType

BEZEICHNUNG

`Button.IconType` – setzt den zu verwendenden Icon-Typ (V2.0)

BESCHREIBUNG

Mit diesem Attribut können Sie den Typ des Hollywood-Bildobjekts festlegen, das in dem Attribut `Button.Icon` übergeben wird. Standardmäßig erwartet `Button.Icon` einen Hollywood-Pinsel. Durch das Setzen von `Button.IconType` können Sie jedoch einen anderen Hollywood-Bildtyp verwenden.

Die folgenden Bildtypen sind derzeit verfügbar:

- | | |
|--------------|--|
| Brush | Verwendet einen Hollywood-Pinsel. Dies ist der Standardtyp. Sie können entweder Raster- oder Vektorpinsel verwenden. Vektorpinsel haben den Vorteil, dass sie ohne Qualitätsverlust auf jede beliebige Auflösung skaliert werden können. Dies ist sehr nützlich, wenn Sie Programme entwerfen, die mit High-DPI-Monitoren kompatibel sein sollen. Siehe Abschnitt 3.12 [High-DPI-Unterstützung] , Seite 21, für Details. |
| Icon | Verwendet ein Hollywood-Icon. Dieser Bildtyp hat den Vorteil, dass er mehrere Teilbilder unterschiedlicher Größe enthalten kann. Dadurch ist es möglich, Bilder in unterschiedlichen Auflösungen bereitzustellen, was bei der Entwicklung von Programmen, die mit High-DPI-Monitoren kompatibel sein sollen, sehr nützlich sein kann. Siehe Abschnitt 3.12 [High-DPI-Unterstützung] , Seite 21, für Details. |

Beachten Sie, dass Sie den Standard aller `IconType`-Attribute global auf `Hollywood-Icon` ändern können, indem Sie den Tag `Application.UseIcons` setzen. Siehe [Abschnitt 9.13 \[Application.UseIcons\]](#), Seite 70, für Details.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

I

12.7 Button.Pressed

BEZEICHNUNG

Button.Pressed – benachrichtigt, ob eine Schaltfläche gedrückt wird

BESCHREIBUNG

Dieses Attribut wird ausgelöst, wenn der Benutzer die Schaltfläche drückt. RapaGUI überwacht automatisch dieses Attribut für alle Schaltflächen, so dass Sie nicht explizit eine Benachrichtigung mit dem Attribut MOAI.Notify anfordern müssen.

TYP

Boolesch

ANWENDBARKEIT

N

12.8 Button.Selected

BEZEICHNUNG

Button.Selected – Auswahlzustand erfahren/umschalten/benachrichtigen

BESCHREIBUNG

Benutzen Sie diese Option, um den Auswahlzustand eines Umschaltknopfes zu erfahren, umzuschalten oder sich über ein Benutzer-Umschalt-Ereignis benachrichtigen zu lassen. Wenn Sie dieses Attribut verwenden wollen, müssen Sie zuerst `Button.Toggle` auf `True` setzen.

RapaGUI überwacht dieses Attribut automatisch für alle Schaltflächen, sodass Sie nicht explizit eine Benachrichtigung mit dem Attribut MOAI.Notify anfordern müssen.

TYP

Boolesch

ANWENDBARKEIT

ISGN

12.9 Button.Text

BEZEICHNUNG

Button.Text – setzt/ermittelt den Schaltflächentext

BESCHREIBUNG

Mit diesem Attribut setzen oder holen Sie die Beschriftung der Schaltfläche.

TYP

Zeichenkette

ANWENDBARKEIT

SG

12.10 Button.Toggle

BEZEICHNUNG

Button.Toggle – erstellt einen Umschaltknopf

BESCHREIBUNG

Wenn Sie diesen Wert auf `True` setzen, wird ein Umschaltknopf erstellt. Umschaltknöpfe haben zwei Zustände: Ein und Aus. Sie können das Attribut `Button.Selected` verwenden, um den Zustand umzuschalten und sich über Zustandswechsel-Ereignisse informieren zu lassen.

Standardmäßig ist die Schaltfläche eine normale Schaltfläche.

TYP

Boolesch

ANWENDBARKEIT

I

13 Checkbox-Klasse (Auswahlkästchen)

13.1 Übersicht

Die Checkbox-Klasse ist von Area-Klasse abgeleitet und ermöglicht die Erstellung von Checkbox-Widgets. Eine Checkbox (Auswahlkästchen) ist eine beschriftete Box, die standardmäßig entweder aktiviert (Häkchen ist sichtbar) oder deaktiviert ist (kein Häkchen). Sie wird am häufigsten verwendet, um dem Benutzer das Umschalten zwischen Optionen zu ermöglichen. Der Inhalt des XML-Tags wird als Text für das Auswahlkästchen verwendet. Hier ist ein Beispiel, wie man ein Auswahlkästchen in XML erstellt:

```
<checkbox>Yes, I'm over 18 years old</checkbox>
```

13.2 Checkbox.Right

BEZEICHNUNG

Checkbox.Right – legt die Position des Auswahlkästchen-Textes fest

BESCHREIBUNG

Setzen Sie dies auf `True`, damit das Auswahlkästchen rechts neben dem zugehörigen Text erscheint. Standardmäßig wird das Auswahlkästchen links neben dem Text angezeigt.

TYP

Boolesch

ANWENDBARKEIT

I

13.3 Checkbox.Selected

BEZEICHNUNG

Checkbox.Selected – setzt oder ermittelt den das Auswahlkästchen-Status

BESCHREIBUNG

Setzt oder ermittelt den ausgewählten Zustand eines Checkbox-Objekts.

Sie können auch eine Benachrichtigung über dieses Attribut einrichten, um zu erfahren, dass der Benutzer den Status des Auswahlkästchen umschaltet.

TYP

Boolesch

ANWENDBARKEIT

ISGN

14 Choice-Klasse (Auswahl)

14.1 Übersicht

Die Choice-Klasse (Auswahl) leitet sich von der Area-Klasse ab und erstellt Widgets, die es dem Benutzer ermöglichen, einen Eintrag aus einer vordefinierten Liste von Elementen auszuwählen. Nur das ausgewählte Element ist sichtbar, bis der Benutzer die Auswahlliste herunterzieht.

Wenn Sie ein Auswahl-Widget definieren, können Sie den Tag `<item>` verwenden, um das Auswahl-Widget mit Elementen zu füllen. Hier ist ein XML-Beispielauszug zur Erstellung eines Auswahl-Widget mit drei Einträgen:

```
<choice id="printer">
  <item>HP Deskjet</item>
  <item>NEC P6</item>
  <item>Okimate 20</item>
</choice>
```

Alternativ können Sie auch ein leeres Auswahl-Widgets erstellen und später mit der Methode `Choice.Insert` mit Einträgen füllen. So erstellen Sie ein leeres Auswahlwidget:

```
<choice/>
```

14.2 Choice.Active

BEZEICHNUNG

`Choice.Active` – setzt oder ermittelt die aktive Auswahl

BESCHREIBUNG

Setzt oder ermittelt den aktive Eintrag im Auswahl-Widget von Index 0 für den ersten Eintrag bis zur Anzahl der Einträge -1 für den letzten Eintrag.

Sie können auch eine Benachrichtigung für dieses Attribut einrichten, um zu erfahren, wenn der Benutzer die aktive Auswahlmöglichkeit ändert.

Sie können auch die speziellen Werte `Next` oder `Prev` übergeben, um durch die Einträge des Auswahl-Widget zu blättern.

TYP

Zahl oder Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

ISGN

14.3 Choice.Clear

BEZEICHNUNG

`Choice.Clear` – löscht alle Einträge (V1.1)

ÜBERSICHT

```
moai.DoMethod(id, "Clear")
```

BESCHREIBUNG

Entfernt alle Einträge aus dem vom Auswahl-Widgets.

Beachten Sie, dass diese Methode unter AmigaOS und kompatiblen Betriebssystemen nur unter MUI 4.0 oder höher verfügbar ist.

EINGABEN

`id` ID des Auswahl-Widgets

14.4 Choice.Count**BEZEICHNUNG**

`Choice.Count` – ruft die Anzahl der Einträge im Auswahl-Widget ab (V1.1)

BESCHREIBUNG

Gibt die aktuelle Anzahl der Einträge im Auswahl-Widget zurück.

TYP

Zahl

ANWENDBARKEIT

G

14.5 Choice.GetEntry**BEZEICHNUNG**

`Choice.GetEntry` – gibt einen Eintrag zurück (V1.1)

ÜBERSICHT

```
e$ = moai.DoMethod(id, "GetEntry", pos)
```

BESCHREIBUNG

Diese Methode gibt Ihnen den in `pos` angegebenen Eintrag von einem Auswahl-Widget zurück. Sie können entweder einen absoluten Index oder den speziellen Wert `Active` in `pos` angeben, um den aktiven Eintrag zu erhalten. `Choice.GetEntry` gibt dann den Eintrag als Zeichenkette zurück.

EINGABEN

`id` ID des Auswahl-Widget

`pos` Index des Eintrags oder "Aktiv"

RÜCKGABEWERTE

`e$` Eintrag vom angegebenen Index

14.6 Choice.Insert

BEZEICHNUNG

Choice.Insert – fügt neuen Eintrag ein (V1.1)

ÜBERSICHT

`moai.DoMethod(id, "Insert", pos, e$)`

BESCHREIBUNG

Diese Methode fügt den in `e$` angegebenen Eintrag in das Auswahl-Widget ein. Die Einfügeposition wird im Argument `pos` angegeben und der neue Eintrag wird vor dem in `pos` angegebenen Eintrag eingefügt. Dies kann eine absolute Indexposition ab 0 für den ersten Eintrag oder einer der folgenden Sonderwerte sein:

Top Als ersten Eintrag einfügen.

Active Wird vor dem aktiven Eintrag eingefügt. Wenn es keinen aktiven Eintrag gibt, wird der Eintrag ganz oben in die Eintragsliste eingefügt.

Bottom Als letzten Eintrag einfügen.

Wenn `pos` größer oder gleich der Anzahl der Einträge im Auswahl-Widget ist, wird der Eintrag als letzter Eintrag eingefügt.

Beachten Sie, dass diese Methode unter AmigaOS und kompatiblen Betriebssystemen nur unter MUI 4.0 oder höher verfügbar ist.

EINGABEN

`id` ID des Auswahl-Widget

`pos` Position als Absolutwert oder Sonderwert einfügen (siehe oben)

`e$` Eintrag, der eingefügt werden soll

14.7 Choice.Remove

BEZEICHNUNG

Choice.Remove – entfernt einen Eintrag aus dem Auswahl-Widget (V1.1)

ÜBERSICHT

`moai.DoMethod(id, "Remove", pos)`

BESCHREIBUNG

Entfernt einen Eintrag aus einem Auswahl-Widget. Die Position kann als absoluter Indexwert oder als einer der folgenden Sonderwerte angegeben werden:

First Entfernt den ersten Eintrag.

Active Entfernt den aktiven Eintrag.

Last Entfernt den letzten Eintrag.

Wenn der aktive Eintrag entfernt wird, wird der folgende Eintrag aktiv.

Beachten Sie, dass diese Methode unter AmigaOS und kompatiblen Betriebssystemen nur unter MUI 4.0 oder höher verfügbar ist.

EINGABEN

<code>id</code>	ID des Auswahl-Widget
<code>pos</code>	Index des zu entfernenden Eintrags oder einer der speziellen Werte (siehe oben)

14.8 Choice.Rename

BEZEICHNUNG

Choice.Rename – benennt einen Eintrag um (V1.1)

ÜBERSICHT

```
moai.DoMethod(id, "Rename", pos, newname$)
```

BESCHREIBUNG

Benennt den Auswahleintrag an der in `pos` angegebenen Position in den in `newname$` genannten Namen um. `pos` kann eine absolute Indexposition ab 0 für den ersten Eintrag oder einer der folgenden Sonderwerte sein:

Active Benennt den aktiven Eintrag um.

Beachten Sie, dass diese Methode unter AmigaOS und kompatiblen Betriebssystemen nur unter MUI 4.0 oder höher verfügbar ist.

EINGABEN

<code>id</code>	ID des Auswahl-Widget
<code>pos</code>	Index des Eintrags, der umbenannt wird oder einer der speziellen Werte (siehe oben)
<code>newname\$</code>	Neuer Name des Eintrags

15 Combobox-Klasse (Textlisten)

15.1 Übersicht

Die Combobox-Klasse (Textlisten) leitet sich von der Area-Klasse ab und erzeugt ein Widget, das eine Kombination aus einer Liste und einem Text-Widget ist. Der Benutzer kann entweder einen Eintrag aus einer vordefinierten Liste von Einträgen auswählen oder einzelne Daten in das Text-Widget eingeben.

Wenn Sie ein Combobox-Objekt erstellen, können Sie den Tag `<item>` verwenden, um es mit Einträgen zu füllen. Hier ist ein Beispiel:

```
<combobox>
  <item>The</item>
  <item>quick</item>
  <item>brown</item>
  <item>fox</item>
  <item>jumps</item>
  <item>over</item>
  <item>the</item>
  <item>lazy</item>
  <item>dog</item>
</combobox>
```

Sie können auch eine leere Textliste erstellen und diese später mit Einträgen füllen, indem Sie die Methode `Combobox.Insert` verwenden. So erstellen Sie eine leere Textliste:

```
<combobox/>
```

15.2 Combobox.Acknowledge

BEZEICHNUNG

Combobox.Acknowledge – benachrichtigt, wenn der Benutzer RETURN drückt (V2.0)

BESCHREIBUNG

Immer wenn der Benutzer auf Return drückt, wird dieses Attribut auf `True` gesetzt. Sie können diese Benachrichtigung überwachen und die entsprechenden Maßnahmen ergreifen.

TYP

Boolesch

ANWENDBARKEIT

N

15.3 Combobox.Clear

BEZEICHNUNG

Combobox.Clear – entfernt alle Textlisten-Einträge (V1.1)

ÜBERSICHT

```
moai.DoMethod(id, "Clear")
```

BESCHREIBUNG

Entfernt alle Einträge aus der in `id` angegebene Textliste.

EINGABEN

`id` ID des Combobox-Objekts

15.4 Combobox.Close**BEZEICHNUNG**

Combobox.Close – schließt die Liste der Combobox (V2.0)

ÜBERSICHT

```
moai.DoMethod(id, "Close")
```

BESCHREIBUNG

Schließt die der Combobox zugeordnete Liste der vordefinierten Werte.

EINGABEN

`id` Identifikator des Combobox-Objekts

15.5 Combobox.Count**BEZEICHNUNG**

Combobox.Count – gibt die Anzahl der Einträge in der Textliste zurück (V1.1)

BESCHREIBUNG

Gibt die aktuelle Anzahl der Einträge in der Textliste zurück.

TYP

Zahl

ANWENDBARKEIT

G

15.6 Combobox.GetEntry**BEZEICHNUNG**

Combobox.GetEntry – gibt den Textlisten-Eintrag zurück (V1.1)

ÜBERSICHT

```
e$ = moai.DoMethod(id, "GetEntry", pos)
```

BESCHREIBUNG

Diese Methode gibt den in Position `pos` angegebenen Eintrag aus dem Textlisten-Widget zurück.

EINGABEN

`id` ID des Combobox-Objekts
`pos` Index des Eintrags

RÜCKGABEWERTE

`e$` Eintrag vom angegebenen Index

15.7 Combobox.Insert

BEZEICHNUNG

Combobox.Insert – fügt einen neuen Eintrag ein (V1.1)

ÜBERSICHT

```
moai.DoMethod(id, "Insert", pos, e$)
```

BESCHREIBUNG

Mit dieser Methode fügen Sie den in `e$` angegebenen Eintrag in die Textliste ein. Die Einfügeposition wird im Argument `pos` angegeben und der neue Eintrag wird vor dem in `pos` angegebenen Eintrag eingefügt. Dies kann eine absolute Indexposition ab 0 für den ersten Eintrag oder einer der folgenden Sonderwerte sein:

`Top` Als ersten Eintrag einfügen.
`Bottom` Als letzten Eintrag einfügen.

Wenn `pos` größer oder gleich der Anzahl der Einträge in der Textliste ist, wird der Eintrag als letzter Eintrag eingefügt.

EINGABEN

`id` ID des Combobox-Objekts
`pos` fügt die Position als Absolut- oder Sonderwert ein (siehe oben)
`e$` Eintrag, der eingefügt werden soll

15.8 Combobox.Open

BEZEICHNUNG

Combobox.Open – öffnet die Liste der Combobox (V2.0)

ÜBERSICHT

```
moai.DoMethod(id, "Open")
```

BESCHREIBUNG

Zeigt die zugehörige Liste der vordefinierten Werte der Combobox an.

EINGABEN

`id` Identifikator des Combobox-Objekts

15.9 Combobox.Popup

BEZEICHNUNG

Combobox.Popup – benachrichtigt, wenn Combobox-Listen geöffnet oder geschlossen werden (V2.0)

BESCHREIBUNG

Immer wenn der Benutzer die Liste der Combobox öffnet oder schließt, löst dieses Attribut eine Benachrichtigung aus. Der `TriggerValue` enthält den aktuellen Status der Popup-Liste, d.h. `True` für offen und `False` für geschlossen.

TYP

Boolesch

ANWENDBARKEIT

N

15.10 Combobox.Remove

BEZEICHNUNG

Combobox.Remove – entfernt einen Eintrag aus der Textliste (V1.1)

ÜBERSICHT

```
moai.DoMethod(id, "Remove", pos)
```

BESCHREIBUNG

Diese Methode entfernt der an der in `pos` angegebene Eintrag aus der Textliste.

EINGABEN

<code>id</code>	ID des Combobox-Objekts
<code>pos</code>	Index des Eintrags, der entfernt wird

15.11 Combobox.Rename

BEZEICHNUNG

Combobox.Rename – benennt einen Eintrag um (V1.1)

ÜBERSICHT

```
moai.DoMethod(id, "Rename", pos, newname$)
```

BESCHREIBUNG

Hiermit benennen Sie den Textlisten-Eintrag an der angegebenen Position in den in `newname$` angegebenen Namen um. Die Eingabeposition wird im Argument `pos` angegeben. Dies muss eine absolute Indexposition sein, die beim ersten Eintrag mit 0 beginnt.

EINGABEN

<code>id</code>	ID des Combobox-Objekts
<code>pos</code>	Eingabeposition als absolute Zahl
<code>newname\$</code>	Neuer Name für den Textlisten-Eintrag

15.12 Combobox.Selected

BEZEICHNUNG

Combobox.Selected – benachrichtigt bei einer Auswähländerung (V2.0)

BESCHREIBUNG

Immer wenn der Benutzer ein Element aus der Liste der Combobox auswählt, wird dieses Attribut ausgelöst.

TYP

Boolesch

ANWENDBARKEIT

N

15.13 Combobox.Value

BEZEICHNUNG

Combobox.Value – setzt oder ermittelt den aktuellen Inhalt der Textliste

BESCHREIBUNG

Holt oder setzt den aktuellen Inhalt der Textliste.

Sie können auch eine Benachrichtigung für dieses Attribut einrichten, die ausgelöst wird, sobald sich der Inhalt des Combobox-Objekts ändert.

TYP

Zeichenkette

ANWENDBARKEIT

ISGN

16 Dialog-Klasse

16.1 Übersicht

Die Dialog-Klasse leitet sich von Window-Klasse ab (Fenster) und erzeugt modale (oder nicht-modale) Dialoge. Modale Dialoge sind spezielle Top-Level-Fenster, die den Rest des Programmes blockieren, bis sie geschlossen werden. Dialoge werden typischerweise verwendet, wenn Benutzeraktionen erforderlich sind, um mit dem Programm fortzufahren oder um anzuzeigen, dass das Programm gerade beschäftigt ist. Ein Dialog könnte dann z.B. einen Fortschrittsbalken anzeigen.

Wie bei Fenstern muss das Wurzelement eines Dialogs immer ein einzelnes Gruppenobjekt sein, d.h. eine Instanz der Group-Klasse (Gruppen). Siehe [Abschnitt 18.1 \[Group-Klasse\]](#), [Seite 109](#), für Details. Darüber hinaus ist es nicht erlaubt, mehrere Elemente auf der Stammebene des Dialogs zu haben. Sie müssen nur ein einzelnes Gruppenobjekt als Wurzelement verwenden. Hier ein Beispiel für einen einfachen Fortschrittsdialog in XML:

```
<dialog id="mydlg" title="Working...">
  <vgroup>
    <progressbar id="prg"/>
    <button id="cancel">Cancel</button>
  </vgroup>
</dialog>
```

Sie sollten Dialoge nur dann erstellen, wenn Sie sie brauchen, und sie aus dem Speicher löschen, sobald Sie damit fertig sind. Es wird nicht empfohlen, alle Ihre Dialoge beim Start mit `moai.CreateApp()` zu erstellen und sie ständig im Speicher zu behalten. Stattdessen sollten Sie `moai.CreateDialog()` verwenden, um einen Dialog zu erstellen, wenn Sie ihn benötigen und ihn dann anschliessend aus dem Speicher löschen. Der Grund dafür ist, dass Fenster auf einigen von RapaGUI unterstützten Betriebssystemen eine begrenzte Ressource sind. Zum Beispiel gibt es unter Windows ein Limit von ungefähr 10.000 Fenstern pro Prozess. Das klingt vielleicht ausreichend genug, aber bedenken Sie, dass unter Windows jedes Widget ein "Fenster" für das Betriebssystem ist, z.B. alle Beschriftungen, Schaltflächen, Rahmen, Auswahlkästchen, Gruppe, etc. in Ihrem Programm ist ein Fenster. So sollten Sie darauf achten, dass Sie Ihre Dialoge nur bei Bedarf mit `moai.CreateDialog()` erstellen und anschliessend direkt wieder löschen.

In der Praxis empfiehlt es sich, für jeden Ihrer Dialoge eine eigene XML-Datei zu erstellen und dann `moai.CreateDialog()` zu verwenden, um die XML-Datei zur Laufzeit in einen Dialog zu konvertieren und anschliessend den Dialog automatisch von RapaGUI löschen zu lassen, sobald der Benutzer ihn schließt.

Als Unterklasse der Window-Klasse können die meisten Attribute und Methoden der Window-Klasse auch mit der Dialog-Klasse verwendet werden. Siehe [Abschnitt 62.1 \[Window-Klasse\]](#), [Seite 319](#), für Details. Beachten Sie jedoch, dass modale Dialoge nicht durch Setzen von `Window.Open` geöffnet werden dürfen, sondern durch die Ausführung der Methode `Dialog.ShowModal`. Um einen modalen Dialog zu schließen, rufen Sie einfach die Methode `Dialog.EndModal` auf. Dies löscht auch automatisch den Dialog aus dem Speicher, d.h. es wird implizit `moai.FreeDialog()` auf den Dialog aufgerufen, es sei denn, Sie fordern ausdrücklich, dass der Dialog durch Setzen eines optionalen Arguments auf `True` nicht gelöscht werden soll.

In ähnlicher Weise wird der Dialog auch automatisch gelöscht, wenn der Benutzer auf das Schließsymbol eines Dialogs klickt. Wenn Sie dies nicht möchten oder wenn Sie das Verhalten des Programmes für das Schließsymbol anpassen müssen, brauchen Sie eine Überwachung mit dem Attribut `Window.CloseRequest`. Wenn für dieses Attribut keine Überwachung vorhanden ist, ruft RapaGUI einfach `Dialog.EndModal` mit dem Parameter 0 auf, sobald auf das Schließsymbol des Dialogs geklickt wird.

Wenn Sie einen nicht-modalen Dialog öffnen möchten, d.h. einen Dialog, der Ihr Skript nicht blockiert, während es geöffnet ist, dürfen Sie nicht `Dialog.ShowModal` verwenden, sondern stattdessen `Window.Open` für den Dialog auf `True` setzen (und eventuell auch `Application.Sleep` auf `True` setzen, wenn Sie das Programm in eine Art Ruhezustand versetzen möchten, während der Dialog geöffnet ist). Das eigentliche Verhalten des Dialogs implementieren Sie (z.B. ein ständig aktualisierter Fortschrittsbalken), indem Sie wiederholt einen Verwaltungsbefehl aufrufen, entweder mit `SetTimeout()` oder des neuen Befehls `RunCallback()` von Hollywood 9.0. Wenn Sie mit einem nicht-modalen Dialog fertig sind, setzen Sie `Window.Open` darauf auf `False`. Verwenden Sie nicht `Dialog.EndModal` mit nicht-modalen Dialogen! Ein Beispiel für einen nicht-modalen Dialog finden Sie im Beispiel "Dialoge", das mit RapaGUI geliefert wird (wählen Sie "Test progress bar dialog" im Menü).

16.2 Dialog.EndModal

BEZEICHNUNG

`Dialog.EndModal` – schließt den Dialog und beendet die Modal-Schleife

ÜBERSICHT

```
moai.DoMethod(id, "EndModal", retval[, nodestroy])
```

BESCHREIBUNG

Diese Methode schließt den angegebenen Dialog, der mit `Dialog.ShowModal` geöffnet wurde und bricht seine Modalschleife ab. Der Wert, den Sie in `retval` übergeben, ist dann der Rückgabewert des Aufrufs der Methode `Dialog.ShowModal` dieses Dialogs. Dieser Wert wird oft verwendet, um Erfolg oder Misserfolg anzuzeigen, d.h. Sie können `True` zurückgeben, wenn der Benutzer die Taste "OK" gedrückt hat, andernfalls `False`. Zusätzlich löscht automatisch diese Methode den Dialog aus dem Speicher, d.h. sie ruft implizit den Befehl `moai.FreeDialog()` für dieses Dialogobjekt auf. Wenn Sie dies nicht wollen, müssen Sie das optionale Argument `nodestroy` auf `True` setzen. Es wird jedoch empfohlen und es ist eine gute Programmierpraxis, jeden Dialog aus dem Speicher zu löschen, sobald Sie damit fertig sind. Siehe [Abschnitt 16.1 \[Dialog-Klasse\]](#), Seite 99, für Details.

EINGABEN

<code>id</code>	ID des Dialogobjekts
<code>retval</code>	gewünschter Rückgabewert für <code>Dialog.ShowModal</code>
<code>nodestroy</code>	optional: <code>True</code> , wenn diese Methode den Dialog nicht automatisch löschen soll (Standard ist <code>False</code> , was bedeutet, dass der Dialog nach dem Schließen aus dem Speicher gelöscht wird).

16.3 Dialog.ShowModal

BEZEICHNUNG

Dialog.ShowModal – öffnet einen Dialog und startet die Modal-Schleife

ÜBERSICHT

```
retval = moai.DoMethod(id, "ShowModal")
```

BESCHREIBUNG

Diese Methode öffnet das Dialogfenster und startet eine modale Schleife, die den Rest des Programmes blockiert und auf das Schließen des Dialogs wartet, bevor die Steuerung an das Skript zurückgegeben wird. Mit anderen Worten: `Dialog.ShowModal` kehrt erst zurück, wenn der Benutzer das Dialogfenster schließt oder das Skript `Dialog.EndModal` für das Dialogfenster aufruft.

Wenn Sie einen nicht-modalen Dialog öffnen möchten, d.h. einen Dialog, der Ihr Skript nicht blockiert, während es geöffnet ist, dürfen Sie nicht `Dialog.ShowModal` verwenden, sondern müssen stattdessen `Window.Open` für den Dialog auf `True` setzen (und möglicherweise auch `Application.Sleep` auf `True` setzen, wenn Sie das Programm in eine Art Ruhezustand versetzen möchten, während der Dialog geöffnet ist). Das eigentliche Verhalten des Dialogs implementieren Sie (z.B. ein ständig aktualisierter Fortschrittsbalken), indem Sie wiederholt einen Verwaltungsbefehl aufrufen, entweder mit `SetTimeout()` oder des neuen Befehls `RunCallback()` von Hollywood 9.0. Wenn Sie mit einem nicht-modalen Dialog fertig sind, setzen Sie `Window.Open` darauf auf `False`. Verwenden Sie nicht `Dialog.EndModal` mit nicht-modalen Dialogen!

Beachten Sie, dass es eine gute Programmierpraxis ist, Dialoge nach Bedarf zu erstellen und zu löschen. Sie sollten sie nicht für die gesamte Lebensdauer Ihres Programmes im Speicher behalten, sondern nur bei Bedarf erstellen und löschen, sobald Sie damit fertig sind. Siehe [Abschnitt 16.1 \[Dialog-Klasse\], Seite 99](#), für Details.

Wenn Sie benachrichtigt werden möchten, wenn der Benutzer auf das Schließsymbol eines Dialogs klickt, müssen Sie eine Überwachung mit dem Attribut `Window.CloseRequest` installieren. Wenn es keine Überwachung auf diesem Attribut gibt, ruft RapaGUI beim Klick auf das Schließsymbol einfach `Dialog.EndModal` mit 0 als Parameter auf.

EINGABEN

<code>id</code>	ID des Dialogobjekts
<code>dlgfunc</code>	optional: Funktion, die den Dialog verwaltet
<code>userdata</code>	optional: Benutzerdaten, welche an die Dialogfunktion übergeben werden

RÜCKGABEWERTE

<code>retval</code>	Rückgabewert des Dialogs
---------------------	--------------------------

BEISPIEL

Siehe `@command{moai.CreateObject()}`

17 Finddialog-Klasse (Systemsuchdialog)

17.1 Übersicht

Die Finddialog-Klasse (Systemsuchdialog) leitet sich von der Dialog-Klasse ab und erzeugt einen Standard-Systemsuchdialog. Suchdialoge werden normalerweise in Verbindung mit der Texteditor-Klasse verwendet, um dem Benutzer zu ermöglichen, den gesamten Inhalt eines Texteditor-Widgets zu durchsuchen.

Obwohl die Finddialog-Klasse eine Unterklasse von der Dialog-Klasse ist, müssen die von ihr erstellten Dialoge immer nicht-modal sein. Es ist also weder erlaubt, sie mit `Dialog.ShowModal` anzuzeigen, noch mit `Dialog.EndModal` zu schließen. Stattdessen müssen Dialoge, die von der Finddialog-Klasse erstellt wurden, immer durch Setzen des Attributs `Window.Open` ein- und ausgeblendet werden.

Die Finddialog-Klasse ist seit RapaGUI 2.0 verfügbar.

Beachten Sie, dass diese Klasse derzeit auf AmigaOS (und kompatiblen Systemen) und Android nicht unterstützt wird.

17.2 Finddialog.Down

BEZEICHNUNG

`Finddialog.Down` – setzt/ermittelt die Suchrichtung (V2.0)

BESCHREIBUNG

Setzt oder ermittelt die Suchrichtung (`True` für die Suche vorwärts/nach unten, `False` für die Suche rückwärts/nach oben).

TYP

Boolesch

ANWENDBARKEIT

ISG

17.3 Finddialog.Find

BEZEICHNUNG

`Finddialog.Find` – benachrichtigt, wenn die Schaltfläche Suchen gedrückt wird (V2.0)

BESCHREIBUNG

Richtet eine Benachrichtigung über dieses Attribut ein, um zu erfahren, wann die Schaltfläche Suchen des Dialogs gedrückt wird.

TYP

Boolesch

ANWENDBARKEIT

N

17.4 Finddialog.FindNext

BEZEICHNUNG

Finddialog.FindNext – benachrichtigt, wenn die Schaltfläche Weitersuchen gedrückt wird (V2.0)

BESCHREIBUNG

Richtet eine Benachrichtigung über dieses Attribut ein, um zu erfahren, wann die Schaltfläche Weitersuchen des Dialogs gedrückt wird.

TYP

Boolesch

ANWENDBARKEIT

N

17.5 Finddialog.FindString

BEZEICHNUNG

Finddialog.FindString – setzt/ermittelt die aktuelle Suchzeichenkette (V2.0)

BESCHREIBUNG

Hiermit wird die aktuelle Suchzeichenkette gesetzt oder ermittelt.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

17.6 Finddialog.MatchCase

BEZEICHNUNG

Finddialog.MatchCase – setzt/ermittelt die Einstellung der Groß-/Kleinschreibung (V2.0)

BESCHREIBUNG

Setzt oder ermittelt die Einstellung der Groß-/Kleinschreibung im Dialog.

TYP

Boolesch

ANWENDBARKEIT

ISG

17.7 Finddialog.NoMatchCase

BEZEICHNUNG

Finddialog.NoMatchCase – sucht ohne Berücksichtigung der Groß- und Kleinschreibung (V2.0)

BESCHREIBUNG

Wenn Sie dieses Attribut auf `True` setzen, lässt das Dialogfeld die Suche nach Groß-/Kleinschreibung nicht zu.

TYP

Boolesch

ANWENDBARKEIT

I

17.8 Finddialog.NoUpDown

BEZEICHNUNG

Finddialog.NoUpDown – verhindert die Suchrichtung zu ändern (V2.0)

BESCHREIBUNG

Wenn Sie dieses Attribut auf `True` setzen, lässt der Dialog das Ändern der Suchrichtung nicht zu.

TYP

Boolesch

ANWENDBARKEIT

I

17.9 Finddialog.NoWholeWord

BEZEICHNUNG

Finddialog.NoWholeWord – verhindert nach ganzen Wörtern zu suchen (V2.0)

BESCHREIBUNG

Wenn Sie dieses Attribut auf `True` setzen, lässt der Dialog die Suche nach ganzen Wörtern nicht zu.

TYP

Boolesch

ANWENDBARKEIT

I

17.10 Finddialog.Replace

BEZEICHNUNG

Finddialog.Replace – benachrichtigt, wenn die Schaltfläche Ersetzen gedrückt wird (V2.0)

BESCHREIBUNG

Richtet eine Benachrichtigung für dieses Attribut ein, um zu erfahren, wann die Ersetzen-Schaltfläche des Dialogfelds gedrückt wird.

TYP

Boolesch

ANWENDBARKEIT

N

17.11 Finddialog.ReplaceAll

BEZEICHNUNG

Finddialog.ReplaceAll – benachrichtigt, wenn die Schaltfläche Alle ersetzen gedrückt wird (V2.0)

BESCHREIBUNG

Richtet eine Benachrichtigung für dieses Attribut ein, um zu erfahren, wenn die Schaltfläche Alle ersetzen des Dialogfelds gedrückt wird.

TYP

Boolesch

ANWENDBARKEIT

N

17.12 Finddialog.ReplaceMode

BEZEICHNUNG

Finddialog.ReplaceMode – setzt den Suchdialog in den Ersetzungsmodus (V2.0)

BESCHREIBUNG

Erstellt einen Ersetzungsdialog anstelle eines Suchdialogs.

TYP

Boolesch

ANWENDBARKEIT

I

17.13 Finddialog.ReplaceString

BEZEICHNUNG

Finddialog.ReplaceString – setzt/ermittelt die aktuelle Ersetzungszeichenkette (V2.0)

BESCHREIBUNG

Hiermit setzen Sie die aktuelle Ersetzungszeichenkette oder ermitteln sie.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

17.14 Finddialog.WholeWord

BEZEICHNUNG

Finddialog.WholeWord – setzt/ermittelt die Ganzwortsuche (V2.0)

BESCHREIBUNG

Setzt/ermittelt das Flag für die Suche nach ganzen Wörtern im Suchdialog.

TYP

Boolesch

ANWENDBARKEIT

ISG

18 Group-Klasse (Gruppen)

18.1 Übersicht

Die Group-Klasse (Gruppen) kann verwendet werden, um eine Reihe von Elementen auf verschiedene Weise zu gestalten. Wenn sich die Größe des übergeordneten Objekts ändert, passen sich dementsprechend die Größe ihrer Elementen an, was sehr nützlich sein kann, um in alle Richtungen frei skalierbare Fenster zu erstellen.

Die folgenden verschiedenen Gruppentypen werden von RapaGUI unterstützt:

`<hgroup>` Gruppenelemente werden in einer Reihe (horizontal, waagrecht) angeordnet.

`<vgroup>` Gruppenelemente werden in einer Spalte (vertikal, senkrecht) angeordnet.

`<colgroup>`

Gruppenelemente werden in mehreren Spalten angeordnet.

`<scrollgroup>`

Eine Gruppe mit Bildlaufleiste. Dies ist eine spezielle Gruppe, die eine eingebettete Gruppe mit Bildlaufleiste anzeigt. Siehe [Abschnitt 44.1 \[Scrollgroup-Klasse\]](#), [Seite 233](#), für Details.

Spaltengruppen sind nützlich, wenn Sie identische Widget-Größen für alle Ihre Elemente in einer Gruppe benötigen, um ein angenehmeres Erscheinungsbild zu erhalten. Stellen Sie sich zum Beispiel ein Formular vor, das aus Texteingabe-Widgets und Textobjekten besteht. Es wird empfohlen, hier ein `<colgroup>` zu verwenden, da es zu einem klaren und geordneten Erscheinungsbild führt. Hier ist ein Beispiel:

```
<colgroup columns="2">
  <label>Name</label>
  <textentry/>
  <label>Street</label>
  <textentry/>
  <label>City</label>
  <textentry/>
  <label>Zip code</label>
  <textentry/>
  <label>Country</label>
  <textentry/>
  <label>Telephone</label>
  <textentry/>
  <label>Email</label>
  <textentry/>
</colgroup>
```

Wenn wir `<vgroup>` mit `<hgroup>` pro Zeile verwenden würden, wäre die Darstellung ziemlich schlecht, da die Breite des Texteingabe-Widgets für jede Zeile unterschiedlich ist, was ziemlich hässlich aussieht.

Beachten Sie, dass die Group-Klasse nicht von der Area-Klasse (Bereich) abgeleitet wird, da Gruppen nicht als physische Widgets existieren, sondern nur als Layoutwerkzeuge für

ihre untergeordneten Widgets oder Gruppen. Aus diesem Grund können Sie keine Attribute und Methoden der Area-Klasse für Gruppenobjekte verwenden.

18.2 Group.Append

BEZEICHNUNG

Group.Append – fügt das losgelöste Objekt als letztes Gruppenelement hinzu

ÜBERSICHT

```
moai.DoMethod(id, "Append", obj)
```

BESCHREIBUNG

Diese Methode kann verwendet werden, um das in `obj` angegebene losgelöste Objekt dem durch `id` angegebenen Gruppenobjekt hinzuzufügen. Das losgelöste Objekt wird als letztes untergeordnetes Objekt der Gruppe hinzugefügt. Nachdem diese Methode abgeschlossen ist, ändert das angegebene Objekt seinen Status von gelöst in angehängt. Deshalb dürfen Sie keine Befehle mehr für losgelöste Objekte mit diesem Objekt verwenden.

Bevor Sie diese Methode aufrufen können, müssen Sie die Gruppe in einen speziellen Zustand versetzen, in dem Elemente hinzugefügt und entfernt werden können. Dies kann durch Ausführen der Methoden `Group.InitChange` und `Group.ExitChange` für das jeweilige Gruppenobjekt erfolgen.

Losgelöste MOAI-Objekte können entweder durch Aufrufen des Befehls `moai.CreateObject()` oder durch explizites Trennen ihrer MOAI-Objekte mithilfe der Methode `Group.Remove` erstellt werden.

EINGABEN

`id` ID des Gruppenobjekts
`obj` ID des anzuhängenden Objekts

BEISPIEL

Siehe `@command{moai.CreateObject()}`

18.3 Group.Color

BEZEICHNUNG

Group.Color – setzt die Hintergrundfarbe der Gruppe

BESCHREIBUNG

Legen Sie die Hintergrundfarbe für die Gruppe fest. Dies darf nur mit Objekten wie `<hgroup>` oder `<vgroup>` verwendet werden. Es wird nicht beim Objekt `<colgroup>` unterstützt. Außerdem dürfen Sie `Group.Frame` nicht auf `True` setzen, wenn Sie dieses Attribut verwenden. Wenn Sie die Hintergrundfarbe einer gerahmten Gruppe einstellen möchten, müssen Sie eine Hilfsgruppe um die Gruppe herum anlegen, deren Hintergrundfarbe Sie dann einstellen, z.B. so:

```
<vgroup frame="true" padding="0">
  <vgroup color="#ffffff">
```



```

    ...
  </vgroup>
</vgroup>

```

Wenn Sie die Hintergrundfarbe der Gruppe `<colgroup>` setzen wollen, können Sie auch einfach eine Hilfsgruppe um die Spaltengruppe herum anlegen (siehe oben).

TYP

Zahl

ANWENDBARKEIT

I

18.4 Group.Columns

BEZEICHNUNG

Group.Columns – definiert die Anzahl Gruppenspalten

BESCHREIBUNG

Legen Sie die Anzahl der Spalten für eine zweidimensionale Gittergruppe fest. Wenn Sie diesen Tag angeben, stellen Sie sicher, dass das Gitter ausgeglichen ist, d.h. es gibt keinen Rest bei der Division der Gesamtzahl der Elemente durch die Anzahl der Spalten.

Dieses Attribut darf nur mit dem Tag `<colgroup>` verwendet werden.

TYP

Zahl

ANWENDBARKEIT

I

18.5 Group.ExitChange

BEZEICHNUNG

Group.ExitChange – beendet den Gruppenänderungsmodus

ÜBERSICHT

```
moai.DoMethod(id, "ExitChange"[, force])
```

BESCHREIBUNG

Diese Methode beendet den von `Group.InitChange` festgelegten Modus. Wenn Elemente hinzugefügt oder entfernt wurden, aktualisiert RapaGUI die Gruppe und macht die Änderungen für den Benutzer sichtbar. Sie können RapaGUI zwingen, diese Aktualisierung durchzuführen, indem Sie das Argument `force` auf `True` setzen. In diesem Fall aktualisiert RapaGUI immer die gesamte Gruppe, unabhängig davon, ob Objekte hinzugefügt oder entfernt wurden. Das Erzwingen einer Aktualisierung ist nützlich, wenn sich in Ihrer Gruppe ein Objekt befindet, für das eine Aktualisierung erzwungen werden soll.

EINGABEN

`id` ID des Gruppenobjekts

force optional: Geben Sie hier **True** an, um eine vollständige Aktualisierung zu erzwingen. Andernfalls aktualisiert RapaGUI die Gruppe nur, wenn Objekte hinzugefügt oder entfernt wurden

BEISPIEL

Siehe `@command{moai.CreateObject()}`

18.6 Group.Frame

BEZEICHNUNG

`Group.Frame` – erstellt eine gerahmte Gruppe

BESCHREIBUNG

Setzen Sie dieses Attribut auf **True**, um einen Rahmen um diese Gruppe hinzuzufügen. Sie können auch einen Titel für den Rahmen hinzufügen, indem Sie das Attribut `Group.FrameTitle` verwenden.

TYP

Boolesch

ANWENDBARKEIT

I

18.7 Group.FrameTitle

BEZEICHNUNG

`Group.FrameTitle` – setzt den Titel für die gerahmte Gruppe

BESCHREIBUNG

Wenn Sie eine gerahmte Gruppe erstellen, indem Sie `Group.Frame` auf **True** setzen, können Sie mit diesem Attribut für den Rahmen einen Titeltext hinzufügen. Dieses Attribut darf nur verwendet werden, wenn `Group.Frame` auf **True** gesetzt wurde.

Beachten Sie, dass das Ändern des Titels einer gerahmten Gruppe auf AmigaOS und kompatiblen Systemen MUI 4.0 oder besser erfordert.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

18.8 Group.HAlign

BEZEICHNUNG

`Group.HAlign` – setzt die Standard-Horizontalausrichtung

BESCHREIBUNG

Setzen Sie dieses Attribut, um festzulegen, wie nicht veränderbare Elemente ausgerichtet werden sollen, wenn sie kleiner als die ihnen zugewiesene Größe sind. Die folgenden Optionen stehen zur Verfügung:

Left Links ausgerichtet.

Right Rechts ausgerichtet.

Center Zentriert ausgerichtet. Dies ist auch voreingestellt.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

I

18.9 Group.Hide

BEZEICHNUNG

Group.Hide – blendet die Gruppe ein/aus

BESCHREIBUNG

Setzen Sie dieses Attribut, um eine Gruppe ein- oder auszublenden. Beachten Sie, dass Gruppen ihren privaten Sichtbarkeitsstatus beibehalten, so dass das Setzen von diesem Attribut nicht bedeutet, dass die Gruppe einfach **Area.Hide** auf alle ihre Elemente setzt. Sowohl Widgets als auch Gruppen behalten ihren eigenen Sichtbarkeitsstatus.

TYP

Boolesch

ANWENDBARKEIT

ISG

18.10 Group.HorizSpacing

BEZEICHNUNG

Group.HorizSpacing – setzt den horizontalen Abstand

BESCHREIBUNG

Legen Sie die Anzahl der geräteunabhängigen Pixel fest, die als horizontaler Abstand zwischen den Elementen der Gruppe verwendet werden soll.

TYP

Zahl

ANWENDBARKEIT

I

18.11 Group.Icon

BEZEICHNUNG

Group.Icon – setzt das Gruppen-Icon für die Seitenansicht

BESCHREIBUNG

Wenn Sie Gruppen für eine Seitenansicht definieren, können Sie mit diesem Attribut ein Bild für eine Gruppe angeben, das in dem Widget zum Durchblättern der einzelnen Seiten angezeigt werden soll. Siehe [Abschnitt 34.1 \[Pageview-Klasse \(Seitenansicht\)\]](#), Seite 199, für Details. Ob dieses Attribut einen Hollywood-Pinsel oder ein Hollywood-Icon erwartet, hängt davon ab, was Sie im Attribut `Group.IconType` angeben. Standardmäßig erwartet `Group.Icon` einen Hollywood-Pinsel. Um das Bild von einer Gruppenseite zu entfernen, setzen Sie dieses Attribut auf `-1`.

Beachten Sie, dass RapaGUI das Bild möglicherweise skaliert, um es an die DPI-Einstellung des aktuellen Monitors anzupassen. Bitte lesen Sie das Kapitel über High-DPI-Unterstützung für weitere Informationen. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), Seite 21, für Details.

Bitte lesen Sie auch den Bilder-Cache von RapaGUI, um mehr über die Unterstützung von Icons in RapaGUI zu erfahren. Siehe [Abschnitt 3.20 \[Bilder-Cache\]](#), Seite 32, für Details.

Ab RapaGUI 1.1 hat dieses Attribut eine Anwendbarkeit von ISG. Beachten Sie, dass unter AmigaOS und Kompatiblen das Ändern des Gruppensymbols zur Laufzeit für alle Seitenansichtsmodi (außer dem Listenmodus) nur unter MUI 4.0 oder höher verfügbar ist.

TYP

Zahl

ANWENDBARKEIT

ISG

18.12 Group.IconScale

BEZEICHNUNG

Group.IconScale – konfiguriert die automatische Bildskalierung (V2.0)

BESCHREIBUNG

Wenn `Group.Icon` auf einen Rasterpinsel gesetzt wurde und RapaGUI auf einem High-DPI-Display ausgeführt wird, skaliert RapaGUI automatisch die Rastergrafiken des Pinsels, um sie an die DPI-Einstellung des aktuellen Monitors anzupassen. Wenn Sie dies nicht möchten, setzen Sie diesen Tag auf `False`.

Alternativ können Sie die automatische Bildskalierung auch global deaktivieren, indem Sie beim Aufrufen vom Befehl `@REQUIRE` mit RapaGUI den Tag `ScaleGUI` auf `False` setzen. Siehe [Abschnitt 3.4 \[Initialisierung von RapaGUI\]](#), Seite 11, für Details.

Bitte lesen Sie auch das Kapitel über High-DPI-Unterstützung in RapaGUI, um mehr über die Unterstützung von High-DPI-Displays zu erfahren. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), Seite 21, für Details.

TYP

Boolesch

ANWENDBARKEIT

I

18.13 Group.IconType

BEZEICHNUNG

Group.IconType – legt den zu verwendenden Icon-Typ fest (V2.0)

BESCHREIBUNG

Mit diesem Attribut können Sie den Typ des Hollywood-Bildobjekts festlegen, das im Attribut `Group.Icon` übergeben wird. Standardmäßig erwartet `Group.Icon` einen Hollywood-Pinsel. Durch Setzen von `Group.IconType` können Sie jedoch einen anderen Hollywood-Bildtyp verwenden.

Die folgenden Bildtypen sind derzeit verfügbar:

Brush Verwendet einen Hollywood-Pinsel. Dies ist der Standardtyp. Sie können entweder Raster- oder Vektorpinsel verwenden. Vektorpinsel haben den Vorteil, dass sie ohne Qualitätsverlust auf jede beliebige Auflösung skaliert werden können. Dies ist sehr nützlich, wenn Sie Programme entwerfen, die mit High-DPI-Monitoren kompatibel sein sollen. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), [Seite 21](#), für Details.

Icon Verwendet ein Hollywood-Icon. Dieser Bildtyp hat den Vorteil, dass er mehrere Teilbilder unterschiedlicher Größe enthalten kann. Dadurch ist es möglich, Bilder in unterschiedlichen Auflösungen bereitzustellen, was bei der Entwicklung von Programmen, die mit High-DPI-Monitoren kompatibel sein sollen, sehr nützlich sein kann. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), [Seite 21](#), für Details.

Beachten Sie, dass Sie den Standard aller `IconType`-Attribute global auf Hollywood-Icon ändern können, indem Sie den Tag `Application.UseIcons` setzen. Siehe [Abschnitt 9.13 \[Application.UseIcons\]](#), [Seite 70](#), für Details.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

I

18.14 Group.InitChange

BEZEICHNUNG

Group.InitChange – bereitet die Gruppe für das Hinzufügen/Entfernen von Elementen vor

ÜBERSICHT

```
moai.DoMethod(id, "InitChange")
```

BESCHREIBUNG

Diese Methode bereitet eine Gruppe zum Hinzufügen oder Entfernen von Elementen vor. Wann immer Sie Methoden wie `Group.Append`, `Group.Prepend`, `Group.Insert` oder `Group.Remove` verwenden, müssen Sie zuerst diese Methode aufrufen, um die Gruppe in diesen speziellen Austauschmodus zu versetzen. Wenn Sie fertig sind, verwenden Sie `Group.ExitChange`, um die Gruppe neu zu gestalten.

EINGABEN

`id` ID des Gruppenobjekts

BEISPIEL

Siehe `@command{moai.CreateObject()}`

18.15 Group.Insert

BEZEICHNUNG

`Group.Insert` – fügt ein losgelöstes Objekt nach dem angegebenen Element ein

ÜBERSICHT

`moai.DoMethod(id, "Insert", obj, pred)`

BESCHREIBUNG

Diese Methode kann verwendet werden, um das durch `obj` spezifizierte losgelöste Objekt in das durch `id` spezifizierte Gruppenobjekt einzufügen. Das losgelöste Objekt wird nach dem in `pred` angegebenen Element eingefügt. Nachdem diese Methode beendet ist, ändert das angegebene Objekt seinen Zustand von losgelöst in angehängt. Deshalb dürfen Sie keine Befehle mehr für losgelöste Objekte für dieses Objekt verwenden.

Bevor Sie diese Methode aufrufen können, müssen Sie die Gruppe in einen speziellen Zustand versetzen, in dem Elemente hinzugefügt und entfernt werden können. Dies kann durch Ausführen der Methoden `Group.InitChange` und `Group.ExitChange` für das jeweilige Gruppenobjekt erfolgen.

Losgelöste MOAI-Objekte können entweder durch Aufrufen des Befehls `moai.CreateObject()` oder durch explizites Trennen ihrer MOAI-Objekte mithilfe der Methode `Group.Remove` erstellt werden.

EINGABEN

`id` ID des Gruppenobjekts

`obj` ID des einzufügenden Objekts

`pred` Das Objekt wird nach diesem Objekt eingefügt

BEISPIEL

Siehe `@command{moai.CreateObject()}`

18.16 Group.Padding

BEZEICHNUNG

Group.Padding – legt den Abstand zwischen Rahmen und Gruppe fest

BESCHREIBUNG

Wenn Sie eine gerahmte Gruppe erstellen, können Sie mit diesem Attribut die Anzahl der geräteunabhängigen Füll-Pixel zwischen dem Rahmen und den Elementen der Gruppe festlegen.

Unter Windows, Linux und macOS definiert dieses Attribut auch die Anzahl der Füll-Pixel um die Gruppe herum, wenn die Gruppe Teil eines Pageview-Objekts (Seitenansicht) ist. Siehe [Abschnitt 34.1 \[Pageview-Klasse\]](#), [Seite 199](#), für Details.

TYP

Zahl

ANWENDBARKEIT

I

18.17 Group.Paint

BEZEICHNUNG

Group.Paint – fordert eine Zeichnungsbenachrichtigung an

BESCHREIBUNG

Richten Sie eine Überwachung für dieses Attribut ein, damit Ihre Callback-Funktion immer dann aufgerufen wird, wenn der Gruppenhintergrund gezeichnet werden muss. Ihre Callback-Funktion kann dann eigene Grafiken in den Gruppenhintergrund zeichnen.

RapaGUI übergibt den Identifikator eines Hollywood-Pinsels, dessen Größe genau so groß ist wie der Gruppenhintergrund. Sie müssen dann die gewünschten Hintergrundgrafiken zu diesem Pinsel zeichnen. Genau, Sie müssen nur in dem Rechteck zeichnen, das durch die vier Koordinaten `X`, `Y`, `Width` und `Height` definiert ist, die auch an Ihre Callback-Funktion übergeben werden. Diese vier Koordinaten beschreiben einen rechteckigen Bereich innerhalb der Dimensionen des Pinsels, der an Ihre Callback-Funktion übergeben wird. Wenn ein vollständiges Neuzeichnen benötigt wird, sind `X` sowie `Y` 0 und `Width` sowie `Height` stimmen mit den Abmessungen des Pinsels überein. Meistens wird jedoch nur ein partielles Neuzeichnen benötigt und dann müssen Sie nur auf den Teil des Pinsels zeichnen, der durch diese Koordinaten definiert ist.

Die folgenden zusätzlichen Argumente werden an Ihre Callback-Funktion übergeben:

Brush: Enthält den Identifikator eines Pinsels, auf den Sie zeichnen müssen. Verwenden Sie den Befehl `SelectBrush()` von Hollywood, um diesen Pinsel als Ausgabegerät in Ihrer Callback-Funktion auszuwählen. Vergessen Sie nicht, `EndSelect()` aufzurufen, wenn Sie fertig sind!

ViewWidth: Enthält die gesamte Gruppenbreite. Diese ist auch identisch mit der Breite des Pinsels, der an Ihre Callback-Funktion übergeben wird.

- ViewHeight:** Enthält die gesamte Gruppenhöhe. Diese ist auch identisch mit der Höhe des Pinsels, der an Ihre Callback-Funktion übergeben wird.
- X:** Enthält die x-Position innerhalb des Pinsels, an der Sie mit dem Zeichnen beginnen sollen. Siehe oben für Details.
- Y:** Enthält die y-Position innerhalb des Pinsels, an der Sie mit dem Zeichnen beginnen sollen. Siehe oben für Details.
- Width:** Enthält die Anzahl der Spalten, die Sie auf den Pinsel malen sollten (ab X). Siehe oben für Details.
- Height:** Enthält die Anzahl der Zeilen, die Sie auf den Pinsel malen sollten (ab Y). Siehe oben für Details.

Beachten Sie, dass `Group.Paint` nur mit Objekten wie `<hgroup>` oder `<vgroup>` verwendet werden darf. `<colgroup>`-Objekte werden nicht unterstützt. Außerdem dürfen Sie `Group.Frame` nicht auf `True` setzen, wenn Sie dieses Attribut verwenden. Wenn Sie den Hintergrund einer gerahmten Gruppe zeichnen möchten, müssen Sie eine Hilfsgruppe um die Gruppe herum erstellen, auf deren Hintergrund Sie zeichnen möchten, z.B. so:

```
<vgroup frame="true" padding="0">
  <vgroup notify="paint">
    ...
  </vgroup>
</vgroup>
```

Wenn Sie den Hintergrund eines `<colgroup>` zeichnen möchten, können Sie auch einfach eine Hilfsgruppe um die Spaltengruppe herum erstellen (siehe oben).

Beachten Sie, dass dieses Attribut bei AmigaOS und Kompatiblen nur mit MUI 4.0 oder höher verfügbar ist.

TYP

Boolesch

ANWENDBARKEIT

N

18.18 Group.Prepend

BEZEICHNUNG

`Group.Prepend` – fügt ein losgelöstes Objekt als erstes Gruppenelement hinzu

ÜBERSICHT

```
moai.DoMethod(id, "Prepend", obj)
```

BESCHREIBUNG

Diese Methode kann verwendet werden, um das durch `obj` angegebene losgelöste Objekt dem durch `id` angegebene Gruppenobjekt hinzuzufügen. Das losgelöste Objekt wird als erstes Element der Gruppe hinzugefügt. Nachdem diese Methode beendet wurde, ändert das angegebene Objekt seinen Zustand von losgelöst in angehängt. Deshalb dürfen Sie keine Befehle mehr für losgelöste Objekte für dieses Objekt verwenden.

Bevor Sie diese Methode aufrufen können, müssen Sie die Gruppe in einen speziellen Zustand versetzen, in dem Elemente hinzugefügt und entfernt werden können. Dies kann durch Ausführen der Methoden `Group.InitChange` und `Group.ExitChange` für das jeweilige Gruppenobjekt erfolgen.

Losgelöste MOAI-Objekte können entweder durch Aufrufen des Befehls `moai.CreateObject()` oder durch explizites Trennen ihrer MOAI-Objekte mithilfe der Methode `Group.Remove` erstellt werden.

EINGABEN

`id` ID des Gruppenobjekts
`obj` ID des einzufügenden Objekts

BEISPIEL

Siehe `@command{moai.CreateObject()}`

18.19 Group.Remove

BEZEICHNUNG

`Group.Remove` – löst das Objekt von der Gruppe

ÜBERSICHT

`moai.DoMethod(id, "Remove", obj)`

BESCHREIBUNG

Mit dieser Methode kann das angegebene Objekt von der angegebenen Gruppe gelöst werden. Nachdem diese Methode ausgeführt wurde, wird das angegebene Objekt seinen Zustand von angehängt auf losgelöst ändern. Das bedeutet, dass Sie es nun mit der Methode `Group.Insert` an eine andere Gruppe anhängen oder mit dem Befehl `moai.FreeObject()` aus dem Speicher löschen können.

Bevor Sie diese Methode aufrufen können, müssen Sie die Gruppe in einen speziellen Zustand versetzen, in dem Elemente hinzugefügt und entfernt werden können. Dies kann durch Ausführen der Methoden `Group.InitChange` und `Group.ExitChange` für das jeweilige Gruppenobjekt erfolgen.

EINGABEN

`id` ID des Gruppenobjekts
`obj` ID des zu loslösenden Objekts

BEISPIEL

```
moai.DoMethod("mygroup", "initchange")
moai.DoMethod("mygroup", "remove", "mychild")
moai.DoMethod("mygroup", "exitchange", false)
```

Der obige Code entfernt das Element "mychild" aus der Gruppe "mygroup". Sie können dann "mychild" an eine andere Gruppe anhängen oder aus dem Speicher löschen.

18.20 Group.SameSize

BEZEICHNUNG

Group.SameSize – stellt die gleiche Größe für alle Elemente ein

BESCHREIBUNG

Setzen Sie dieses Attribut auf `True`, um die Gruppe zu zwingen, die gleiche Größe für alle Elemente zu verwenden. Dadurch entsteht oft ein einheitlicheres und besseres Erscheinungsbild. RapaGUI setzt dieses Attribut automatisch für horizontale Gruppen, die eine Reihe von Schaltflächen enthalten.

TYP

Boolesch

ANWENDBARKEIT

I

18.21 Group.Spacing

BEZEICHNUNG

Group.Spacing – setzt den Abstand

BESCHREIBUNG

Stellen Sie die Anzahl der geräteunabhängigen Pixel ein, die als horizontaler und vertikaler Abstand zwischen den Elementen der Gruppe verwendet werden soll. Das Setzen dieses Attributs hat den gleichen Effekt wie das Setzen von `Group.HorizSpacing` und `Group.VertSpacing`.

TYP

Zahl

ANWENDBARKEIT

I

18.22 Group.Title

BEZEICHNUNG

Group.Title – setzt den Gruppentitel für eine Seitenansicht

BESCHREIBUNG

Wenn Sie Gruppen für eine Pageview-Klasse (Seitenansicht) definieren, können Sie mit diesem Attribut einen Titel für diese Gruppe angeben, der in dem Widget angezeigt werden soll, das zum Durchblättern der einzelnen Seiten verwendet wird.

Ab RapaGUI 1.1 hat dieses Attribut eine Anwendbarkeit von `ISG`. Zuvor war seine Anwendbarkeit nur `I`. Beachten Sie, dass bei AmigaOS und Kompatiblen der Gruppentitel zur Laufzeit für alle Seitenansichtsmodi (außer dem Listenmodus) nur bei MUI 4.0 oder höher geändert werden kann.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

18.23 Group.VAlign**BEZEICHNUNG**

Group.VAlign – setzt die Standard-Vertikalausrichtung

BESCHREIBUNG

Setzen Sie dieses Attribut, um festzulegen, wie nicht veränderbare Elemente ausgerichtet werden sollen, wenn sie kleiner als die ihnen zugewiesene Größe sind. Die folgenden Optionen stehen zur Verfügung:

Top Oben ausgerichtet.

Bottom Unten ausgerichtet.

Center Zentriert ausgerichtet. Dies ist auch voreingestellt.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

I

18.24 Group.VertSpacing**BEZEICHNUNG**

Group.VertSpacing – setzt den vertikalen Abstand

BESCHREIBUNG

Legen Sie die Anzahl der geräteunabhängigen Pixel fest, die als vertikaler Abstand zwischen den Elementen der Gruppe verwendet werden soll.

TYP

Zahl

ANWENDBARKEIT

I

18.25 Group.Weight**BEZEICHNUNG**

Group.Weight – setzt die Gruppengröße

BESCHREIBUNG

Dieses Attribut setzt die Größe eines Gruppenobjekts innerhalb einer anderen Gruppe von Objekten. Standardmäßig haben alle Objekte einer Gruppe eine Größe von 100. Wenn Sie ein Objekt haben wollen, das doppelt so groß erscheint, müssen Sie ihm eine

Größe von 200 geben. Um eine Gruppe eineinhalb Mal so groß erscheinen zu lassen, verwenden Sie einfach 150 und so weiter.

Natürlich ist dieses Attribut nur für Gruppen sinnvoll, die in der Größe veränderbar sind.

TYP

Zahl

ANWENDBARKEIT

I

19 HLine-Klasse (HorizontalLinie)

19.1 Übersicht

Die HLine-Klasse (HorizontalLinie) leitet sich von der Area-Klasse ab und erzeugt eine horizontale Trennlinie, die als separate Gruppe von Widgets verwendet werden kann. Es gibt auch eine VLine-Klasse, welche vertikale Trennlinien erzeugt. Siehe [Abschnitt 59.1 \[VLine-Klasse\]](#), [Seite 311](#), für Details.

Die HLine-Klasse hat keine Attribute.

20 Hollywood-Klasse

20.1 Übersicht

Die Hollywood-Klasse leitet sich von der Area-Klasse ab, die es Ihnen erlaubt, ein komplettes Hollywood-Display als Widget in Ihre GUI einzubetten. Immer wenn Sie etwas auf ein Hollywood-Display zeichnen, das an ein Widget angehängt ist, wird es automatisch auch in Ihr Widget gezeichnet. Sie können sogar das Hollywood-Display ausblenden und es wird immer noch funktionieren. Außerdem werden alle Mausklicks und Tastaturanschläge, die innerhalb der Hollywood-Klasse stattfinden, als normale Hollywood-Ereignisse an das entsprechende Widget weitergeleitet. So erlaubt Ihnen die Hollywood-Klasse, fast alle leistungsstarken Befehle von Hollywood in einem Widget transparent zu nutzen.

Hier ist ein Beispiel, wie Sie das Hollywood-Display 1 in Ihre GUI einbinden können:

```
<hollywood display="1"/>
```

Beachten Sie, dass Hollywood-Widgets standardmäßig nicht skalierbar sind. Sie können dies ändern, indem Sie die Attribute `Area.FixWidth` und `Area.FixHeight` entsprechend setzen. Wenn Sie eines dieser Attribute auf `False` setzen, erhalten Sie ein skalierbares Hollywood-Widget und wie bei normalen Hollywood-Displays wird Ihrem Hollywood-Widget auch eine `SizeWindow`-Benachrichtigung gesendet, wenn sich die Größe des Widgets ändert. Sie können dann den Inhalt Ihres Widgets an die neue Größe anpassen.

Beachten Sie auch, dass standardmäßig alle Hollywood-Widgets so skaliert werden, dass sie der DPI-Einstellung des aktuellen Monitors entsprechen. Wenn Sie das nicht möchten, können Sie den Tag `ScaleHollywood` in `@REQUIRE` auf `False` setzen. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), Seite 21, für Details.

Siehe [Abschnitt 3.19 \[Hollywood-Brücke\]](#), Seite 27, für Details.

20.2 Hollywood.Display

BEZEICHNUNG

`Hollywood.Display` – integriert das Hollywood-Display in einem Objekt

BESCHREIBUNG

Integriert ein Hollywood-Display in einem Objekt. Hier müssen Sie den Identifikator eines Hollywood-Displays angeben.

Dieses Attribut ist obligatorisch und muss immer dann angegeben werden, wenn Sie ein Hollywood-Objekt erstellen. Sie können kein leeres Hollywood-Objekt ohne angehängtes Display erstellen.

Standardmäßig ist das Widget nicht in der Größe veränderbar. Sie können dies ändern, indem Sie die Attribute `Area.FixWidth` und `Area.FixHeight` auf `False` setzen. In diesem Fall wird Ihr Hollywood-Widget größenveränderbar und wenn der Benutzer die Fenstergröße ändert, erhält Ihre Hollywood-Display ein "SizeWindow"-Benachrichtigung, das Sie mit dem Befehl `InstallEventHandler()` überwachen können. Sie können dann auf dieses Ereignis entsprechend reagieren und Ihr Display neu zeichnen, etc.

TYP

Zahl

ANWENDBARKEIT

IS

20.3 Hollywood.DropFile

BEZEICHNUNG

Hollywood.DropFile – benachrichtigt, wenn Datei(en) über das Widget gezogen wird (V1.1)

BESCHREIBUNG

Wenn Sie eine Benachrichtigung für dieses Attribut einrichten, führt RapaGUI Ihre Callback-Funktion aus, sobald eine oder mehrere Dateien auf das Hollywood-Widget gezogen wurden. Das Feld **TriggerValue** der Nachrichtentabelle wird auf eine Tabelle gesetzt, die eine Liste aller auf das Widget gezogenen Dateien enthält.

Zusätzlich enthält diese Tabelle die folgenden zwei weiteren Felder:

- X:** Die x-Position, an der der Benutzer die Datei(en) abgelegt hat. Dies ist relativ zur linken Ecke des Widgets.
- Y:** Die y-Position, an der der Benutzer die Datei(en) abgelegt hat. Dies ist relativ zur oberen Ecke des Widgets.

Siehe [Abschnitt 3.7 \[Benachrichtigungen der Attribute\], Seite 14](#), für Details.

Beachten Sie, dass **Hollywood.DropFile** nur dann ausgelöst wird, wenn **Hollywood.DropTarget** zuerst auf **True** gesetzt wurde.

TYP

Boolesch

ANWENDBARKEIT

N

20.4 Hollywood.DropTarget

BEZEICHNUNG

Hollywood.DropTarget – lässt das Ablegen von Dateien überm Widget zu oder nicht (V1.1)

BESCHREIBUNG

Setzen Sie dies auf **True**, wenn Dateien auf diesem Hollywood-Widget abgelegt werden können. Sie können das Attribut **Hollywood.DropFile** überwachen, um zu erfahren, wenn der Benutzer eine oder mehrere Dateien auf dem Hollywood-Widget ablegt.

TYP

Boolesch

ANWENDBARKEIT

ISG

21 HSpace-Klasse (HorizontalAbstand)

21.1 Übersicht

HSpace-Klasse (HorizontalAbstand) leitet sich von der Area-Klasse ab und erzeugt Objekte mit einer festen geräteunabhängigen Pixelgröße. Dies wird typischerweise zur Feinabstimmung des GUI-Layouts verwendet. Um HSpace-Objekte zu erstellen, die frei skalierbar sind, können Sie stattdessen Rectangle-Klasse (Rechteck) verwenden. Siehe [Abschnitt 41.1 \[Rectangle-Klasse\]](#), [Seite 221](#), für Details.

21.2 HSpace.Width

BEZEICHNUNG

HSpace.Width – setzt den horizontalen Abstand

BESCHREIBUNG

Legt den gewünschten horizontalen Abstand für dieses Objekt in geräteunabhängigen Pixel fest.

TYP

Zahl

ANWENDBARKEIT

I

22 HSplitter-Klasse

22.1 Übersicht

Die HSplitter-Klasse ist eine spezielle Variante der Group-Klasse. Sie erstellt eine horizontale Gruppe aus zwei untergeordneten Elementen mit einem Schieberahmen dazwischen, der es dem Benutzer ermöglicht, die Größe der beiden Elementen der Gruppe durch Ziehen des Schieberahmens individuell anzupassen.

Beachten Sie, dass HSplitter-Gruppen immer genau zwei untergeordneten Elementen enthalten müssen. Darüber hinaus müssen die Elemente in der Größe angepasst werden, da die HSplitter-Klasse eine Größenänderung ihrer untergeordneten Elemente ermöglicht. Hier ist ein Beispiel-XML-Auszug zum Erstellen eines horizontalen Splitter-Layouts mit zwei Textansichten:

```
<hsplitter>
  <textview>One</textview>
  <textview>Two</textview>
</hsplitter>
```

Die HSplitter-Klasse ist seit RapaGUI 2.0 verfügbar.

22.2 HSplitter.Border

BEZEICHNUNG

HSplitter.Border – stellt den Stil des Schieberahmens ein (V2.0)

BESCHREIBUNG

Setzen Sie diesen Wert auf `False`, damit der Schieberahmen einen anderen Rahmenstil verwendet. Beachten Sie, dass dies nicht auf allen Plattformen unterstützt wird.

TYP

Boolesch

ANWENDBARKEIT

I

22.3 HSplitter.Gravity

BEZEICHNUNG

HSplitter.Gravity – setzt/ermittelt die Anziehungskraft des Schieberahmens (V2.0)

BESCHREIBUNG

Stellt die Anziehungskraft des Schieberahmens ein oder ermittelt sie. Die Anziehungskraft ist ein Wert zwischen 0 und 100, der die Position des Schieberahmens steuert, während die Größe der Splittergruppe geändert wird. Der Anziehungskraftwert teilt der Splittergruppe mit, um wie viel das linke Element während der Größenänderung ändert. Zum Beispiel:

- 0: Nur das rechte Element wird automatisch in der Größe angepasst.
- 50: Beide Elemente ändern im gleichen Verhältnis ihre Größe.

- 100: Nur das linke Element wird automatisch in der Größe angepasst.

Die Standardanziehungskraft des Schieberahmens ist 0.

TYP

Zahl

ANWENDBARKEIT

SG

22.4 HSplitter.MinPaneSize

BEZEICHNUNG

HSplitter.MinPaneSize – setzt oder ermittelt die minimale Elementgröße (V2.0)

BESCHREIBUNG

Legt die minimale Elementgröße fest oder ermittelt sie. Die standardmäßige Mindestelementgröße ist 0, was bedeutet, dass jedes Element durch Ziehen des Schieberahmens auf Null reduziert werden kann, wodurch eines der Elemente entfernt wird. Um dieses Verhalten zu verhindern (und das Ziehen des Schieberahmens außerhalb des Bereichs zu unterbinden), legen Sie eine Mindestgröße fest, zum Beispiel 20.

Dieser Wert ist in geräteunabhängige Pixel.

TYP

Zahl

ANWENDBARKEIT

ISG

22.5 HSplitter.Position

BEZEICHNUNG

HSplitter.Position – setzt/ermittelt die Position des Schieberahmens (V2.0)

BESCHREIBUNG

Legt die Position des Schieberahmens fest oder ermittelt sie, der die Elemente der Gruppe trennt. Dieser Wert ist in geräteunabhängige Pixel.

TYP

Zahl

ANWENDBARKEIT

SG

22.6 HSplitter.Split

BEZEICHNUNG

HSplitter.Split – teilt die Gruppe in zwei Elemente auf (V2.0)

ÜBERSICHT

```
moai.DoMethod(id, "Split", pos)
```

BESCHREIBUNG

Hiermit wird die Gruppe in zwei Elemente aufgeteilt. Das Argument `pos` gibt die Position des Schieberahmens in geräteunabhängige Pixel an. Wenn dieser Wert positiv ist, gibt er die Größe des linken Elements an. Wenn er negativ ist, gibt sein absoluter Wert die Größe des rechten Elements an. Geben Sie schließlich 0 an, um die Standardposition zu wählen (die Hälfte der gesamten Fensterbreite).

EINGABEN

<code>id</code>	Identifikator des Splitterobjekts
<code>pos</code>	gewünschte Position des Schieberahmens

22.7 HSplitter.Unsplit

BEZEICHNUNG

`HSplitter.Unsplit` – hebt die Teilung der Elemente auf (V2.0)

ÜBERSICHT

```
moai.DoMethod(id, "Unsplit", idx)
```

BESCHREIBUNG

Dadurch wird die Teilung der Elemente aufgehoben und das in `idx` angegebene Element ausgeblendet. Wenn `idx` 0 ist, wird das linke Element ausgeblendet, wenn es 1 ist, wird das rechte Element ausgeblendet. Um beide Elemente wieder sichtbar zu machen, verwenden Sie die Methode `HSplitter.Split`.

EINGABEN

<code>id</code>	Identifikator des Splitterobjekts
<code>idx</code>	Index des zu entfernenden Elements

23 HTMLview-Klasse (HTMLAnsicht)

23.1 Übersicht

HTMLview-Klasse (HTMLAnsicht) leitet sich von der Area-Klasse ab und erstellt ein Widget, das eine HTML-Seite entweder als Rohdaten, als lokal oder auf einem entfernten Server gespeicherte Dateiquelle anzeigt. Sie unterstützt auch einige Browser-Fähigkeiten wie eine Liste der besuchten Seiten und die Möglichkeit, durch diese Liste zu blättern.

Unter Windows und macOS verwendet diese Klasse ein vom Betriebssystem bereitgestellte HTML-Steuerung. Unter Linux erfordert diese Klasse die Installation von WebKitGTK+. Da dies nicht auf jedem System verfügbar ist, gibt es auch eine separate RapaGUI-Version ohne diese Klasse und damit ohne die Abhängigkeit von WebKitGTK+. Auf AmigaOS und kompatiblen Systemen benötigt diese Klasse die Erweiterung `HTMLview.mcc`.

23.2 HTMLview.CanGoBack

BEZEICHNUNG

HTMLview.CanGoBack – ermittelt, ob der Browser zurück gehen kann

BESCHREIBUNG

Dieses Attribut wird auf `True` gesetzt, wenn es eine Seite im Verlauf gibt, zu der der Browser zurückkehren kann. Sie können eine Benachrichtigung für dieses Attribut einrichten und die Schaltfläche "Zurück" Ihrer Symbolleiste je nach Zustand dieses Attributs aktivieren oder deaktivieren.

TYP

Boolesch

ANWENDBARKEIT

GN

23.3 HTMLview.CanGoForward

BEZEICHNUNG

HTMLview.CanGoForward – ermittelt, ob der Browser vorwärts gehen kann

BESCHREIBUNG

Dieses Attribut wird auf `True` gesetzt, wenn es eine Seite im Verlauf gibt, zu der der Browser vorwärts gehen kann. Sie können eine Benachrichtigung für dieses Attribut einrichten und die Schaltfläche "Vorwärts" Ihrer Symbolleiste je nach Zustand dieses Attributs deaktivieren oder aktivieren.

TYP

Boolesch

ANWENDBARKEIT

GN

23.4 HTMLview.ClearHistory

BEZEICHNUNG

HTMLview.ClearHistory – löscht den Browserverlauf

ÜBERSICHT

```
moai.DoMethod(id, "ClearHistory")
```

BESCHREIBUNG

Diese Methode löscht den gesamten Browserverlauf.

EINGABEN

id ID des HTMLview-Objekts

23.5 HTMLview.Contents

BEZEICHNUNG

HTMLview.Contents – setzt oder ermittelt den Widget-Inhalt

BESCHREIBUNG

Setzt oder ermittelt die vom Widget angezeigten HTML-Daten. Wenn Sie dieses Attribut setzen, müssen Sie dem Widget gültigen HTML-formatierten Code übergeben. Mit diesem Attribut erhalten Sie den Quellcode der aktuell angezeigten Seite.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

23.6 HTMLview.File

BEZEICHNUNG

HTMLview.File – setzt/ermittelt die Datei zum Anzeigen

BESCHREIBUNG

Legen Sie die Datei fest, die im Widget angezeigt werden soll. Dies ist normalerweise eine HTML-Seite, aber Sie können auch einen Pfad zu einem Bild übergeben.

Beachten Sie, dass dieses Attribut im Gegensatz zu `HTMLview.URL` nur einen normalen Pfad zu einer Datei erwartet, d.h. verwenden Sie hier nicht den Protokoll-Präfix `file://`.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

23.7 HTMLview.GoBack

BEZEICHNUNG

HTMLview.GoBack – geht im Browserverlauf zurück

ÜBERSICHT

```
moai.DoMethod(id, "GoBack")
```

BESCHREIBUNG

Navigiert im Verlauf der besuchten Seiten zurück.

EINGABEN

id ID des HTMLview-Objekts

23.8 HTMLview.GoForward

BEZEICHNUNG

HTMLview.GoForward – geht im Browserverlauf vorwärts

ÜBERSICHT

```
moai.DoMethod(id, "GoForward")
```

BESCHREIBUNG

Navigiert im Verlauf der besuchten Seiten vorwärts.

EINGABEN

id ID des HTMLview-Objekts

23.9 HTMLview.Reload

BEZEICHNUNG

HTMLview.Reload – lädt die aktuelle Seite neu

ÜBERSICHT

```
moai.DoMethod(id, "Reload")
```

BESCHREIBUNG

Die aktuelle Seite wird neu geladen.

EINGABEN

id ID des HTMLview-Objekts

23.10 HTMLview.Search

BEZEICHNUNG

HTMLview.Search – durchsucht die aktuelle Seite

ÜBERSICHT

```
found = moai.DoMethod(id, "Search", t$, flags$)
```

BESCHREIBUNG

Diese Methode sucht auf der aktuellen Seite nach der Zeichenkette `t$` und scrollt das Ergebnis in die Ansicht und wählt es aus.

`flags$` kann eine Kombination der folgenden Flags sein:

`CaseSensitive`

Groß- und Kleinschreibung wird beachtet.

`Backwards`

Sucht rückwärts.

Wenn Sie mehrere Optionen in `flags$` angeben, trennen Sie sie mit einem Semikolon/Strichpunkt (;).

EINGABEN

`id` ID des HTMLview-Objekts

`t$` Zeichenkette, die gesucht wird

`flags$` Kombination von Flags oder leere Zeichenfolge für Standardoptionen

RÜCKGABEWERTE

`found` gibt `True` oder `False` zurück, je nachdem, ob der Text gefunden wurde (V2.0)

23.11 HTMLview.Title

BEZEICHNUNG

`HTMLview.Title` – ermittelt den Titel der aktuellen Seite

BESCHREIBUNG

Ermittelt den Titel der aktuell angezeigten Seite, d.h. den Wert des Tags `<title>`.

TYP

Zeichenkette

ANWENDBARKEIT

G

23.12 HTMLview.URL

BEZEICHNUNG

`HTMLview.URL` – setzt/ermittelt die aktuelle URL

BESCHREIBUNG

Setzt oder ermittelt die URL, die im Widget angezeigt werden soll oder wird. Diese URL muss ein Protokoll-Präfix enthalten, typischerweise `http://`. Beim Öffnen lokaler Dateien müssen Sie `file://` als Protokoll-Präfix verwenden. Aber Sie können auch einfach `HTMLview.File` benutzen, was in diesem Fall bequemer ist.

TYP

Zeichenkette

ANWENDBARKEIT
ISG

24 Hyperlink-Klasse

24.1 Übersicht

Die Hyperlink-Klasse leitet sich von der Area-Klasse ab und erstellt ein anklickbares Textobjekt, das die angegebene URL in einem Browser öffnet, wenn der Benutzer darauf klickt. Hier ist ein XML-Beispielauszug zum Erstellen eines Hyperlink-Widgets, das <http://www.hollywood-mal.com> öffnet, wenn Sie darauf klicken:

```
<hyperlink>http://www.hollywood-mal.com</hyperlink>
```

Wenn Sie nicht möchten, dass die URL angezeigt wird, können Sie das Attribut `Hyperlink.Label` verwenden, um eine andere Beschriftung für das Hyperlink-Objekt festzulegen, zum Beispiel:

```
<hyperlink label="Click me">http://www.hollywood-mal.com</hyperlink>
```

Die Hyperlink-Klasse ist seit RapaGUI 2.0 verfügbar.

24.2 Hyperlink.Label

BEZEICHNUNG

`Hyperlink.Label` – setzt/ermittelt den Beschriftungstext (V2.0)

BESCHREIBUNG

Legt den Beschriftungstext fest, der vom Hyperlink-Widget angezeigt werden soll, oder ermittelt ihn. Wenn dies nicht angegeben wird, zeigt das Hyperlink-Widget die URL an, die mit `Hyperlink.URL` festgelegt wurde.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

24.3 Hyperlink.URL

BEZEICHNUNG

`Hyperlink.URL` – setzt/ermittelt die Ziel-URL (V2.0)

BESCHREIBUNG

Legt die Ziel-URL fest oder ermittelt sie, die in einem Browser geöffnet werden soll, wenn der Benutzer auf das Hyperlink-Widget klickt. Diese muss immer angegeben werden.

TYP

Zeichenkette

ANWENDBARKEIT

SG

25 Image-Klasse (Bild)

25.1 Übersicht

Die Image-Klasse (Bild) leitet sich von der Area-Klasse ab und zeigt das angegebene Bild an. Die Bildpixeldaten stammen von einem Hollywood-Pinsel oder -Icon, je nachdem, was in `Image.BrushType` eingestellt ist.

Hier ist eine Beispiel für eine XML-Definition:

```
<image brush="1"/>
```

Der obige XML-Code erzeugt ein Bildobjekt aus dem Hollywood-Pinsel Nummer 1. Die Image-Klasse unterstützt in Hollywood-Pinseln/-Icons Masken- und Alphakanaltransparenz, so dass Sie auch Bilder mit transparenten Bereichen einbetten können.

25.2 Image.Brush

BEZEICHNUNG

`Image.Brush` – zeigt ein Bild an

BESCHREIBUNG

Setzen Sie dieses Attribut auf den Identifikator eines Hollywood-Pinsels oder -Icons, um das Bild anzugeben, das vom Widget angezeigt werden soll. Ob dieses Attribut einen Hollywood-Pinsel oder ein Hollywood-Icon erwartet, hängt davon ab, was Sie im Attribut `Image.BrushType` angeben. Wie der Name schon sagt, erwartet `Image.Brush` standardmäßig einen Hollywood-Pinsel.

Die Image-Klasse unterstützt Masken- und Alphakanaltransparenz in Hollywood-Pinseln/-Icons, so dass Sie auch Bilder mit transparenten Bereichen einbetten können.

Beachten Sie, dass RapaGUI das Bild möglicherweise skaliert, um es an die DPI-Einstellung des aktuellen Monitors anzupassen. Bitte lesen Sie das Kapitel über High-DPI-Unterstützung für weitere Informationen. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), Seite 21, für Details.

TYP

Zahl

ANWENDBARKEIT

IS

25.3 Image.BrushScale

BEZEICHNUNG

`Image.BrushScale` – konfiguriert die automatische Bildskalierung (V2.0)

BESCHREIBUNG

Wenn `Image.Brush` auf einen Rasterpinsel gesetzt wurde und RapaGUI auf einem High-DPI-Display ausgeführt wird, skaliert RapaGUI automatisch die Rastergrafiken des Pinsels, um ihn an die DPI-Einstellung des aktuellen Monitors anzupassen. Wenn Sie dies nicht möchten, setzen Sie diesen Tag auf `False`.

Alternativ können Sie die automatische Bildskalierung auch global deaktivieren, indem Sie beim Aufrufen vom Befehl `@REQUIRE` mit RapaGUI den Tag `ScaleGUI` auf `False` setzen. Siehe [Abschnitt 3.4 \[Initialisierung von RapaGUI\]](#), Seite 11, für Details.

Bitte lesen Sie auch das Kapitel über High-DPI-Unterstützung in RapaGUI, um mehr über die Unterstützung von High-DPI-Displays zu erfahren. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), Seite 21, für Details.

TYP

Boolesch

ANWENDBARKEIT

I

25.4 Image.BrushType

BEZEICHNUNG

Image.BrushType – setzt den zu verwendenden Bildtyp (V2.0)

BESCHREIBUNG

Mit diesem Attribut können Sie den Typ des Hollywood-Bildobjekts festlegen, das im Attribut `Image.Brush` übergeben wird. Standardmäßig erwartet `Image.Brush` einen Hollywood-Pinsel. Durch Setzen von `Image.BrushType` können Sie jedoch einen anderen Hollywood-Bildtyp verwenden.

Die folgenden Bildtypen sind derzeit verfügbar:

- | | |
|--------------|--|
| Brush | Verwendet einen Hollywood-Pinsel. Dies ist der Standardtyp. Sie können entweder Raster- oder Vektorpinsel verwenden. Vektorpinsel haben den Vorteil, dass sie ohne Qualitätsverlust auf jede beliebige Auflösung skaliert werden können. Dies ist sehr nützlich, wenn Sie Programme entwerfen, die mit High-DPI-Monitoren kompatibel sein sollen. Siehe Abschnitt 3.12 [High-DPI-Unterstützung] , Seite 21, für Details. |
| Icon | Verwendet ein Hollywood-Icon. Dieser Bildtyp hat den Vorteil, dass er mehrere Teilbilder unterschiedlicher Größe enthalten kann. Dadurch ist es möglich, Bilder in unterschiedlichen Auflösungen bereitzustellen, was bei der Entwicklung von Programmen, die mit High-DPI-Monitoren kompatibel sein sollen, sehr nützlich sein kann. Siehe Abschnitt 3.12 [High-DPI-Unterstützung] , Seite 21, für Details. |

Beachten Sie, dass Sie den Standardwert dieses Attributs global auf Hollywood-Icon ändern können, indem Sie den Tag `Application.UseIcons` setzen. Siehe [Abschnitt 9.13 \[Application.UseIcons\]](#), Seite 70, für Details.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

I

26 Label-Klasse (Beschriftung)

26.1 Übersicht

Die Label-Klasse (Beschriftung) leitet sich von der Area-Klasse ab und kann verwendet werden, um komfortabel Beschriftungen für Ihre Widgets zu erstellen. Beschriftungen enthalten typischerweise ein Unterstrichzeichen, welches das Tastaturkürzel für den Zugriff auf das beschreibende Widget angibt.

Hier ist ein Beispiel für die Verwendung von `<label>`:

```
<hgroup>
  <label>_Name</label>
  <textentry/>
</hgroup>
```

Der obige Code richtet ein Texteingabe-Widget (Textentry) ein, das mit "Name" beschriftet ist. Außerdem wird eine Verknüpfung eingerichtet: Wenn der Benutzer **ALT+N** drückt, wird das Texteingabe-Widget automatisch aktiviert. Siehe [Abschnitt 3.13 \[Tastaturkürzel\]](#), [Seite 22](#), für Details.

Bitte beachten Sie, dass die Beschriftungen horizontal nicht skalierbar sind. Das ist der Grund, warum sie die Größenanpassung der gesamten GUI leicht blockieren können. Um dieses Problem zu umgehen, können Sie das Label einfach in einer `<hgroup>` zusammen mit einem Rectangle-Objekt `<rectangle>` (Rechteck) einfügen. Siehe [Abschnitt 41.1 \[Rectangle-Klasse\]](#), [Seite 221](#), für Details.

26.2 Label.Align

BEZEICHNUNG

Label.Align – setzt die Ausrichtung der Beschriftung

BESCHREIBUNG

Stellen Sie die gewünschte Ausrichtung für das Label ein. Dies kann einer der folgenden Werte sein:

- Left** Links ausgerichtet.
- Right** Rechts ausgerichtet. Dies ist auch voreingestellt.
- Center** Zentriert ausgerichtet.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

I

26.3 Label.Text

BEZEICHNUNG

Label.Text – setzt den Text der Beschriftung oder gibt ihn zurück

BESCHREIBUNG

Setzen oder ermitteln Sie den Text der Beschriftung. Wenn der Text einen Unterstrich enthält, wird das Zeichen hinter dem Unterstrich automatisch als Tastaturkürzel hervorgehoben. Sie können dieses Verhalten deaktivieren, indem Sie das Attribut `Area.NoAutoKey` auf `True` setzen.

TYP

Zeichenkette

ANWENDBARKEIT

SG

27 Listview-Klasse (Listenansicht)

27.1 Übersicht

Die Listview-Klasse (Listenansicht) leitet sich von der Area-Klasse ab und zeigt einen Listencontainer, der mit Datenelementen gefüllt werden kann. Die Listview-Klasse von RapGUI ist sehr mächtig und unterstützt mehrspaltige Listen (siehe weiter unten), Auswahlkästchen (Checkboxes), editierbare Listenelemente, Datensortierung über benutzerdefinierte Callback-Funktionen und Icons für die einzelnen Listenelemente.

Wenn Sie eine Listenansicht im XML-Code erstellen, müssen Sie immer mindestens eine Spalte hinzufügen. Dies geschieht über die Listviewcolumn-Klasse (Listenansichtsspalten). Hier ist ein Beispiel für eine minimale Listenansicht-Definition mit nur einer einzigen Spalte:

```
<listview>
  <column/>
</listview>
```

Es ist auch möglich, einige Einträge direkt während der Definition in die Listenansicht einzufügen. Dies kann mit dem Tag `<item>` geschehen:

```
<listview>
  <column>
    <item>Entry 1</item>
    <item>Entry 2</item>
    <item>Entry 3</item>
  </column>
</listview>
```

Wenn Sie eine mehrspaltige Liste haben wollen, müssen Sie den Tag `<column>` mehrmals verwenden. Hier ist ein Beispiel:

```
<listview>
  <column title="Column 1">
    <item>Entry 1</item>
    <item>Entry 2</item>
    <item>Entry 3</item>
  </column>
  <column title="Column 2">
    <item>Entry 1</item>
    <item>Entry 2</item>
    <item>Entry 3</item>
  </column>
  <column title="Column 3">
    <item>Entry 1</item>
    <item>Entry 2</item>
    <item>Entry 3</item>
  </column>
</listview>
```

In diesem Beispiel haben wir auch das Attribut `Listviewcolumn.Title` verwendet, um jeder unserer Spalten eine Titelleiste hinzuzufügen. Es gibt noch einige weitere Attribute, mit

denen Sie das Aussehen Ihrer Spalten anpassen können. Beispielsweise können Sie Ihren Spalten Auswahlkästchen (Checkboxes) hinzufügen und die Bearbeitung von Spaltenelementen erlauben. Siehe [Abschnitt 28.1 \[ListViewcolumn-Klasse\]](#), [Seite 171](#), für Details.

Beachten Sie, dass RapaGUI für diese Klasse bis zu drei verschiedene Arten von Widgets verwenden kann: Wenn Sie eine Listenansicht erstellen, die keine erweiterten Eigenschaften verwendet (z.B. mehrere Spalten, Symbole, Sortierung), erstellt RapaGUI möglicherweise ein einfacheres Listenansichts-Widget für Sie, weil einige Betriebssysteme verschiedene Arten von listenbasierten Widgets anbieten. So gibt es z.B. unter Windows ein Listbox-Widget und ein Listenansichts-Widget. RapaGUI verwendet das Listbox-Widget, falls Ihre Listenansicht keine der erweiterten Eigenschaften verwendet, da Listbox-Widgets normalerweise schneller sind als Listenansichts-Widgets. Wenn Sie das nicht möchten, können Sie RapaGUI erzwingen, Ihnen immer eine vollständige Listenansicht zu geben, indem Sie das Attribut `ListView.ForceMode` auf den entsprechenden Tag setzen. Siehe [Abschnitt 27.17 \[ListView.ForceMode\]](#), [Seite 153](#), für Details.

27.2 ListView.AbortEditing

BEZEICHNUNG

`ListView.AbortEditing` – benachrichtigt, wenn der Benutzer die Bearbeitung abbricht (V1.1)

BESCHREIBUNG

Wenn Sie eine Benachrichtigung für dieses Attribut einrichten, führt RapaGUI Ihre Callback-Funktion aus, wenn der Benutzer einen Bearbeitungsvorgang an einem Element abbricht, z.B. durch Drücken der Escape-Taste oder durch Klicken außerhalb des gerade bearbeitenden Element-Widgets.

Beachten Sie, dass Sie `ListViewcolumn.Editable` auf `True` setzen müssen, bevor Sie dieses Attribut verwenden können.

Ihre Callback-Funktion wird mit den folgenden zusätzlichen Argumenten aufgerufen:

Row: Zeilenindex des Eintrags, den der Benutzer bearbeitet hat, als er den Vorgang abbrach.

Column: Spaltenindex des Eintrags, den der Benutzer bearbeitet hat, als er den Vorgang abbrach.

Siehe [Abschnitt 3.7 \[Benachrichtigungen der Attribute\]](#), [Seite 14](#), für Details.

TYP

Boolesch

ANWENDBARKEIT

N

27.3 ListView.Active

BEZEICHNUNG

`ListView.Active` – setzt oder ermittelt den aktiven Listeneintrag

BESCHREIBUNG

Mit diesem Attribut kann der aktive Listeneintrag gesetzt oder ermittelt werden. Dieser liegt immer zwischen 0 und `Listview.Entries-1` oder dem speziellen Wert `-1`, falls es momentan keinen aktiven Eintrag gibt.

Wenn Sie dieses Attribut setzen, scrollt die Listenansicht automatisch die Position des angegebenen Eintrags in die Ansicht.

Neben einem absoluten Index können Sie hier auch die folgenden speziellen Werte übergeben:

<code>Off</code>	Löscht die Auswahl.
<code>Top</code>	Markiert den ersten Eintrag.
<code>Bottom</code>	Markiert den letzten Eintrag.
<code>Up</code>	Markiert den vorherigen Eintrag.
<code>Down</code>	Markiert den nächsten Eintrag.
<code>PageUp</code>	Bewegt den Listencursor eine Seite nach oben.
<code>PageDown</code>	Bewegt den Listencursor eine Seite nach unten.

Sie können auch eine Benachrichtigung für dieses Attribut einrichten, um bei jeder Änderung des aktiven Eintrags benachrichtigt zu werden.

TYP

Zahl oder Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

ISGN

27.4 Listview.Alternate

BEZEICHNUNG

`Listview.Alternate` – verwendet wechselnde Zeilenfarben

BESCHREIBUNG

Setzen Sie dieses Attribut auf `True`, damit die Listenansicht mit wechselnden Zeilenfarben erscheint.

Beachten Sie, dass dieses Attribut auf AmigaOS und kompatiblen Systemen mindestens MUI 5.0 erfordert.

TYP

Boolesch

ANWENDBARKEIT

I

27.5 Listview.Clear

BEZEICHNUNG

Listview.Clear – löscht alle Listenansichts-Einträge

ÜBERSICHT

```
moai.DoMethod(id, "Clear")
```

BESCHREIBUNG

Diese Methode entfernt alle Einträge aus der Listenansicht.

EINGABEN

id ID des Listenansichts-Objekts

27.6 Listview.ClickColumn

BEZEICHNUNG

Listview.ClickColumn – ermittelt, auf welche Spalte zuletzt geklickt wurde

BESCHREIBUNG

In mehrspaltigen Listenansichten ermittelt dieses Attribut die Nummer der Spalte, auf die der Benutzer zuletzt geklickt hat.

TYP

Zahl

ANWENDBARKEIT

GN

27.7 Listview.Columns

BEZEICHNUNG

Listview.Columns – ermittelt die Anzahl der Spalten (V2.0)

BESCHREIBUNG

Ruft die Anzahl der Listenansichtsspalten ab.

TYP

Zahl

ANWENDBARKEIT

G

27.8 Listview.CompareItems

BEZEICHNUNG

Listview.CompareItems – legt fest, wie die Einträge sortiert werden sollen

BESCHREIBUNG

Wenn Sie eine Benachrichtigung für dieses Attribut einrichten, führt RapaGUI die Callback-Funktion aus, wenn die Listeneinträge sortiert werden müssen. Die Callback-Funktion erhält zwei Einträge als Argumente und muss bestimmen, welcher Eintrag zuerst eingefügt werden soll.

Ihre Callback-Funktion wird mit den folgenden zusätzlichen Argumenten aufgerufen:

Entry1: Den ersten Eintrag.

Entry2: Den zweiten Eintrag.

Ihre Callback-Funktion muss dann einen Wert zurückgeben, der angibt, wie die beiden Einträge in der Listenansicht ausgerichtet werden sollen. Soll der Eintrag 1 vor 2 gesetzt werden, muss die Callback-Funktion -1 zurückgegeben. Soll Eintrag 1 nach Eintrag 2 gesetzt werden, muss von Ihrer Callback-Funktion 1 zurückkommen. Wenn die beiden Einträge gleich sind, gibt die Callback-Funktion 0 zurück.

Siehe [Abschnitt 3.7 \[Benachrichtigungen der Attribute\]](#), Seite 14, für Details.

TYP

Boolesch

ANWENDBARKEIT

N

27.9 Listview.DefClickColumn

BEZEICHNUNG

Listview.DefClickColumn – setzt die Standardspalte

PLATTFORMEN

Nur AmigaOS und kompatible Betriebssysteme

BESCHREIBUNG

Wenn Sie die Listenansicht über die Tastatur steuern und RETURN drücken, wird die hier eingestellte Spaltennummer als Standard für Listview.ClickColumn verwendet.

TYP

Zahl

ANWENDBARKEIT

ISG

27.10 Listview.DoubleClick

BEZEICHNUNG

Listview.DoubleClick – benachrichtigt, wenn auf Listenansichts-Einträge Doppelgeklickt wird

BESCHREIBUNG

Richten Sie eine Benachrichtigung für dieses Attribut ein, um bei Doppelklicks auf Listeneinträge zu reagieren.

TYP

Boolesch

ANWENDBARKEIT

N

27.11 Listview.DropFile

BEZEICHNUNG

Listview.DropFile – benachrichtig, wenn Dateien auf die Listenansicht gezogen werden (V1.1)

BESCHREIBUNG

Wenn Sie eine Benachrichtigung für dieses Attribut einrichten, führt RapaGUI Ihre Callback-Funktion aus, sobald eine oder mehrere Dateien auf das Listenansichts-Widget gezogen wurden. Das Feld `TriggerValue` der Nachrichtentabelle wird auf eine Tabelle gesetzt, die eine Liste aller Dateien enthält, die auf das Widget gezogen wurden.

Zusätzlich enthält die Nachrichtentabelle die folgenden zwei Felder:

- X:** Die x-Position, an der der Benutzer die Datei(en) abgelegt hat. Dies ist relativ zur linken Ecke des Widgets.
- Y:** Die y-Position, an der der Benutzer die Datei(en) abgelegt hat. Dies ist relativ zur oberen Ecke des Widgets.

Siehe [Abschnitt 3.7 \[Benachrichtigungen der Attribute\]](#), Seite 14, für Details.

Beachten Sie, dass `Listview.DropFile` nur dann ausgelöst wird, wenn `Listview.DropTarget` zuerst auf `True` gesetzt wurde.

TYP

Boolesch

ANWENDBARKEIT

N

27.12 Listview.DropTarget

BEZEICHNUNG

Listview.DropTarget – stellt das Ablegen von Dateien ein (V1.1)

BESCHREIBUNG

Setzen Sie dies auf `True`, wenn Dateien auf diesem Listenansichts-Widget abgelegt werden können. Sie können das `Listview.DropFile` Attribut überwachen, um zu erfahren, wenn der Benutzer eine oder mehrere Dateien auf dem Listenansicht-Widget ablegt.

TYP

Boolesch

ANWENDBARKEIT

ISG

27.13 Listview.Edit

BEZEICHNUNG

Listview.Edit – fordert den Benutzer auf, einen Eintrag zu bearbeiten

ÜBERSICHT

```
moai.DoMethod(id, "Edit", row, column)
```

BESCHREIBUNG

Diese Methode kann verwendet werden, um die Bearbeitung von Einträgen vom Programm aus zu starten. Normalerweise wird die Bearbeitung eines Eintrags vom Benutzer durch einen langsamen Doppelklick auf einen Eintrag gestartet. Diese Methode stellt eine Alternative zu diesem Benutzermechanismus dar.

Dies funktioniert nur, wenn `Listviewcolumn.Editable` für die jeweilige Listenansichtsspalte auf `True` gesetzt wurde.

Wenn der Benutzer die Bearbeitung beendet hat, wird das Attribut `Listview.ValueChange` ausgelöst.

Bitte beachten Sie, dass wenn Sie eine Überwachung auf dem Attribut `Listview.StartEditing` installiert haben, dann wird diese Callback-Funktion zuerst um Erlaubnis gefragt, bevor die Bearbeitung tatsächlich gestartet wird.

Um zu erfahren, wenn Bearbeitungsvorgänge abgebrochen werden, können Sie das Attribut `Listview.AbortEditing` überwachen.

Beachten Sie auch, dass diese Methode unter AmigaOS und kompatiblen Betriebssystemen nur unter MUI 4.0 oder höher verfügbar ist.

EINGABEN

<code>id</code>	ID des Listview-Objekts
<code>row</code>	Zeilenindex des zu bearbeitenden Eintrags
<code>column</code>	Spaltenindex des zu bearbeitenden Eintrags

27.14 Listview.Entries

BEZEICHNUNG

Listview.Entries – ermittelt die Anzahl der Listeneinträge

BESCHREIBUNG

Gibt die aktuelle Anzahl der Einträge in der Listenansicht zurück.

TYP

Zahl

ANWENDBARKEIT

G

27.15 Listview.Exchange

BEZEICHNUNG

Listview.Exchange – tauscht zwei Einträge aus

ÜBERSICHT

```
moai.DoMethod(id, "Exchange", pos1, pos2)
```

BESCHREIBUNG

Diese Methode tauscht zwei Einträge in einer Listenansicht aus. Die Positionen können entweder als absolute Werte von 0 bis Listview.Entries-1 oder als einer der folgenden Sonderwerte übergeben werden:

- Top** Verwendet den ersten Eintrag.
- Active** Verwendet den aktiven Eintrag.
- Bottom** Verwendet den letzten Eintrag.
- Next** Verwendet den nächsten Eintrag. Dies geht nur für den Parameter pos2.
- Previous** Verwendet den vorherigen Eintrag. Dies geht nur für den Parameter pos2.

Diese Methode kann nur bei nicht sortierbaren Listenansichten verwendet werden, da die Elementreihenfolge in sortierbaren Listenansichten feststeht und ausschließlich durch die Einträge in der Spalte bestimmt wird, die aktuell den Sortierfokus hat.

EINGABEN

- id** ID des Listview-Objekts
- pos1** Nummer des ersten Eintrags
- pos2** Nummer des zweiten Eintrags

27.16 Listview.First

BEZEICHNUNG

Listview.First – ermittelt den ersten sichtbaren Eintrag

BESCHREIBUNG

Liefert den ersten sichtbaren Listeneintrag.

TYP

Zahl

ANWENDBARKEIT

SG

27.17 Listview.ForceMode

BEZEICHNUNG

Listview.ForceMode – überschreibt den Standard-Listenansichtmodus

BESCHREIBUNG

RapaGUI kann je nach Einstellung bis zu drei verschiedene Widgets für die Listview-Klasse verwenden. Wenn beispielsweise eine einspaltige Liste ohne Symbole und Überschriften verwendet wird, kann RapaGUI aus Gründen der Effizienz ein anderes Widget verwenden, wenn das Host-Betriebssystem ein solches Widget bereitstellt. Beispielsweise verwendet RapaGUI unter Windows in diesen Fällen das Listbox-Widget anstelle einer voll ausgestatteten Listenansicht. Wenn Sie das nicht wollen, setzen Sie dieses Attribut auf das gewünschte Widget und RapaGUI wird versuchen, es zu verwenden.

Folgende Modi werden derzeit erkannt:

- | | |
|-----------------|---|
| Normal | Wählt automatisch das Widget aus, das am besten passt. Dies ist die Voreinstellung. |
| Listbox | Verwendet ein Listbox-Widget. Listbox-Widgets unterstützen nur eine einzelne Spalte, keine Symbole, keine Überschriften, keine Auswahlkästchen (Checkboxes), keine bearbeitbaren Einträge und keine ausgeblendeten Spalten. |
| Listview | Verwendet ein Listenansichts-Widget. Listenansichts-Widgets unterstützen alles wie Auswahlkästchen, editierbare Einträge, der rechten und zentrierte Ausrichtung und dem Ausblenden von Spalten. |
| Dataview | Verwendet ein Dataview-Widget. Unterstützt alles, verwendet aber derzeit eine generische Implementierung unter Windows. |

Beachten Sie, dass dieses Attribut auf AmigaOS und Kompatiblen keinen Effekt hat, da RapaGUI auf diesen Plattformen immer das gleiche Widget verwendet.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

I

27.18 Listview.GetColumnID

BEZEICHNUNG

Listview.GetColumnID – ermittelt die Spalten-ID (V2.0)

ÜBERSICHT

```
id$ = moai.DoMethod(id, "GetColumnID", pos)
```

BESCHREIBUNG

Gibt den Identifikator der Spalte an der durch `pos` angegebenen Position zurück. `pos` kann entweder ein absoluter Index beginnend bei 0 für die erste Spalte oder einer der folgenden Sonderwerte sein:

- | | |
|--------------|-----------------------------|
| First | Verwendet die erste Spalte. |
|--------------|-----------------------------|

`Last` Verwendet die letzte Spalte.

EINGABEN

`id` ID des Listview-Objekts

`pos` absoluter Index der Spalte oder spezieller Wert (siehe oben)

RÜCKGABEWERTE

`id$` ID der Spalte mit dem angegebenen Index

27.19 Listview.GetDisabled

BEZEICHNUNG

`Listview.GetDisabled` – ermittelt, ob das Auswahlkästchen deaktiviert ist

ÜBERSICHT

```
state = moai.DoMethod(id, "GetDisabled", row, column)
```

BESCHREIBUNG

Gibt den deaktivierten Zustand des Auswahlkästchen (Checkbox) in der angegebenen Zeile `row` und Spalte `column` zurück. Dies ist entweder `True` (deaktiviert) oder `False` (aktiviert).

EINGABEN

`id` ID des Listview-Objekts

`row` Zeilenindex des Auswahlkästchen

`column` Spaltenindex des Auswahlkästchen

RÜCKGABEWERTE

`state` `True` wenn das Auswahlkästchen deaktiviert ist, sonst `False`

27.20 Listview.GetEntry

BEZEICHNUNG

`Listview.GetEntry` – gibt den Listeneintrag zurück

ÜBERSICHT

```
column1$, ... = moai.DoMethod(id, "GetEntry", pos[, icons])
```

BESCHREIBUNG

Holt einen Eintrag aus der in `id` angegebenen Listenansicht. Sie können entweder einen absoluten Index in `pos` oder den speziellen Wert `Active` übergeben, um den aktiven Eintrag zu erhalten. `Listview.GetEntry` gibt dann die Einträge aller Spalten der durch `pos` angegebenen Zeile zurück. Sie erhalten so viele Rückgabewerte, wie es Spalten in der Listenansicht gibt.

Ab RapaGUI 2.0 akzeptiert `Listview.GetEntry` jetzt ein optionales `icons`-Argument. Wenn dies auf `True` gesetzt ist, gibt `Listview.GetEntry` auch den Identifikator des Pinsels/Icons zurück, der für ein Listenelement verwendet wird, oder `-1`, wenn für dieses

Element kein Pinsel/Icon gesetzt wurde. Beachten Sie, dass der Identifikator nur für Spalten zurückgegeben wird, bei denen das Attribut `Listviewcolumn.Icon` auf `True` gesetzt ist.

EINGABEN

`id` ID des Listview-Objekts

`pos` Index der Listenansichts-Zeile oder "Active"

`icons` optional: `True`, wenn auch Informationen über Icons zurückgegeben werden sollen (V2.0)

RÜCKGABEWERTE

`column1$` Daten der ersten Spalte

`...` Weitere Daten bei mehrspaltiger Listenansicht

27.21 Listview.GetSelection**BEZEICHNUNG**

`Listview.GetSelection` – ermittelt die ausgewählten Einträge

ÜBERSICHT

```
t = moai.DoMethod(id, "GetSelection")
```

BESCHREIBUNG

Gibt eine Tabelle mit allen markierten Einträgen einer Mehrfachauswahl-Listenansicht zurück. Beachten Sie, dass dies nur bei Mehrfachauswahllisten verwendet werden sollte. Für einfach ausgewählte Listenansichten können Sie `Listview.Active` verwenden, um den ausgewählten Eintrag zu erhalten.

EINGABEN

`id` ID des Listview-Objekts

RÜCKGABEWERTE

`t` Tabelle mit den ausgewählten Einträgen

27.22 Listview.GetState**BEZEICHNUNG**

`Listview.GetState` – fragt den Umschaltstatus des Auswahlkästchen ab

ÜBERSICHT

```
state = moai.DoMethod(id, "GetState", row, column)
```

BESCHREIBUNG

Gibt den Umschaltstatus des Auswahlkästchen (Checkbox) in der angegebenen Zeile `row` und Spalte `column` zurück. Dies ist entweder `True`, wenn das Auswahlkästchen aktiviert ist, oder `False`.

EINGABEN

<code>id</code>	ID des Listview-Objekts
<code>row</code>	Zeilenindex des Auswahlkästchen
<code>column</code>	Spaltenindex des Auswahlkästchen

RÜCKGABEWERTE

<code>state</code>	True, wenn das Auswahlkästchen markiert ist, ansonsten False
--------------------	--

27.23 Listview.HRules**BEZEICHNUNG**

Listview.HRules – zeichnet horizontale Rasterlinien zwischen den Zeilen

PLATTFORMEN

Windows, Linux, macOS

BESCHREIBUNG

Setzen Sie dies auf True, um horizontale Rasterlinien zwischen den Zeilen der Listenansicht zu aktivieren.

TYP

Boolesch

ANWENDBARKEIT

I

27.24 Listview.Insert**BEZEICHNUNG**

Listview.Insert – fügt einen neuen Eintrag ein

ÜBERSICHT

```
pos = moai.DoMethod(id, "Insert", pos, [icon1,] column1$, ...)
```

BESCHREIBUNG

Fügen Sie einen neuen Eintrag in die Listenansicht ein. Wenn die Listenansicht mehrere Spalten hat, müssen Sie für alle Spalten der Listenansicht individuelle Eingabedaten übergeben.

Die Eingabedaten bestehen aus einer Textzeichenkette und, wenn die Spalte das Attribut `Listviewcolumn.Icon` hat, einem Symbol für jede Spalte. Das Symbol muss vor der Textzeichenkette übergeben werden und es muss ein Identifikator eines Hollywood-Pinsels/-Icon sein, der als Symbol für den Eintrag verwendet werden soll. Wenn `Listviewcolumn.Icon` nicht gesetzt ist, müssen Sie den Parameter `Icon` weglassen und nur Textdaten für den Listenansichts-Eintrag übergeben. Wenn Sie `Listviewcolumn.Icon` auf `True` gesetzt haben und kein Symbol in dieser bestimmten Zeile und Spalte anzeigen möchten, können Sie auch den speziellen Wert `-1` übergeben. In diesem Fall zeigt RapaGUI kein Symbol an, obwohl `Listviewcolumn.Icon` auf `True`

gesetzt wurde. Bitte beachten Sie, dass automatisch generierte IDs nicht verwendet werden können. Bitte lesen Sie auch den Bilder-Cache von RapaGUI, um mehr über die Unterstützung von Symbolen in RapaGUI zu erfahren. Siehe [Abschnitt 3.20 \[Bilder-Cache\]](#), [Seite 32](#), für Details.

Falls eine Spalte ein Auswahlkästchen (Checkbox) zeigt, müssen Sie "On", "True" oder "1" übergeben, um das Auswahlkästchen und einen beliebigen anderen Text zu markieren.

Die Einfügeposition wird im Argument `pos` angegeben. Der neue Eintrag wird vor dem in `pos` angegebenen Eintrag eingefügt. Dies kann eine absolute Indexposition ab 0 für den ersten Eintrag oder einer der folgenden Sonderwerte sein:

<code>Top</code>	Als ersten Eintrag einfügen.
<code>Active</code>	Vor dem aktiven Eintrag einfügen. Wenn es keinen aktiven Eintrag gibt, wird der Eintrag ganz oben in die Liste eingefügt.
<code>Bottom</code>	Als letzten Eintrag einfügen.

Wenn `pos` größer oder gleich der Anzahl der Einträge in der Listenansicht ist, wird der Eintrag als letzter Eintrag eingefügt.

Hat die Listenansicht sortierbare Spalten, wird `pos` ignoriert und der Eintrag wird automatisch in die Listenansicht einsortiert, je nachdem welche Spalte aktuell den Sortierfokus hat. Siehe [Abschnitt 27.40 \[Listview.SortColumn\]](#), [Seite 166](#), für Details.

`Listview.Insert` gibt die Position des neu eingefügten Eintrags zurück. Dies ist besonders nützlich, wenn die Listenansicht sortierbare Spalten hat, weil in diesem Fall nicht einfach berechnen kann, wo der Eintrag in der Listenansicht landen wird.

Beachten Sie, dass unter AmigaOS und kompatiblen die Unterstützung von Symbolen nur mit MUI 4.0 oder höher verfügbar ist.

EINGABEN

<code>id</code>	ID des Listview-Objekts
<code>pos</code>	Position als Absolut- oder Sonderwert einfügen (siehe oben)
<code>icon1</code>	optional: Symbol für die erste Spalte; diese muss nur übergeben werden, wenn die Spalte das Icon-Attribut gesetzt hat; beachten Sie, dass dies ein numerischer Identifikator sein muss und automatisch generierte IDs in diesem Fall nicht gültig sind
<code>column1\$</code>	Eintrag zum Einfügen in die erste Spalte
<code>...</code>	weitere Einträge bei mehrspaltiger Listenansicht

RÜCKGABEWERTE

<code>pos</code>	Position des neu eingefügten Eintrags (ab 0)
------------------	--

27.25 Listview.InsertColumn

BEZEICHNUNG

`Listview.InsertColumn` – fügt eine neue Spalte ein (V2.0)

ÜBERSICHT

```
moai.DoMethod(id, "InsertColumn", pos, cid$, t)
```

BESCHREIBUNG

Fügt eine neue Spalte in die Listenansicht ein. Die Spalte wird an der durch `pos` angegebenen Position eingefügt und erhält den durch `cid$` angegebene Identifikator. Die neue Spalte wird vor der durch `pos` angegebenen Spalte eingefügt. Dies kann eine absolute Indexposition beginnend bei 0 für die erste Spalte oder einer der folgenden Sonderwerte sein:

First Fügt sie als erste Spalte ein.

Last Fügt sie als letzte Spalte ein.

Das optionale Argument `t` kann auf eine Tabelle gesetzt werden, die eine oder mehrere der folgenden Optionen enthält:

Align Gewünschte Spaltenausrichtung. Siehe [Abschnitt 28.2 \[Listviewcolumn.Align\]](#), Seite 171, für Details.

Checkbox Dieses Flag gibt an, ob die Spalte im Auswahlkästchen-Modus verwendet werden soll. Siehe [Abschnitt 28.3 \[Listviewcolumn.Checkbox\]](#), Seite 171, für Details.

Editable Dieses Flag gibt an, ob die Spalte bearbeitbar sein soll. Siehe [Abschnitt 28.4 \[Listviewcolumn.Editable\]](#), Seite 172, für Details.

Hide Dieses Flag gibt an, ob die Spalte ausgeblendet werden soll. Siehe [Abschnitt 28.5 \[Listviewcolumn.Hide\]](#), Seite 173, für Details.

Icon Dieses Flag gibt an, ob in der Spalte Icons verwendet werden sollen. Siehe [Abschnitt 28.6 \[Listviewcolumn.Icon\]](#), Seite 173, für Details.

IconType Icon-Typ, der verwendet werden soll. Siehe [Abschnitt 28.8 \[Listviewcolumn.IconType\]](#), Seite 174, für Details.

IconScale Schaltet die Icon-Skalierung um. Siehe [Abschnitt 28.7 \[Listviewcolumn.IconScale\]](#), Seite 174, für Details.

Sortable Dieses Flag gibt an, ob die Spalte sortierbar sein soll. Siehe [Abschnitt 28.9 \[Listviewcolumn.Sortable\]](#), Seite 175, für Details.

SortOrder Gibt die gewünschte Spaltensortierreihenfolge an. Siehe [Abschnitt 28.10 \[Listviewcolumn.SortOrder\]](#), Seite 175, für Details.

Title Gewünschter Spaltentitel. Siehe [Abschnitt 28.11 \[Listviewcolumn.Title\]](#), Seite 176, für Details.

Width Gewünschte Spaltenbreite. Siehe [Abschnitt 28.12 \[Listviewcolumn.Width\]](#), Seite 176, für Details.

Beachten Sie, dass `Listview.InsertColumn` nur unterstützt wird, wenn sich die Listenansicht im `Listview-` oder `Dataview-`Modus befindet.

EINGABEN

<code>id</code>	ID des ListView-Objekts
<code>pos</code>	fügt die Position als Absolutzahl oder Sonderwert ein (siehe oben)
<code>cid\$</code>	Identifikator für die neue Spalte
<code>t</code>	optional: Tabelle mit weiteren Parametern

27.26 ListView.ItemStyle**BEZEICHNUNG**

ListView.ItemStyle – setzt den Stil der Listenansicht (V2.0)

PLATTFORMEN

Android

BESCHREIBUNG

Unter Android können Sie einen Stil angeben, der für die Elemente der Listenansicht verwendet werden soll. Dies kann einer der folgenden speziellen Werte sein:

<code>Default</code>	Der Standardstil.
<code>Normal</code>	Normaler Listenelementstil. Dies entspricht dem Stil <code>android.R.layout.simple_list_item_1</code> .
<code>Big</code>	Verwendet große Listenelemente. Dies entspricht dem Stil <code>android.R.layout.simple_expandable_list_item_1</code> .

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

I

27.27 ListView.Jump**BEZEICHNUNG**

ListView.Jump – scrollt zu einem Eintrag

ÜBERSICHT

`moai.DoMethod(id, "Jump", pos)`

BESCHREIBUNG

Scrollt den angegebenen Eintrag in den sichtbaren Teil der Listenansicht. Dabei kann `pos` ein absoluter Index oder einer der folgenden Sonderwerte sein:

<code>Top</code>	Scrollt den obersten Eintrag in den sichtbaren Teil.
<code>Active</code>	Scrollt den aktiven Eintrag in den sichtbaren Teil.
<code>Bottom</code>	Scrollt den letzten Eintrag in den sichtbaren Teil.

EINGABEN

<code>id</code>	ID des Listview-Objekts
<code>pos</code>	Nummer des Eintrags, der sichtbar gemacht werden soll oder spezieller Wert (siehe oben)

27.28 Listview.LongClick**BEZEICHNUNG**

Listview.LongClick – benachrichtigt, bei einem langen Klick auf der Listenansicht (V2.0)

PLATTFORMEN

Android

BESCHREIBUNG

Richtet eine Benachrichtigung für dieses Attribut ein, um über lange Klicks auf Listenansichtseinträge informiert zu werden. Lange Klicks werden auf Android derzeit nur ausgelöst, wenn der Nutzer länger auf ein Listenansichtselement tippt.

TYP

Boolesch

ANWENDBARKEIT

N

27.29 Listview.Move**BEZEICHNUNG**

Listview.Move – verschiebt einen Eintrag an die neue Position

ÜBERSICHT

`moai.DoMethod(id, "Move", from, to)`

BESCHREIBUNG

Verschiebt den in `from` angegebenen Eintrag an die neue Position `to`. Positionen müssen als absolute Werte von 0 bis `Listview.Entries-1` oder einen der folgenden speziellen Werte übergeben werden:

<code>Top</code>	Verwendet den ersten Eintrag.
<code>Active</code>	Verwendet den aktiven Eintrag.
<code>Bottom</code>	Verwendet den letzten Eintrag.
<code>Next</code>	Verwendet den nächsten Eintrag. Dies geht nur für den Parameter <code>to</code> .
<code>Previous</code>	Verwendet den vorherigen Eintrag. Dies geht nur für den Parameter <code>to</code> .

Diese Methode kann nur bei nicht sortierbaren Listenansichten verwendet werden, da die Elementreihenfolge in sortierbaren Listenansichten fest und ausschließlich durch die Einträge in der Spalte bestimmt wird, die aktuell den Sortierfokus hat.

EINGABEN

<code>id</code>	ID des Listview-Objekts
<code>from</code>	Nummer des ersten Eintrags
<code>to</code>	Nummer des zweiten Eintrags

27.30 Listview.MultiSelect

BEZEICHNUNG

Listview.MultiSelect – aktiviert den Mehrfachauswahl-Modus

BESCHREIBUNG

Setzen Sie dieses Attribut auf `True`, um die Auswahl mehrerer Einträge in dieser Listenansicht zu ermöglichen. Standardmäßig kann immer nur ein Eintrag ausgewählt werden.

TYP

Boolesch

ANWENDBARKEIT

I

27.31 Listview.Quiet

BEZEICHNUNG

Listview.Quiet – deaktiviert die Aktualisierung der Listenansicht

BESCHREIBUNG

Das Hinzufügen oder Entfernen vieler Einträge auf einmal ist in der Regel recht aufwendig, da das Widget nach jedem Hinzufügen und Entfernen aktualisiert wird. Wenn Sie `Listview.Quiet` auf `True` setzen, wird die Aktualisierung der Listenansicht vorübergehend deaktiviert, bis Sie das Attribut wieder auf `False` setzen. Dies kann das Einfügen und Entfernen von Operationen erheblich beschleunigen, wenn viele Einträge betroffen sind.

TYP

Boolesch

ANWENDBARKEIT

S

27.32 Listview.Remove

BEZEICHNUNG

Listview.Remove – entfernt einen Eintrag aus der Listenansicht

ÜBERSICHT

```
moai.DoMethod(id, "Remove", pos)
```

BESCHREIBUNG

Entfernt einen Eintrag aus einer Listenansicht. Die Position kann als absoluter Indexwert oder als einer der folgenden Sonderwerte angegeben werden:

First Entfernt den ersten Eintrag.

Active Entfernt den aktiven Eintrag.

Last Entfernt den letzten Eintrag.

Wenn der aktive Eintrag entfernt ist, wird der folgende Eintrag aktiv.

EINGABEN

id ID des Listview-Objekts

pos Index des zu entfernenden Eintrags oder einer der speziellen Werte (siehe oben)

27.33 Listview.RemoveColumn**BEZEICHNUNG**

Listview.RemoveColumn – entfernt eine Spalte (V2.0)

ÜBERSICHT

```
moai.DoMethod(id, "RemoveColumn", pos)
```

BESCHREIBUNG

Entfernt die Spalte am Index `pos` aus der Listenansicht. `pos` kann eine absolute Indexposition beginnend bei 0 für die erste Spalte oder einer der folgenden speziellen Werte sein:

First Entfernt die erste Spalte.

Last Entfernt die letzte Spalte.

Beachten Sie, dass `Listview.RemoveColumn` nur unterstützt wird, wenn sich die Listenansicht im Modus `Listview` oder `Dataview` befindet. Siehe [Abschnitt 27.17 \[Listview.ForceMode\]](#), Seite 153, für Details.

EINGABEN

id ID des Listview-Objekts

pos Position zum Entfernen als absoluten Index oder speziellen Wert (siehe oben)

27.34 Listview.Rename**BEZEICHNUNG**

Listview.Rename – benennt einen Eintrag um

ÜBERSICHT

```
moai.DoMethod(id, "Rename", pos, [icon1,] column1$, ...)
```

BESCHREIBUNG

Benennt den Listeneintrag an der angegebenen Position `pos` um. Wenn die Listenansicht mehrere Spalten hat, müssen Sie für jede Spalte einen neuen Namen übergeben. Es ist nicht möglich, nur einen einzelnen Spalteneintrag umzubenennen; diese Methode wirkt sich immer auf die gesamte Zeile aus, so dass Sie so viele Zeichenketten übergeben müssen, wie es Spalten in Ihrer Listenansicht gibt.

Für alle Spalten, die das Attribut `Listviewcolumn.Icon` gesetzt haben, müssen Sie auch ein Symbol vor dem eigentlichen Text übergeben, welches ein Identifikator eines Hollywood-Pinsels/-Icons sein muss. Wenn `Listviewcolumn.Icon` nicht gesetzt ist, müssen Sie den Parameter `icon` weglassen und nur Textdaten für den Listenansichtsintrag übergeben. Wenn Sie `Listviewcolumn.Icon` auf `True` gesetzt haben und kein Symbol in dieser bestimmten Zeile und Spalte anzeigen möchten, können Sie auch den speziellen Wert `-1` übergeben. In diesem Fall zeigt RapaGUI kein Symbol an, obwohl `Listviewcolumn.Icon` auf `True` gesetzt wurde. Beachten Sie, dass automatisch generierte IDs nicht verwendet werden können. Bitte lesen Sie auch den Bilder-Cache von RapaGUI, um mehr über die Unterstützung von Symbolen in RapaGUI zu erfahren. Siehe [Abschnitt 3.20 \[Bilder-Cache\]](#), [Seite 32](#), für Details.

Falls eine Spalte ein Auswahlkästchen (Checkbox) zeigt, müssen Sie `"On"`, `"True"` oder `"1"` übergeben, um das Auswahlkästchen und einen beliebigen anderen Text zu markieren.

Die Einfügeposition wird im Argument `pos` angegeben. Dies kann eine absolute Indexposition ab 0 für den ersten Eintrag oder einer der folgenden Sonderwerte sein:

`Active` Benennt den aktiven Eintrag um.

Beachten Sie, dass unter AmigaOS und kompatiblen die Unterstützung von Symbolen nur mit MUI 4.0 oder höher verfügbar ist.

EINGABEN

<code>id</code>	ID des Listview-Objekts
<code>pos</code>	Eingabeposition als absolute Zahl oder Sonderwert (siehe oben)
<code>icon1</code>	optional: Symbol für die erste Spalte; diese muss nur übergeben werden, wenn die Spalte das Icon-Attribut gesetzt hat; beachten Sie, dass dies ein numerischer Identifikator sein muss und automatisch generierte IDs in diesem Fall nicht gültig sind
<code>column1\$</code>	neuer Text für den Eintrag in der ersten Spalte
<code>...</code>	weitere Einträge bei mehrspaltiger Listenansicht

27.35 Listview.RowHeight**BEZEICHNUNG**

`Listview.RowHeight` – setzt die Zeilenhöhe (V2.0)

BESCHREIBUNG

Legt die Zeilenhöhe der Listenansicht fest. Dies ist normalerweise nicht erforderlich, da die Zeilenhöhe automatisch berechnet wird. Es kann jedoch nützlich sein, eine bestimmte

Zeilenhöhe festzulegen, wenn Sie sehr große Symbole verwenden, die sonst abgeschnitten würden.

TYP

Zahl

ANWENDBARKEIT

I

27.36 Listview.Select**BEZEICHNUNG**

Listview.Select – markiert einen Listeneintrag oder wählt/fragt ihn ab

ÜBERSICHT

```
state = moai.DoMethod(id, "Select", pos, seltype)
```

BESCHREIBUNG

Markiert einen Listeneintrag, wählt ihn ab oder Sie können den Auswahlzustand eines Eintrags abfragen.

Entweder kann `pos` die Nummer des Eintrags oder einer der folgenden Sonderwerte sein:

Active Verwendet den aktiven Eintrag.

All Verwendet alle Einträge.

Einer der folgenden Werte kann `seltype` sein:

Off Wählt den Eintrag ab.

On Markiert den Eintrag.

Toggle Wechselt den Auswahlzustand für den Eintrag.

Ask Ermittelt den Auswahlzustand des angegebenen Eintrags. Wenn dies festgelegt ist, gibt Listview.Select den Auswahlstatus des angegebenen Eintrags (1 oder 0) zurück.

EINGABEN

`id` ID des Listview-Objekts

`pos` Eingabeposition als absolute Zahl oder Sonderwert (siehe oben)

`seltype` Selektionsart (siehe oben)

RÜCKGABEWERTE

`state` Auswahlzustand des Eintrags; dies gilt nur bei Verwendung von "Ask" für `seltype` (siehe oben)

27.37 Listview.SetDisabled

BEZEICHNUNG

Listview.SetDisabled – setzt den Deaktivierungsstatus des Auswahlkästchen

ÜBERSICHT

```
moai.DoMethod(id, "SetDisabled", row, column, state)
```

BESCHREIBUNG

Setzt den deaktivierten Zustand des Auswahlkästchen (Checkbox) in der angegebenen Zeile `row` und Spalte `column`. Übergeben Sie `True`, um das Auswahlkästchen zu deaktivieren, oder `False`, um sie zu aktivieren.

EINGABEN

<code>id</code>	ID des Listview-Objekts
<code>row</code>	Zeilenindex des Auswahlkästchen
<code>column</code>	Spaltenindex des Auswahlkästchen
<code>state</code>	<code>True</code> , wenn das Auswahlkästchen deaktiviert werden soll, andernfalls <code>False</code> .

27.38 Listview.SetState

BEZEICHNUNG

Listview.SetState – setzt den Umschaltstatus des Auswahlkästchen

ÜBERSICHT

```
moai.DoMethod(id, "SetState", row, column, state)
```

BESCHREIBUNG

Setzt den Umschaltstatus des Auswahlkästchen (Checkbox) in der angegebenen Zeile `row` und Spalte `column`. Übergeben Sie `True`, um das Auswahlkästchen zu markieren, oder `False`, um sie abzuwählen.

EINGABEN

<code>id</code>	ID des Listview-Objekts
<code>row</code>	Zeilenindex des Auswahlkästchen
<code>column</code>	Spaltenindex des Auswahlkästchen
<code>state</code>	<code>True</code> wenn das Auswahlkästchen markiert werden soll, andernfalls <code>False</code>

27.39 Listview.Sort

BEZEICHNUNG

Listview.Sort – sortiert die Einträge

ÜBERSICHT

```
moai.DoMethod(id, "Sort")
```

BESCHREIBUNG

Diese Methode sortiert alle Einträge der Listenansicht. Normalerweise ist es nicht notwendig, diese Methode aufzurufen, da die Listenansicht automatisch alle Elemente sortiert, sobald es eine sortierbare Spalte gibt.

EINGABEN

`id` ID des Listview-Objekts

27.40 Listview.SortColumn**BEZEICHNUNG**

Listview.SortColumn – setzt/ermittelt die Sortierspalte (V2.0)

BESCHREIBUNG

Setzt oder ermittelt die Spalte, die zum Sortieren der Listeneinträge verwendet werden soll. Diese muss auf den Index der Spalte gesetzt werden, die den Sortierfokus erhalten soll. Spaltenindizes beginnen bei 0 für die erste Spalte. Beachten Sie, dass Sie nur Spalten als Sortierspalten festlegen können, bei denen `Listviewcolumn.Sortable` auf `True` gesetzt ist. Die Sortierreihenfolge kann durch Setzen des Attributs `Listviewcolumn.SortOrder` konfiguriert werden.

Beachten Sie, dass der Benutzer die Sortierspalte auch einstellen kann, indem er auf ihre Überschrift klickt. Wenn Sie über solche Ereignisse benachrichtigt werden möchten, können Sie eine Benachrichtigung für dieses Attribut installieren.

TYP

Zahl

ANWENDBARKEIT

ISGN

27.41 Listview.StartEditing**BEZEICHNUNG**

Listview.StartEditing – benachrichtigt, wenn ein Eintrag bearbeitet wird

BESCHREIBUNG

Wenn Sie eine Benachrichtigung für dieses Attribut einrichten, führt RapaGUI Ihre Callback-Funktion aus, wenn der Benutzer ein Listenelement durch langsames Doppelklicken bearbeiten möchte oder wenn Ihr Skript die Methode `Listview.Edit` ausführt. Ihre Callback-Funktion kann dann den Bearbeitungsauftrag des Benutzers erlauben oder verbieten. Um die Anfrage zu verbieten, muss Ihre Callback-Funktion `False` oder um die Anfrage zu erlauben, muss sie `True` zurückgeben.

Beachten Sie, dass Sie `Listviewcolumn.Editable` auf `True` setzen müssen, bevor Sie dieses Attribut verwenden können.

Ihre Callback-Funktion wird mit den folgenden zusätzlichen Argumenten aufgerufen:

Row: Zeilenindex des Eintrags, den der Benutzer bearbeiten möchte.

Column: Spaltenindex des Eintrags, den der Benutzer bearbeiten möchte.

Siehe [Abschnitt 3.7 \[Benachrichtigungen der Attribute\]](#), Seite 14, für Details.

TYP

Boolesch

ANWENDBARKEIT

N

27.42 Listview.SystemTheme

BEZEICHNUNG

Listview.SystemTheme – verwendet das System-Design für die Listenansicht (V2.0)

PLATTFORMEN

Windows

BESCHREIBUNG

Setzen Sie dies auf `True`, damit die Listenansicht das Design des Systems verwendet. Dies wird nur unterstützt, wenn `Listview.ForceMode` auf `Listview` gesetzt ist oder wenn RapaGUI implizit `Listview.ForceMode` auf `Listview` setzt.

TYP

Boolesch

ANWENDBARKEIT

I

27.43 Listview.TitleClick

BEZEICHNUNG

Listview.TitleClick – legt die Spaltenindexnummer ab

BESCHREIBUNG

Dieses Attribut wird auf die Spaltenindexnummer gesetzt, wenn der Benutzer auf eine Spaltentitel-Schaltfläche klickt.

Unter AmigaOS und kompatiblen Betriebssystemen erfordert dieses Attribut mindestens MUI 4.0.

TYP

Zahl

ANWENDBARKEIT

N

27.44 Listview.ValueChange

BEZEICHNUNG

Listview.ValueChange – benachrichtigt, wenn sich der Wert des Listeneintrags ändert

BESCHREIBUNG

Wenn Sie eine Benachrichtigung für dieses Attribut einrichten, führt RapaGUI Ihre Callback-Funktion aus, wenn sich der Wert eines Listeneintrags geändert hat, weil der Benutzer das Auswahlkästchen (Combobox) umgeschaltet oder das Element bearbeitet hat. Beachten Sie, dass `Listview.ValueChange` nicht ausgelöst wird, wenn der Wert des Elements mit der Methode `Listview.Rename` geändert wurde.

Für Elemente in Auswahlkästchen-Spalten wird `TriggerValue` entweder auf `True` oder `False` gesetzt, was den neuen Auswahlkästchen-Status widerspiegelt. Für Elemente in Textspalten enthält `TriggerValue` den neuen Eintragstext.

Zusätzlich wird Ihre Callback-Funktion mit den folgenden zusätzlichen Argumenten aufgerufen:

Row: Zeilenindex des Eintrags, dessen Wert sich geändert hat.

Column: Spaltenindex des Eintrags, dessen Wert sich geändert hat.

Siehe [Abschnitt 3.7 \[Benachrichtigungen der Attribute\]](#), Seite 14, für Details.

TYP

Boolesch oder Zeichenkette (abhängig vom Spaltentyp)

ANWENDBARKEIT

N

27.45 Listview.Visible

BEZEICHNUNG

Listview.Visible – ermittelt die Anzahl der sichtbaren Einträge

BESCHREIBUNG

Ermittelt die Anzahl der Einträge in der Listenansicht, die gerade sichtbar sind.

TYP

Zahl

ANWENDBARKEIT

G

27.46 Listview.VRules

BEZEICHNUNG

Listview.VRules – zeichnet vertikale Rasterlinien zwischen den Spalten

BESCHREIBUNG

Setzen Sie dies auf `True`, um vertikale Rasterlinien zwischen den Spalten der Listenansicht zu aktivieren.

TYP

Boolesch

ANWENDBARKEIT

I

28 Listviewcolumn-Klasse (Listenansichtsspalten)

28.1 Übersicht

Die Listviewcolumn-Klasse (Listenansichtsspalten) wird beim Erstellen von Listenansichten benötigt. Es erlaubt Ihnen, verschiedene Attribute für die Spalten Ihrer Listenansichten anzugeben.

Die Listviewcolumn-Klasse muss immer in eine `<listview>`-Definition eingebettet werden. Der XML-Tag ist `<column>`. Siehe [Abschnitt 27.1 \[Listview-Klasse\], Seite 145](#), für Details.

Beachten Sie, dass Sie keine Instanzen dieser Klasse mit `moai.CreateObject()` erzeugen können. Die Spaltennummern der Listenansicht sind derzeit statisch, d.h. Sie können zur Laufzeit keine Spalten hinzufügen oder entfernen.

28.2 Listviewcolumn.Align

BEZEICHNUNG

Listviewcolumn.Align – setzt/ermittelt die Spaltenausrichtung

BESCHREIBUNG

Setzt oder ermittelt die Spaltenausrichtung. Dies kann einer der folgenden Werte sein:

Left Links ausgerichtet. Dies ist auch voreingestellt.

Right Rechts ausgerichtet.

Center Zentriert ausgerichtet.

Auf Nicht-AmigaOS-Systemen wird dieses Attribut nur für das DataView-Widget unterstützt. Wenn Sie eine Listenansicht erstellen und Listviewcolumn.Align auf einen anderen Wert als Left gesetzt ist, wechselt RapaGUI automatisch zum DataView-Widget. Wenn Sie dieses Attribut nicht zum Zeitpunkt der Erstellung angeben, sondern es später mit `moai.Set()` festlegen möchten, müssen Sie explizit ein DataView-Widget anfordern, indem Sie das Attribut Listview.ForceMode setzen.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

ISG

28.3 Listviewcolumn.Checkbox

BEZEICHNUNG

Listviewcolumn.Checkbox – setzt die Spalte in den Auswahlkästchen-Modus

BESCHREIBUNG

Setzen Sie dies auf True, um diese Spalte als Auswahlkästchen-Spalte (Checkbox) zu benutzen. Auswahlkästchen-Spalten zeigen Auswahlkästchen anstelle von Text an. Immer wenn der Text eines Eintrags in einer Auswahlkästchen-Spalte auf "On", "True"

oder "1" gesetzt ist, wird das Auswahlkästchen ausgewählt. Alle anderen Texte führen zu einem nicht ausgewähltem Auswahlkästchen.

Sie können die Zustände der Auswahlkästchen entweder mit der Methode `Listview.Rename` oder mit der dedizierten Methode `Listview.SetState` ändern. Ebenso ist es möglich, den Status eines Auswahlkästchen über die Methoden `Listview.GetEntry` oder `Listview.GetState` abzurufen.

Um benachrichtigt zu werden, wenn der Benutzer den Status eines Auswahlkästchen umschaltet, müssen Sie das Attribut `Listview.ValueChange` überwachen.

Beachten Sie auch, dass `Listviewcolumn.Checkbox`, `Listviewcolumn.Editable` und `Listviewcolumn.Icon` sich gegenseitig ausschließen. Sie können keine Auswahlkästchen-Spalten erstellen, die editierbar sind oder Symbole anzeigen können.

TYP

Boolesch

ANWENDBARKEIT

IG

28.4 Listviewcolumn.Editable

BEZEICHNUNG

`Listviewcolumn.Editable` – erlaubt das Bearbeiten von Spalteneinträgen

BESCHREIBUNG

Setzen Sie diesen Wert auf `True`, um die Bearbeitung der Elemente in dieser Spalte durch den Benutzer zu ermöglichen. Der Benutzer kann dann alle Elemente in dieser Spalte bearbeiten, indem er einen langsamen Doppelklick ausführt, d.h. die linke Maustaste zweimal hintereinander langsam drückt.

Wenn Sie nur die Bearbeitung einiger Elemente in der Spalte erlauben möchten, müssen Sie dieses Attribut auf `True` setzen und das Attribut `Listview.StartEditing` überwachen. `Listview.StartEditing` wird immer dann ausgelöst, wenn der Benutzer versucht, ein Element zu bearbeiten und Ihr Callback-Funktion kann `False` zurückgeben, um die Bearbeitung bestimmter Elemente zu verbieten. Siehe [Abschnitt 27.41 \[Listview.StartEditing\]](#), Seite 166, für Details.

Um benachrichtigt zu werden, wenn sich der Wert eines Listeneintrags ändert, weil der Benutzer ihn bearbeitet hat, müssen Sie das Attribut `Listview.ValueChange` überwachen.

Um die Bearbeitung eines Listeneintrags manuell zu starten, rufen Sie die Methode `Listview.Edit` auf.

Auch `Listviewcolumn.Checkbox` und `Listviewcolumn.Editable` schließen sich gegenseitig aus. Sie können keine editierbaren Auswahlkästchen-Spalten (Checkbox) anlegen.

Beachten Sie, dass unter AmigaOS und kompatiblen Betriebssystemen dieses Attribut mindestens MUI 4.0 erfordert.

TYP

Boolesch

ANWENDBARKEIT

IG

28.5 Listviewcolumn.Hide**BEZEICHNUNG**

Listviewcolumn.Hide – blendet Spalten ein oder aus

BESCHREIBUNG

Mit diesem Attribut können Sie einzelne Spalten der Listenansicht ein- oder ausblenden. Beachten Sie, dass die Spalten immer noch da sind, sie werden nur unsichtbar sein. Daher dürfen Sie versteckte Spalten nicht vergessen, wenn Sie Listenansicht-Einträge (ListView) mit `Listview.Insert` oder ähnlichen Methoden ändern.

Auf Nicht-AmigaOS-Systemen wird dieses Attribut nur für das DataView-Widget unterstützt. Wenn Sie eine Listenansicht erstellen und `Listviewcolumn.Hide` auf `True` gesetzt ist, wechselt RapaGUI automatisch zum DataView-Widget. Wenn Sie dieses Attribut nicht zum Zeitpunkt der Erstellung angeben, sondern es später mit `moai.Set()` festlegen möchten, müssen Sie explizit ein DataView-Widget anfordern, indem Sie das Attribut `Listview.ForceMode` setzen.

TYP

Boolesch

ANWENDBARKEIT

ISG

28.6 Listviewcolumn.Icon**BEZEICHNUNG**

Listviewcolumn.Icon – aktiviert Symbole für diese Spalte

BESCHREIBUNG

Setzen Sie dies auf `True`, wenn die Einträge in der Listenansicht in dieser Spalte Symbole verwenden sollen. In diesem Fall müssen Sie der Methode `Listview.Insert` die Hollywood-Pinsel/-Icons als Symbole übergeben.

Beachten Sie, dass RapaGUI das Bild möglicherweise skaliert, um es an die DPI-Einstellung des aktuellen Monitors anzupassen. Bitte lesen Sie das Kapitel über High-DPI-Unterstützung für weitere Informationen. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), [Seite 21](#), für Details.

Bitte lesen Sie auch das Kapitel über den Bilder-Cache von RapaGUI, um mehr über die Unterstützung von Icons in RapaGUI zu erfahren. Siehe [Abschnitt 3.20 \[Bilder-Cache\]](#), [Seite 32](#), für Details.

Unter AmigaOS und kompatiblen ist die Unterstützung von Icons nur mit MUI 4.0 oder höher verfügbar.

TYP

Boolesch

ANWENDBARKEIT

IG

28.7 Listviewcolumn.IconScale**BEZEICHNUNG**

Listviewcolumn.IconScale – konfiguriert die automatische Bildskalierung (V2.0)

BESCHREIBUNG

Wenn `Listviewcolumn.Icon` auf einen Rasterpinsel gesetzt wurde und RapaGUI auf einem High-DPI-Display ausgeführt wird, skaliert RapaGUI automatisch die Rastergrafiken des Pinsels, um ihn an die DPI-Einstellung des aktuellen Monitors anzupassen. Wenn Sie dies nicht möchten, setzen Sie diesen Tag auf `False`.

Alternativ können Sie die automatische Bildskalierung auch global deaktivieren, indem Sie beim Aufrufen vom Befehl `@REQUIRE` mit RapaGUI den Tag `ScaleGUI` auf `False` setzen. Siehe [Abschnitt 3.4 \[Initialisierung von RapaGUI\]](#), Seite 11, für Details.

Bitte lesen Sie auch das Kapitel über High-DPI-Unterstützung in RapaGUI, um mehr über die Unterstützung von High-DPI-Displays zu erfahren. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), Seite 21, für Details.

TYP

Boolesch

ANWENDBARKEIT

I

28.8 Listviewcolumn.IconType**BEZEICHNUNG**

Listviewcolumn.IconType – legt den zu verwendenden Icon-Typ fest (V2.0)

BESCHREIBUNG

Mit diesem Attribut können Sie den Typ des Hollywood-Bildobjekts festlegen, welches im Attribut `Listviewcolumn.Icon` übergeben wird. Standardmäßig erwartet `Listviewcolumn.Icon` einen Hollywood-Pinsel. Durch Setzen von `Listviewcolumn.IconType` können Sie jedoch einen anderen Hollywood-Bildtyp verwenden.

Die folgenden Bildtypen sind derzeit verfügbar:

Brush Verwendet einen Hollywood-Pinsel. Dies ist der Standardtyp. Sie können entweder Raster- oder Vektorpinsel verwenden. Vektorpinsel haben den Vorteil, dass sie ohne Qualitätsverlust auf jede beliebige Auflösung skaliert werden können. Dies ist sehr nützlich, wenn Sie Programme entwerfen, die mit High-DPI-Monitoren kompatibel sein sollen. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), Seite 21, für Details.

Icon Verwendet ein Hollywood-Icon. Dieser Bildtyp hat den Vorteil, dass er mehrere Teilbilder unterschiedlicher Größe enthalten kann. Dadurch ist

es möglich, Bilder in unterschiedlichen Auflösungen bereitzustellen, was bei der Entwicklung von Programmen, die mit High-DPI-Monitoren kompatibel sein sollen, sehr nützlich sein kann. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), [Seite 21](#), für Details.

Beachten Sie, dass Sie den Standardwert dieses Attributs global auf Hollywood-Icon ändern können, indem Sie den Tag `Application.UseIcons` setzen. Siehe [Abschnitt 9.13 \[Application.UseIcons\]](#), [Seite 70](#), für Details.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

I

28.9 Listviewcolumn.Sortable

BEZEICHNUNG

Listviewcolumn.Sortable – Spalte wird sortierbar

BESCHREIBUNG

Setzen Sie dies auf `True`, um die Spalte als sortierbar zu markieren. Sie können diese Spalte dann als aktuelle Sortierspalte festlegen, indem Sie das Attribut `Listview.SortColumn` setzen.

Setzen Sie dies auf `True`, um die Spalte als sortierbar zu kennzeichnen. Sie können dann mit `Listview.Insert` Einträge in sortierter Reihenfolge einfügen oder die Methode `Listview.Sort` aufrufen, um alle Einträge sortieren zu lassen. Auf einigen Plattformen setzt RapaGUI auch Sortierpfeile in den Spaltentitel, die der Benutzer anklicken kann, um die Spalteneinträge zu sortieren.

Beachten Sie, dass es pro Listenansicht nur eine sortierbare Spalte geben kann.

TYP

Boolesch

ANWENDBARKEIT

ISG

28.10 Listviewcolumn.SortOrder

BEZEICHNUNG

Listviewcolumn.SortOrder – setzt/ermittelt die Sortierreihenfolge der Spalten (V2.0)

BESCHREIBUNG

Legen Sie die gewünschte Sortierreihenfolge für die Spalte fest oder ermitteln Sie sie. Dies kann einer der folgenden Sonderwerte sein:

Ascending

Aufsteigende Reihenfolge. Dies ist die Voreinstellung.

Descending

Absteigende Reihenfolge.

Der Benutzer kann die Sortierreihenfolge auch ändern, indem er auf die Kopfzeile einer Listenansichtsspalte klickt. Wenn Sie eine Benachrichtigung für dieses Attribut einrichten, werden Sie über solche Ereignisse benachrichtigt.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

ISGN

28.11 Listviewcolumn.Title

BEZEICHNUNG

Listviewcolumn.Title – setzt/ermittelt die Spaltenüberschrift

BESCHREIBUNG

Setzt oder ermittelt den Titel der Spalte. Der Titel wird immer oben in der Listenansicht angezeigt und verschwindet nicht, wenn die Listenansicht gescrollt wird.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

28.12 Listviewcolumn.Width

BEZEICHNUNG

Listviewcolumn.Width – setzt/ermittelt die Spaltenbreite

BESCHREIBUNG

Stellen Sie die Spaltenbreite in geräteunabhängige Pixel ein oder ermitteln Sie sie. Der Standardwert ist -1, d.h. die Spalte sollte so groß wie der größte Eintrag sein.

TYP

Zahl

ANWENDBARKEIT

ISG

29 Listviewitem-Klasse (Listenansichts-Element)

29.1 Übersicht

Die Klasse `Listviewitem` (Listenansichts-Element) kann bei der Erstellung von Listenansichten (`Listview`) verwendet werden, um bereits bei der Objekterstellung Listenansichts-Elemente hinzuzufügen. Siehe [Abschnitt 27.1 \[Listview-Klasse\]](#), Seite 145, für Details.

`Listviewitem`-Klassen-Einträge müssen immer in eine `<column>`-Definition eingebettet werden. Der XML-Tag ist `<item>`. Siehe [Abschnitt 28.1 \[Listviewcolumn-Klasse\]](#), Seite 171, für Details.

Beachten Sie, dass Sie Instanzen dieser Klasse nicht mit `moai.CreateObject()` erstellen können. Stattdessen müssen Sie `Listview.Insert` verwenden, um zur Laufzeit neue Elemente zu erstellen. Außerdem können Sie keine Attribute von Listenansichts-Elemente mit dieser Klasse ändern. Wenn Sie den Text oder das Symbol eines Listenansichts-Elementes ändern möchten, müssen Sie stattdessen `Listview.Rename` verwenden.

29.2 Listviewitem.Icon

BEZEICHNUNG

`Listviewitem.Icon` – setzt das Bild für das Listenansichts-Elementes

BESCHREIBUNG

Setzen Sie dieses Attribut auf den Identifikator eines Hollywood-Pinsels oder -Icons, um ein Bild zu Ihrem Listeneintrag hinzuzufügen. Ob dieses Attribut einen Hollywood-Pinsel oder ein Hollywood-Icon erwartet, hängt davon ab, was Sie im Attribut `Listviewcolumn.IconType` angeben. Standardmäßig erwartet `Listviewitem.Icon` einen Hollywood-Pinsel. Sie müssen auch `Listviewcolumn.Icon` auf `True` setzen, wenn Sie Icons in einer Listenansichts-Spalte verwenden möchten.

Um das Icon eines Listeneintrags später zu ändern, können Sie die Methode `Listview.Rename` verwenden.

Beachten Sie, dass RapaGUI das Bild möglicherweise skaliert, um es an die DPI-Einstellung des aktuellen Monitors anzupassen. Bitte lesen Sie das Kapitel über High-DPI-Unterstützung für weitere Informationen. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), Seite 21, für Details.

Bitte lesen Sie auch das Kapitel über den Bilder-Cache von RapaGUI, um mehr über die Unterstützung von Icons in RapaGUI zu erfahren. Siehe [Abschnitt 3.20 \[Bilder-Cache\]](#), Seite 32, für Details.

Unter AmigaOS und kompatiblen ist die Unterstützung von Icons nur mit MUI 4.0 oder höher verfügbar.

TYP

Zahl

ANWENDBARKEIT

I

30 Menu-Klasse (Menü)

30.1 Übersicht

Die Menu-Klasse (Menü) kann verwendet werden, um ein einzelnes Abroll-Menü zu erstellen, das dann mit Elementen gefüllt werden sollte, die von der MenuItem-Klasse abgeleitet sind. Menüs werden normalerweise in ein `<menubar>`-Objekt (Menüleiste) eingebettet. Siehe [Abschnitt 31.1 \[Menubar-Klasse\], Seite 185](#), für Details. Alternativ können sie auch als Untermenüs anderer Menüobjekte eingebettet werden.

Es ist auch möglich, ein Menüobjekt als Kontextmenü zu einem der Widgets Ihres Fensters mit dem Attribut `Area.ContextMenu` hinzuzufügen. Das Kontextmenü erscheint immer dann, wenn der Benutzer mit der rechten Maustaste auf das Widget klickt, welches das Kontextmenü enthält. Im Folgenden finden Sie ein Beispiel für das Hinzufügen eines Menüobjekts als Kontextmenü zu einem Texteditor-Widget:

```
<menu title="Context menu" id="ctxtmenu">
  <item>Cut</item>
  <item>Copy</item>
  <item>Paste</item>
</menu>

<window>
...
  <texteditor contextmenu="ctxtmenu"/>
...
</window>
```

Beachten Sie, dass das Attribut `Menu.Title` nur unter AmigaOS und kompatiblen Betriebssystemen verwendet wird, wenn Sie die Menu-Klasse zum Erstellen von Kontextmenüs verwenden. Kontextmenüs unter Windows, Linux und macOS zeigen keinen Titel an.

30.2 Menu.Append

BEZEICHNUNG

Menu.Append – fügt ein losgelöstes Objekt als letztes Menü-Element hinzu

ÜBERSICHT

```
moai.DoMethod(id, "Append", obj)
```

BESCHREIBUNG

Diese Methode kann verwendet werden, um das durch `obj` angegebene losgelöste Objekt dem durch `id` angegebenen Menüobjekt hinzuzufügen. Das losgelöste Objekt wird als letztes Element des Menüs hinzugefügt. Nachdem diese Methode beendet wurde, ändert das angegebene Objekt seinen Zustand von losgelöst in angehängt. Deshalb dürfen Sie keine Befehle mehr für losgelöste Objekte für dieses Objekt verwenden.

Losgelöste MOAI-Objekte können entweder durch Aufrufen des Befehls `moai.CreateObject()` oder durch explizites Trennen ihrer MOAI-Objekte mithilfe der Methode `Menu.Remove` erstellt werden.

EINGABEN

<code>id</code>	ID des Menu-Objekts
<code>obj</code>	ID des anzuhängenden Objekts

30.3 Menu.Disabled**BEZEICHNUNG**

`Menu.Disabled` – setzt/ermittelt den Deaktivierungsstatus des Menüs

BESCHREIBUNG

Aktiviert (`False`) oder deaktiviert (`True`) das gesamte Menü. Ausserdem können Sie den Deaktivierungsstatus ermitteln.

TYP

Boolesch

ANWENDBARKEIT

ISG

30.4 Menu.Insert**BEZEICHNUNG**

`Menu.Insert` – fügt das losgelöstes Objekt nach dem angegebenen Element ein

ÜBERSICHT

```
moai.DoMethod(id, "Insert", obj, pred)
```

BESCHREIBUNG

Mit dieser Methode kann das durch `obj` angegebene gelöste Objekt in das durch `id` angegebene Menüobjekt eingefügt werden. Das gelöste Objekt wird nach dem in `pred` angegebenen Element hinzugefügt. Nachdem diese Methode beendet wurde, ändert das angegebene Objekt seinen Zustand von losgelöst in angehängt. Deshalb dürfen Sie keine Befehle mehr für losgelöste Objekte für dieses Objekt verwenden.

Losgelöste MOAI-Objekte können entweder durch Aufrufen des Befehls `moai.CreateObject()` oder durch explizites Trennen ihrer MOAI-Objekte mithilfe der Methode `Menu.Remove` erstellt werden.

EINGABEN

<code>id</code>	ID des Menu-Objekts
<code>obj</code>	ID des einzufügenden Objekts
<code>pred</code>	das Objekt wird nach diesem Element eingefügt

30.5 Menu.NoAutoKey

BEZEICHNUNG

Menu.NoAutoKey – deaktiviert die automatische Tastaturkürzel-Generierung

BESCHREIBUNG

Setzen Sie dieses Attribut, um die automatische Tastaturkürzel-Generierung zu deaktivieren. Standardmäßig wird das Zeichen nach einem Unterstrich in einem Menütitel wie eine Tastaturkürzel behandelt, mit der Sie über die Tastatur auf den Menüeintrag zugreifen können. Wenn Sie dieses Attribut setzen, werden Unterstreichungszeichen niemals als Tastaturkürzel behandelt und in Menüeinträgen normal angezeigt.

Siehe [Abschnitt 3.13 \[Tastaturkürzel\]](#), Seite 22, für Details.

TYP

Boolesch

ANWENDBARKEIT

I

30.6 Menu.Prepend

BEZEICHNUNG

Menu.Prepend – fügt ein losgelöstes Objekt als erstes Menü-Element hinzu

ÜBERSICHT

```
moai.DoMethod(id, "Prepend", obj)
```

BESCHREIBUNG

Diese Methode kann verwendet werden, um das durch `obj` angegebene losgelöste Objekt in dem durch `id` angegebene Menüobjekt hinzuzufügen. Das losgelöste Objekt wird als erstes Element des Menüs hinzugefügt. Nachdem diese Methode beendet wurde, ändert das angegebene Objekt seinen Zustand von losgelöste in angehängt. Deshalb dürfen Sie keine Befehle mehr für losgelöste Objekte für dieses Objekt verwenden.

Losgelöste MOAI-Objekte können entweder durch Aufrufen des Befehls `moai.CreateObject()` oder durch explizites Trennen ihrer MOAI-Objekte mithilfe der Methode `Menu.Remove` erstellt werden.

EINGABEN

<code>id</code>	ID des Menu-Objekts
<code>obj</code>	ID des einzufügenden Objekts

30.7 Menu.Remove

BEZEICHNUNG

Menu.Remove – entfernt/löst ein Objekt von dem Menü

ÜBERSICHT

```
moai.DoMethod(id, "Remove", obj)
```

BESCHREIBUNG

Mit dieser Methode kann das angegebene Objekt aus dem angegebenen Menü entfernt werden. Nachdem diese Methode beendet wurde, wird das angegebene Objekt seinen Zustand von angehängt auf losgelöst ändern. Das bedeutet, dass Sie es nun mit der Methode wie `Menu.Insert` an ein anderes Menü anhängen oder mit `moai.FreeObject()` aus dem Speicher löschen können.

EINGABEN

`id` ID des Menu-Objekts
`obj` ID des Objekts, welches losgelöst wird

30.8 Menu.Title**BEZEICHNUNG**

`Menu.Title` – setzt oder ermittelt den Menütitel

BESCHREIBUNG

Setzt oder ermittelt den Titel des Menüs.

Wenn der Titel einen Unterstrich enthält, richtet RapaGUI automatisch das auf diesen Unterstrich folgende Zeichen als Tastaturkürzel ein. Wenn Sie dieses Verhalten nicht wünschen, setzen Sie `Menu.NoAutoKey` auf `True`.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

30.9 Menu.Type**BEZEICHNUNG**

`Menu.Type` – setzt den Menütyp (V2.0)

BESCHREIBUNG

Mit diesem Attribut kann der Typ dieses Menüs festgelegt werden. Die folgenden Typen sind derzeit möglich:

`Help` Legt das Menü als Hilfemenü fest. Wird nur auf macOS unterstützt.

Beachten Sie, dass RapaGUI das Menü an die entsprechende Position gemäß dem Styleguide von macOS verschiebt, wenn Sie dieses Attribut auf einen der Typen setzen, die nur von macOS unterstützt werden, z.B. das Menü "Hilfe" erscheint immer ganz rechts in der Menüleiste. Auf Nicht-macOS-Systemen haben diese speziellen Typen keine Auswirkungen und werden von RapaGUI einfach ignoriert.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

I

31 Menubar-Klasse (Menüleiste)

31.1 Übersicht

Die Menubar-Klasse (Menüleiste) dient zur Verwaltung von einer Menüleiste, die über das Attribut `Window.Menubar` an Fenster angehängt werden kann. Eine Menüleiste enthält eine Anzahl von Elementen, die Objekte der Menu-Klasse sind, wobei jedes von ihnen genau ein Menü beschreibt.

In einer XML-Datei wird ein Menübaum mit den Tags `<menubar>`, `<menu>` und `<item>` definiert. Hier ist eine Beispiel-Definition einer einfachen Menüleiste:

```
<menubar id="mymenubar">
  <menu title="_File">
    <item>_New...</item>
    <item>_Open...</item>
    <item/>
    <item>_Save</item>
    <item>S_ave as...</item>
    <item/>
    <item>_Quit</item>
  </menu>
  <menu title="Edit">
    <item shortcut="Ctrl+X">_Cut</item>
    <item shortcut="Ctrl+C">C_opy</item>
    <item shortcut="Ctrl+V">_Paste</item>
  </menu>
  <menu title="?">
    <item>Se_ttings...</item>
    <item/>
    <item>A_bout...</item>
    <item>About _RapaGUI...</item>
  </menu>
</menubar>
```

Beachten Sie die Verwendung des Unterstrichzeichens im obigen XML-Code: Sie können dieses Zeichen verwenden, um das nächste Zeichen automatisch als Tastaturkürzel für den Menüeintrag zu kennzeichnen. Dies ist sehr nützlich, da viele Leute gerne die Tastatur anstelle der Maus benutzen, besonders wenn es darum geht, die gleichen Aktionen mehrmals zu wiederholen. Daher ist es immer eine gute Idee, Tastaturkürzel einzurichten. Siehe [Abschnitt 3.13 \[Tastaturkürzel\]](#), Seite 22, für Details.

Wenn Sie komplexere Tastaturkürzel benötigen, z.B. `Ctrl+V` zum Einfügen, können Sie das Attribut `MenuItem.Shortcut` verwenden, um eine solche Verknüpfung einzurichten. Beachten Sie, dass auf einigen Plattformen (z.B. Windows) beide Arten von Verknüpfungen gleichzeitig angegeben werden können: Verknüpfungen, die mit dem Unterstrichzeichen angegeben werden, und Verknüpfungen, die mit dem `MenuItem.Shortcut` definiert werden. Wenn eine bestimmte Plattform nur eine Art von Verknüpfung unterstützt, hat die in `MenuItem.Shortcut` angegebene Verknüpfung Vorrang vor der mit dem Unterstrichzeichen.

Beachten Sie auch die leeren `<item/>` im obigen XML-Code: Diese fügen eine Trennlinie in den Menübaum ein. Die Verwendung von Trennlinien macht Ihr Menü für den Endbenutzer besser lesbar. Nachdem Sie die XML-Definition oben geschrieben haben, können Sie die Menüleiste zu einem Ihrer Fenster hinzufügen, indem Sie das Attribut `Window.Menubar` wie folgt verwenden:

```
<window menubar="mymenubar">
...
</window>
```

Es ist sehr wichtig zu beachten, dass Sie Ihre Menüleisten im Bereich `<application>` definieren müssen, da Menüleisten globale Objekte sind und erst später an Fenster oder Widgets angehängt werden. Deshalb ist es nicht erlaubt, Menüleisten innerhalb eines `<window>` XML-Bereichs zu definieren.

31.2 Menubar.Append

BEZEICHNUNG

Menubar.Append – fügt ein losgelöstes Objekt als letztes Menüleisten-Element hinzu

ÜBERSICHT

```
moai.DoMethod(id, "Append", obj)
```

BESCHREIBUNG

Diese Methode kann verwendet werden, um das durch `obj` angegebene gelöste Objekt dem durch `id` angegebene Menüleisten-Objekt hinzuzufügen. Das losgelöste Objekt wird als letztes Element der Menüleiste hinzugefügt. Nachdem diese Methode beendet wurde, ändert das angegebene Objekt seinen Zustand von losgelöst in angehängt. Deshalb dürfen Sie keine Befehle mehr für losgelöste Objekte für dieses Objekt verwenden.

Losgelöste MOAI-Objekte können entweder durch Aufruf des Befehls `moai.CreateObject()` oder durch explizite Trennung von ihrem übergeordneten Objekt mit der Methode `Menubar.Remove` erzeugt werden.

EINGABEN

<code>id</code>	ID des Menüleisten-Objekts
<code>obj</code>	ID des hinzufügenden Objekts

31.3 Menubar.Insert

BEZEICHNUNG

Menubar.Insert – fügt ein losgelöstes Objekt nach dem angegebenen Menüleisten-Element hinzu

ÜBERSICHT

```
moai.DoMethod(id, "Insert", obj, pred)
```

BESCHREIBUNG

Diese Methode kann verwendet werden, um das durch `obj` angegebene losgelöste Objekt in das durch `id` angegebene Menüobjekt einzufügen. Das losgelöste Objekt wird nach

dem in `pred` angegebenen Element hinzugefügt. Nachdem diese Methode beendet wurde, ändert das angegebene Objekt seinen Zustand von losgelöst in angehängt. Deshalb dürfen Sie keine Befehle mehr für losgelöste Objekte für dieses Objekt verwenden.

Losgelöste MOAI-Objekte können entweder durch Aufruf des Befehls `moai.CreateObject()` oder durch explizite Trennung von ihrem übergeordneten Objekt mit der Methode `Menubar.Remove` erzeugt werden.

EINGABEN

<code>id</code>	ID des Menüleisten-Objekts
<code>obj</code>	ID des hinzufügenden Objekts
<code>pred</code>	Das Objekt wird nach diesem Objekt eingefügt

31.4 Menubar.Prend

BEZEICHNUNG

`Menubar.Prend` – fügt ein losgelöstes Objekt als erstes Menüleisten-Element hinzu

ÜBERSICHT

```
moai.DoMethod(id, "Prepend", obj)
```

BESCHREIBUNG

Diese Methode kann verwendet werden, um das durch `obj` angegebene gelöste Objekt dem durch `id` angegebene Menüleisten-Objekt hinzuzufügen. Das losgelöste Objekt wird als erstes Element der Menüleiste hinzugefügt. Nachdem diese Methode beendet wurde, ändert das angegebene Objekt seinen Zustand von losgelöst in angehängt. Deshalb dürfen Sie keine Befehle mehr für losgelöste Objekte für dieses Objekt verwenden.

Losgelöste MOAI-Objekte können entweder durch Aufruf des Befehls `moai.CreateObject()` oder durch explizite Trennung von ihrem übergeordneten Objekt mit der Methode `Menubar.Remove` erzeugt werden.

EINGABEN

<code>id</code>	ID des Menüleisten-Objekts
<code>obj</code>	ID des hinzufügenden Objekts

31.5 Menubar.Remove

BEZEICHNUNG

`Menubar.Remove` – löst ein Objekt von der Menüleiste

ÜBERSICHT

```
moai.DoMethod(id, "Remove", obj)
```

BESCHREIBUNG

Mit dieser Methode kann das angegebene Objekt aus der angegebenen Menüleiste entfernt werden. Nachdem diese Methode beendet ist, wird das angegebene Objekt seinen Zustand von angehängt auf losgelöst ändern. Das bedeutet, dass Sie es nun mit Methoden

wie `MenuBar.Insert` an eine andere Menüleiste anhängen oder mit `moai.FreeObject()` aus dem Speicher löschen können.

EINGABEN

<code>id</code>	ID des Menüleisten-Objekts
<code>obj</code>	ID des zu loslösenden Objekts

32 MenuItem-Klasse (Menüelement)

32.1 Übersicht

Mit Hilfe der MenuItem-Klasse (Menüelement) kann ein einzelner Menüpunkt erstellt werden. Da solche Menüpunkte immer in einem `<menu>`-Tag eingebettet sein müssen, das wiederum normalerweise in einem `<menubar>`-Tag eingebettet ist. Siehe [Abschnitt 31.1 \[Menubar-Klasse \(Menüleiste\)\]](#), Seite 185, für Details.

Bitte beachten Sie, dass das XML-Tag für die MenuItem-Klasse nur `<item>` und nicht `<menuItem>` ist, wenn Sie Menüpunkte erstellen, die in einem `<menu>`-Baum eingebettet sind. Wenn Sie isolierte Menüeinträge erstellen, die nicht in einem `<menu>`-Baum eingebettet sind, müssen Sie `<menuItem>` verwenden, da RapaGUI sonst nicht wissen könnte, auf welche Klasse Sie sich beziehen, da `<item>` von vielen verschiedenen Klassen verwendet wird.

Wenn Sie keinen Text für den Menüpunkt angeben, wird ein Trennelement erzeugt. Dies ist nützlich, um bestimmte Menüpunkte zusammenzufassen, was zu einer besseren Lesbarkeit des Menüs führt.

Siehe [Abschnitt 31.1 \[Menubar-Klasse\]](#), Seite 185, für ein Beispiel.

32.2 MenuItem.Disabled

BEZEICHNUNG

`MenuItem.Disabled` – setzt/ermittelt den Deaktivierungsstatus des Menüpunktes

BESCHREIBUNG

Aktiviert (`False`) oder deaktiviert (`True`) den Menüpunkt. Ausserdem können Sie den Deaktivierungsstatus ermitteln.

TYP

Boolesch

ANWENDBARKEIT

ISG

32.3 MenuItem.Help

BEZEICHNUNG

`MenuItem.Help` – setzt/ermittelt den Hilfetext des Menüeintrag

PLATTFORMEN

Windows, Linux, macOS

BESCHREIBUNG

Setzt oder ermittelt den Hilfetext des Menüpunktes. Dieser Text wird automatisch in der Statusleiste des Fensters angezeigt, in dem sich das Menü befindet, wenn Sie mit der Maus über den Menüpunkt fahren. Dies ist für den Anwender sehr komfortabel, da es die Funktion der einzelnen Menüpunkte etwas ausführlicher erklärt. Siehe [Abschnitt 46.1 \[Statusbar-Klasse \(Statusleiste\)\]](#), Seite 239, für Details.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

32.4 MenuItem.NoAutoKey

BEZEICHNUNG

MenuItem.NoAutoKey – deaktiviert die automatische Tastaturkürzel-Generierung

BESCHREIBUNG

Setzen Sie dieses Attribut, um die automatische Tastaturkürzel-Generierung zu deaktivieren. Standardmäßig wird das Zeichen nach einem Unterstrich in einem Menüpunkt wie ein Tastaturkürzel behandelt, mit der Sie über die Tastatur auf den Menüpunkt zugreifen können. Wenn Sie dieses Attribut setzen, werden Unterstreichungszeichen niemals als Tastaturkürzel behandelt und in den Menüpunkten normal angezeigt.

Siehe [Abschnitt 3.13 \[Tastaturkürzel\]](#), Seite 22, für Details.

TYP

Boolesch

ANWENDBARKEIT

I

32.5 MenuItem.Selected

BEZEICHNUNG

MenuItem.Selected – benachrichtigt, ob ein Element ausgewählt ist

BESCHREIBUNG

Dieses Attribut wird ausgelöst, wenn der Benutzer einen Menüeintrag auswählt. RapaGUI überwacht dieses Attribut automatisch für alle Menü-Elemente, sodass Sie keine explizite Benachrichtigung über das Attribut `MOAI.Notify` anfordern müssen.

Bei Menüpunkten, die umgeschaltet werden können oder Teil einer Radio-Gruppe sind, gibt die Abfrage dieses Attributs den aktuellen Umschalt- oder Radio-Status des Menüpunktes zurück.

TYP

Boolesch

ANWENDBARKEIT

ISGN

32.6 MenuItem.Shortcut

BEZEICHNUNG

MenuItem.Shortcut – setzt ein benutzerdefiniertes Tastaturkürzel für den Menüpunkt

BESCHREIBUNG

Legen Sie ein benutzerdefiniertes Tastaturkürzel für einen Menüpunkt fest. Normalerweise werden Menüpunkt-Tastaturkürzel definiert, indem man einfach den Unterstrich im Titel des Menüpunkts verwendet, um eine Tastenkombination zu definieren. Siehe [Abschnitt 3.13 \[Tastaturkürzel\]](#), Seite 22, für Details. Es gibt jedoch bestimmte Fälle, in denen dies nicht ausreicht. Zunächst einmal sind Sie durch die Verwendung des Unterstrichzeichens zur Definition einer Tastenkombination auf ein Zeichen beschränkt, die Teil des Textes des Menüpunkts sind. In vielen Fällen werden Sie jedoch ein Tastaturkürzel definieren wollen, die nicht Teil des Textes ist. Z.B. das typische Tastaturkürzel für "Einfügen" ist `Ctrl+V`, obwohl es kein "V" im Wort "Einfügen" gibt. Hier können Sie mit `MenuItem.Shortcut` beliebige Tastenkombination für Ihre Menüpunkte einrichten.

Zusätzlich ist dieses Attribut auch nützlich, um komplexere Tastenkombination zu definieren, z.B. `Alt+F5` oder `Ctrl+Shift+X` etc. Die Definition solcher komplexen Tastenkombination ist auch nicht über die Standardunterstreichung der Menütexpte möglich, daher müssen Sie dafür `MenuItem.Shortcut` verwenden.

Die hier angegebene Zeichenkette muss eine Kombination von Umschalttasten und anderen Tasten sein, die durch die Zeichen + oder - getrennt sind. Die folgenden Umschalttasten und Tasten werden derzeit erkannt:

- | | |
|----------------|---|
| Ctrl | Dies gibt die Control-Taste unter Windows und Linux und die Befehlstaste auf AmigaOS und macOS an. Der Grund, warum dies der Befehlstaste auf AmigaOS und macOS zugeordnet wurde, ist Ihnen das Schreiben von portablem Code zu erleichtern. Unter Windows und Linux ist die Control-Taste die Standard-Zusatztaste, während AmigaOS und macOS die Befehlstaste als Standard-Zusatztaste verwenden. Wenn Sie <code>Ctrl</code> angeben, erhalten Sie immer die Standard-Zusatztasten des Systems. Wenn Sie auf AmigaOS und macOS die echte Control-Taste verwenden müssen, benutzen Sie <code>RawCtrl</code> (siehe unten). Wenn RapaGUI <code>Ctrl</code> nicht als spezielles Token behandelt, müssen Sie immer separaten Code für Windows und Linux im Gegensatz zu AmigaOS und macOS schreiben, d.h. Sie müssen bei Windows und Linux <code>Ctrl+V</code> zum Einfügen angeben und <code>Cmd+V</code> zum Einfügen auf AmigaOS und macOS. Da dies zu unnötigem Overhead (Zusatzarbeit für den Computer) führt, wird <code>Ctrl</code> als spezielle Taste behandelt, die dem standardmäßigen Menükürzel-Taste des Systems zugeordnet ist. |
| Alt | Verwenden Sie die Alt-Taste als Umschalttaste. |
| Shift | Verwenden Sie die Shift-Taste als Umschalttaste. |
| RawCtrl | Unter AmigaOS und macOS können Sie mit dieser Umschalttaste die echte Control-Taste überwachen. Wenn Sie stattdessen <code>Ctrl</code> verwenden, überwacht RapaGUI die Befehlstaste auf AmigaOS und macOS (siehe oben). Unter Windows und Linux ist <code>RawCtrl</code> identisch mit <code>Ctrl</code> . |
| Up | Pfeil nach oben |

Down	Pfeil nach unten
Right	Pfeil nach rechts
Left	Pfeil nach links
Help	Hilfe-Taste
Del	Lösch-Taste
Backspace	Rückschritt-Taste
Tab	Tabulator-Taste
Return	Return-Taste
Enter	Enter-Taste
Esc	Escape-Taste
Space	Leertaste
F1 - F16	Funktionstasten
Insert	Einfüge-Taste
Home	Anfangs-Taste
End	Ende-Taste
PageUp	Seite rauf
PageDown	Seite runter
Print	Drucken-Taste
Pause	Pause-Taste

Darüber hinaus können Sie auch englische Buchstaben von A bis Z sowie die Zahlen 0 bis 9 als Tastaturkürzel benutzen.

Beachten Sie bitte, dass die Angabe von Tastaturkürzel mit dem Unterstrich und `MenuItem.Shortcut` sich nicht gegenseitig ausschließen. Sie können tatsächlich beide angeben und auf einigen Systemen (z. B. Windows) wird RapaGUI sogar beide unterstützen. Unter Windows können Sie beispielsweise über die Tastatur auf Menü-Elemente zugreifen, indem Sie zunächst auf `Alt` und dann auf den Unterstreichungskürzel drücken, oder Sie können auch auf Menü-Elemente zugreifen, indem Sie ein vordefiniertes Tastaturkürzel drücken, z.B. `Ctrl+V` zum Einfügen. Auf Systemen, die beide Arten von Tastaturkürzel nicht unterstützen, hat die in `MenuItem.Shortcut` angegebene Tastaturkürzel Vorrang.

TYP

Zeichenkette

ANWENDBARKEIT

I

32.7 MenuItem.Title

BEZEICHNUNG

MenuItem.Title – setzt/ermittelt den Menüpunkttext

BESCHREIBUNG

Setzt oder ermittelt den Menüpunkttext.

Wenn der Text einen Unterstrich enthält, richtet RapaGUI dem Unterstrich folgenden Zeichen automatisch als Tastaturkürzel ein. Wenn Sie dieses Verhalten nicht möchten, setzen Sie `MenuItem.NoAutoKey` auf `True`.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

32.8 MenuItem.Type

BEZEICHNUNG

MenuItem.Type – setzt/ermittelt den Menüpunkttyp

BESCHREIBUNG

Mit diesem Attribut kann der Typ für diesen Menüpunkt eingestellt werden. Folgende Typen sind derzeit möglich:

<code>Normal</code>	Normaler Menüpunkt.
<code>Toggle</code>	Umschalt-Menüpunkt mit einem Häkchen.
<code>Radio</code>	Radio-Menüpunkt, das mit seinen benachbarten Radio-Menüpunkten eine Gruppe bildet. Es kann immer nur ein Menüpunkt der Gruppe ausgewählt werden.
<code>About</code>	Legt den Menüpunkt als Über fest. Wird nur auf macOS unterstützt. (V2.0)
<code>Help</code>	Legt den Menüpunkt als Hilfe fest. Wird nur auf macOS unterstützt. (V2.0)
<code>Preferences</code>	Bestimmt den Eintrag als Menüpunkt für die Einstellungen. Wird nur auf macOS unterstützt. (V2.0)
<code>Quit</code>	Legt den Menüpunkt als Beenden fest. Wird nur auf macOS unterstützt. (V2.0)

Beachten Sie, dass RapaGUI den Menüpunkt gemäß dem Styleguide von macOS an die entsprechende Position verschiebt, wenn Sie dieses Attribut auf einen der Typen setzen, die nur von macOS unterstützt werden, z.B. die Menüs "Über", "Einstellungen" und "Beenden" sind Teil des Programmmenüs von macOS. Auf Nicht-macOS-Systemen haben diese speziellen Typen keine Auswirkungen und RapaGUI ignoriert sie einfach.

TYP

Zeichenkette (mögliche Werte siehe oben)

ANWENDBARKEIT
IG

33 MOAI-Klasse

33.1 Übersicht

MOAI-Klasse ist die Oberklasse aller anderen MOAI-Klassen. Es verwaltet einerseits den internen Benachrichtigungsmechanismus, andererseits stellt es auch einige allgemeine Attribute bereit, die mit allen Objekttypen verwendet werden können, z.B. können Sie Benutzerdaten innerhalb Ihrer Objekte mit `MOAI.UserData` speichern.

33.2 MOAI.Class

BEZEICHNUNG

MOAI.Class – ermittelt den Namen der Klasse des Objekts

BESCHREIBUNG

Ermittelt den RapaGUI-Klassenamen eines Objekts.

TYP

Zeichenkette

ANWENDBARKEIT

G

33.3 MOAI.I18N

BEZEICHNUNG

MOAI.I18N – setzt den Katalog-Zeichenketten-Index (V2.0)

BESCHREIBUNG

Dieses Attribut kann verwendet werden, um den Katalog-Zeichenketten-Index für das Objekt festzulegen. Dies wird nur von Widgets unterstützt, die es Ihnen ermöglichen, eine Textzeichenkette als Teil ihrer XML-Deklaration zu übergeben, z.B. Widgets vom Typ Button-Klasse. In diesem Tag müssen Sie den Index der Katalog-Zeichenketten für das Widget entweder als numerischen Wert oder als Hollywood-Konstante übergeben.

Siehe [Abschnitt 3.17 \[Internationalisierung\]](#), [Seite 26](#), für Details.

TYP

Zahl oder Hollywood-Konstante

ANWENDBARKEIT

I

33.4 MOAI.ID

BEZEICHNUNG

MOAI.ID – setzt die Objekt-ID

BESCHREIBUNG

Mit diesem Attribut kann die ID für ein MOAI-Objekt gesetzt werden. Sie müssen Ihren Objekten eindeutige IDs geben, damit Sie mit den Befehlen `moai.Set()`, `moai.Get()` und `moai.DoMethod()` darauf zugreifen können.

TYP

Zeichenkette

ANWENDBARKEIT

I

33.5 MOAI.NoNotify

BEZEICHNUNG

MOAI.NoNotify – deaktiviert Benachrichtigungen

BESCHREIBUNG

Beim Einrichten von Benachrichtigungen für Klassenattribute werden Ereignisse auch dann ausgelöst, wenn der Wert eines Attributs manuell durch Aufruf von `moai.Set()` geändert wird. Durch das Setzen von `MOAI.NoNotify` im selben Aufruf wird verhindert, dass die Benachrichtigung ausgelöst wird.

Beachten Sie, dass `MOAI.NoNotify` ein "einmaliges" Attribut ist. Es ist nur während des aktuellen Aufrufs von `moai.Set()` wirksam!

TYP

Boolesch

ANWENDBARKEIT

S

BEISPIEL

```
moai.Set("lv", "active", 5, "nonotify", true)
```

Der obige Code aktiviert den Listeneintrag Nummer 6, löst aber keine Benachrichtigung aus.

33.6 MOAI.Notify

BEZEICHNUNG

MOAI.Notify – richtet Benachrichtigungen ein

BESCHREIBUNG

Mit diesem Attribut legen Sie fest, welche Attribute Sie überwachen möchten. Immer wenn sich der Wert des Attributs ändert, wird RapaGUI Ihre Callback-Funktion ausführen. Mehrere Attribute müssen durch Semikolons/Strichpunkte (;) getrennt werden. Um beispielsweise die Attribute `Listview.Active` und `Listview.DoubleClick` zu überwachen, müssten Sie hier die Zeichenkette "active; doubleclick" übergeben.

Um Benachrichtigungen außerhalb von XML-Definition einzurichten oder zu entfernen, verwenden Sie den Befehl `moai.Notify()`. Siehe [Abschnitt 6.11 \[moai.Notify\]](#), Seite 55, für Details.

TYP

Zeichenkette

ANWENDBARKEIT

I

33.7 MOAI.NotifyData

BEZEICHNUNG

MOAI.NotifyData – setzt/ermittelt ereignisspezifische Benutzerdaten

BESCHREIBUNG

Mit diesem Attribut können Sie benachrichtigungsspezifische Benutzerdaten in einem Objekt definieren. Sie müssen hier eine Zeichenfolge übergeben, die eine oder mehrere Benachrichtigungen und Benutzerdaten für jede Benachrichtigung in der Zeichenfolge enthält. Wenn eine Benachrichtigung ausgelöst wird, die in der Zeichenfolge angegeben ist, empfängt die Callback-Funktion die Benutzerdaten, die in `MOAI.NotifyData` im Feld `NotifyData` der Ereignismeldung angegeben sind.

Die Zeichenkette, die Sie an dieses Attribut übergeben müssen, muss wie folgt formatiert sein: Name des Benachrichtigungsattributs gefolgt von einem Doppelpunkt (:), gefolgt von einem Benutzerdaten-Zeichenkette, gefolgt von einem Semikolon (;). Die Sequenz kann dann beliebig oft wiederholt werden.

Zum Beispiel: "Active: foo; DoubleClick: bar;". Wenn die "Active"-Benachrichtigung ausgelöst wird, wird "foo" an die Callback-Funktion gesendet. Wenn das Attribut "DoubleClick" ausgelöst wird, wird "bar" gesendet.

Siehe [Abschnitt 3.7 \[Benachrichtigungen der Attribute\]](#), Seite 14, für Details.

TYP

Alle

ANWENDBARKEIT

ISG

33.8 MOAI.UserData

BEZEICHNUNG

MOAI.UserData – setzt/ermittelt die Benutzerdaten des Objekts

BESCHREIBUNG

Speichern Sie mit diesem Attribut beliebige Werte in einem Objekt. Sie können diesen Wert später direkt aus dem Objekt holen. Dies ist ein guter Mechanismus, um die Verwendung von globalen Variablen zu vermeiden.

Die hier angegebenen Benutzerdaten werden auch an Ihre Callback-Funktion übergeben, die Sie mit `InstallEventHandler()` in Hollywood installiert haben. Die Callback-Funktion erhält die in `MOAI.UserData` im Feld `MOAIUserData` der Ereignismeldung angegebenen Benutzerdaten. Siehe [Abschnitt 3.7 \[Benachrichtigungen der Attribute\]](#), Seite 14, für Details.

TYP

Alle

ANWENDBARKEIT

ISG

34 Pageview-Klasse (Seitenansicht)

34.1 Übersicht

Die Pageview-Klasse (Seitenansicht) leitet sich von der Area-Klasse ab und ist ein Container für mehrere Seiten von Widgets, die jeweils als eine Seite angezeigt werden. Wenn Sie diese Klasse verwenden, müssen Sie nur eine Anzahl von Gruppen angeben, die die Elemente der Seitenansicht bilden. Diese Elemente können dann auf verschiedene Arten visualisiert werden, abhängig vom Attribut `Pageview.Mode`, das das Widget zum Durchblättern der Seiten bestimmt. Die folgenden Widgets sind derzeit für diesen Zweck verfügbar:

- Register-Widget (voreingestellt)
- Listen-Widget
- Auswahl-Widget (Choice)
- kein, d.h. Seiten können nur programmäßig geändert werden (nützlich für Setup-Assistenten etc.)

Wenn Sie eine Seitenansicht einrichten, müssen Sie das Attribut `Group.Title` verwenden, um Titeltexthe für Ihre Seitenansicht-Elemente zu definieren. Sie können auch das Attribut `Group.Icon` verwenden, um Symbole für Ihre Seitenansicht-Elemente hinzuzufügen.

Hier ist ein Beispiel für eine dreiseitige Seitenansicht-Gruppe:

```
<pageview>
  <vgroup title="Page 1">
    <listview>
      <column>
        <item>Entry</item>
      </column>
    </listview>
  </vgroup>
  <vgroup title="Page 2">
    <texteditor/>
  </vgroup>
  <vgroup title="Page 3">
    <button id="btn">Click me</button>
  </vgroup>
</pageview>
```

34.2 Pageview.Active

BEZEICHNUNG

`Pageview.Active` – setzt/ermittelt die aktive Seite

BESCHREIBUNG

Setzt oder ermittelt die aktive Seite. Das Seitenansicht-Widget zeigt nur diese Seite an, alle anderen Seiten werden ausgeblendet. Die Seitenindizes reichen von 0 für die erste Seite bis zu `Pageview.Pages-1` für die letzte.

Sie können hier auch einen der folgenden speziellen Werte übergeben:

First	Erste Seite.
Last	Letzte Seite.
Prev	Vorherige Seite.
Next	Nächste Seite.

Sie können auch eine Benachrichtigung für dieses Attribut einrichten, um zu erfahren, wenn der Benutzer die Seiten wechselt.

TYP

Zahl oder Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

ISGN

34.3 Pageview.Append

BEZEICHNUNG

Pageview.Append – fügt der Seitenansicht eine neue Seite hinzu

ÜBERSICHT

```
moai.DoMethod(id, "Append", obj[, active])
```

BESCHREIBUNG

Diese Methode kann verwendet werden, um eine neue Seite zu einer Seitenansicht hinzuzufügen. Die durch `obj` angegebene neue Seite muss ein losgelöstes Gruppenobjekt sein. Dieses losgelöste Gruppenobjekt wird dann als letzte Seite der Seitenansicht hinzugefügt. Nachdem diese Methode beendet wurde, ändert das angegebene Gruppenobjekt seinen Zustand von losgelöst in angehängt. Deshalb dürfen Sie keine Befehle mehr für losgelöste Objekte für dieses Objekt verwenden.

Losgelöste MOAI-Objekte können entweder durch Aufruf des Befehls `moai.CreateObject()` oder durch explizite Trennung von ihrem übergeordneten Objekt mit der Methode `Pageview.Remove` erzeugt werden.

Unter AmigaOS und kompatiblen Betriebssystemen erfordert diese Fähigkeit mindestens MUI 4.0.

EINGABEN

id	ID des Pageview-Objekts
obj	ID des Gruppenobjekts, das als neue Seite hinzugefügt werden soll.
active	optional: setzen Sie dies auf <code>True</code> , um die Seite nach dem Hinzufügen zu aktivieren.

34.4 Pageview.GetPageID

BEZEICHNUNG

Pageview.GetPageID – gibt die ID einer Seite innerhalb der Seitenansicht zurück

ÜBERSICHT

```
id$ = moai.DoMethod(id, "GetPageID", idx)
```

BESCHREIBUNG

Diese Methode gibt die ID der Seite am angegebenen Index `idx` innerhalb der Seitenansicht in `id$` zurück. `idx` kann eine absolute Zahl sein, die von 0 bis zur Anzahl der Seiten in der Seitenansicht minus 1 reicht, oder es kann einer der folgenden Sonderwerte sein:

First Erste Seite.
Last Letzte Seite.
Active Aktive Seite.

Unter AmigaOS und kompatiblen Betriebssystemen erfordert diese Fähigkeit mindestens MUI 4.0.

EINGABEN

`id` ID des Pageview-Objekts
`idx` absoluter Index der gewünschten Seite oder spezielle Zeichenkettenkonstante (siehe oben)

RÜCKGABEWERTE

`id$` ID des Seitenobjektes beim angegebenen Seitenansicht-Index

34.5 Pageview.Insert

BEZEICHNUNG

Pageview.Insert – fügt eine neue Seite in die Seitenansicht ein

ÜBERSICHT

```
moai.DoMethod(id, "Insert", obj, pos[, active])
```

BESCHREIBUNG

Diese Methode kann verwendet werden, um eine neue Seite an der in `pos` angegebenen Stelle in einer Seitenansicht einzufügen. Die Einfügepositionen werden ab 0 gezählt, was die Position der ersten Seite markiert. Die durch `obj` spezifizierte neue Seite muss ein losgelöstes Gruppenobjekt sein. Nachdem diese Methode beendet wurde, ändert das angegebene Objekt seinen Zustand von losgelöst in angehängt. Deshalb dürfen Sie keine Befehle mehr für losgelöste Objekte für dieses Objekt verwenden.

Losgelöste MOAI-Objekte können entweder durch Aufruf des Befehls `moai.CreateObject()` oder durch explizite Trennung von ihrem übergeordneten Objekt mit der Methode `Pageview.Remove` erzeugt werden.

Unter AmigaOS und kompatiblen Betriebssystemen erfordert diese Fähigkeit mindestens MUI 4.0.

EINGABEN

<code>id</code>	ID des Pageview-Objekts
<code>obj</code>	ID des Gruppenobjekts, das als neue Seite eingefügt werden soll
<code>pos</code>	gewünschte Einfügeposition für die neue Seite (ab 0)
<code>active</code>	optional: Setzen Sie dies auf <code>True</code> , um die Seite nach dem Hinzufügen zu aktivieren

34.6 Pageview.Mode**BEZEICHNUNG**

`Pageview.Mode` – setzt das Browser-Widget zum Durchblättern der Seiten

BESCHREIBUNG

Wählen Sie das Browser-Widget ein, das dieses Pageview-Objekt verwenden soll. Dies kann einer der folgenden Werte sein:

<code>Tabs</code>	Verwendet Registerkarten, um durch die Seiten zu blättern. Dies ist der Standardmodus.
<code>List</code>	Verwendet ein Listenansichts-Widget (Listview), um durch die Seiten zu blättern.
<code>Choice</code>	Verwendet ein Auswahl-Widget (Choice), um durch die Seiten zu blättern.
<code>None</code>	Verwendet kein Browser-Widget. Dies bedeutet, dass der Benutzer nicht durch die Seiten blättern kann. Seiten können nur durch Setzen des Attributs <code>Pageview.Active</code> geändert werden. Dies ist nützlich für Setup- oder Installationsassistenten, die oft eine benutzerdefinierte Seitenansicht verwenden, die nur dann die Seite wechselt, wenn der Benutzer auf die Schaltfläche "Nächste Seite" oder "Vorherige Seite" klickt.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

I

34.7 Pageview.Multiline**BEZEICHNUNG**

`Pageview.Multiline` – aktiviert mehrzeilige Registerkarten

PLATTFORMEN

Nur Windows

BESCHREIBUNG

Setzen Sie dies auf `True`, damit das Browser-Widget seine Tabs auf mehreren Zeilen anzeigt, anstatt Scroll-Schaltflächen hinzuzufügen, wenn die Anzahl der Tabs den verfügbaren Platz überschreitet. Offensichtlich macht dieses Attribut nur Sinn, wenn der Modus `Tabs` mit `Pageview.Mode` verwendet wird.

TYP

Boolesch

ANWENDBARKEIT

I

34.8 Pageview.Pages

BEZEICHNUNG

Pageview.Pages – ermittelt die Anzahl der Seiten in der Seitenansicht

BESCHREIBUNG

Mit diesem Attribut können Sie die aktuelle Anzahl der Seiten im Pageview-Objekt ermitteln.

Unter AmigaOS und kompatiblen Betriebssystemen erfordert diese Fähigkeit mindestens MUI 4.0.

TYP

Zahl

ANWENDBARKEIT

G

34.9 Pageview.PlainBG

BEZEICHNUNG

Pageview.PlainBG – erzwingt einen normalen Gruppenhintergrund

PLATTFORMEN

Nur AmigaOS und kompatible Betriebssysteme

BESCHREIBUNG

Standardmäßig werden die einzelnen Seitenansichts-Elementgruppen mit einer speziellen Hintergrundfarbe gezeichnet, um anzuzeigen, dass es Seitenansichts-Elemente sind. Setzen Sie dieses Attribut auf `True`, wenn Sie diesen speziellen Hintergrund nicht wünschen.

TYP

Boolesch

ANWENDBARKEIT

I

34.10 Pageview.Position

BEZEICHNUNG

Pageview.Position – legt die Position des Browser-Widgets fest

BESCHREIBUNG

Mit diesem Attribut können Sie konfigurieren, wo das Browser-Widget der Seitenansicht (Register-, Liste- oder Auswahl-Widget) erscheinen soll.

Die folgenden Werte werden durch dieses Attribut erkannt:

Left Setzt das Browser-Widget links neben die Seiten.

Right Setzt das Browser-Widget rechts neben die Seiten.

Bottom Setzt das Browser-Widget an den unteren Rand der Seiten.

Top Setzt das Browser-Widget an den oberen Rand der Seiten.

Unter AmigaOS und kompatiblen Betriebssystemen erfordert diese Fähigkeit mindestens MUI 4.0.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

I

34.11 Pageview.Prepend

BEZEICHNUNG

Pageview.Prepend – stellt eine neue Seite der Seitenansicht voran

ÜBERSICHT

```
moai.DoMethod(id, "Prepend", obj[, active])
```

BESCHREIBUNG

Diese Methode kann verwendet werden, um eine neue Seite einer Seitenansicht voranzustellen. Die durch **obj** angegebene neue Seite muss ein losgelöstes Gruppenobjekt sein. Dieses losgelöste Gruppenobjekt wird dann als erste Seite der Seitenansicht hinzugefügt. Nachdem diese Methode beendet wurde, ändert das angegebene Objekt seinen Zustand von losgelöst in angehängt. Deshalb dürfen Sie keine Befehle mehr für losgelöste Objekte auf dieses Objekt verwenden.

Losgelöste MOAI-Objekte können entweder durch Aufruf des Befehls `moai.CreateObject()` oder durch explizite Trennung von ihrem übergeordneten Objekt mit der Methode `Pageview.Remove` erzeugt werden.

Unter AmigaOS und kompatiblen Betriebssystemen erfordert diese Fähigkeit mindestens MUI 4.0.

EINGABEN

id ID des Pageview-Objekts

obj ID des Gruppenobjekts, das als erste Seite hinzugefügt werden soll

active optional: setzen Sie dies auf `True`, um die Seite nach dem Hinzufügen zu aktivieren

34.12 Pageview.Remove

BEZEICHNUNG

Pageview.Remove – entfernt eine Seite aus der Seitenansicht

ÜBERSICHT

```
moai.DoMethod(id, "Remove", obj)
```

BESCHREIBUNG

Diese Methode entfernt die von `obj` angegebene Seite aus der Seitenansicht. Das Gruppenobjekt, das die zu entfernende Seite darstellt, ändert dann seinen Zustand von angehängt auf losgelöst. Das bedeutet, dass Sie es nun mit Methoden wie `Group.Insert` und `Pageview.Append` an eine andere Seitenansicht oder eine andere Gruppe anhängen oder mit `moai.FreeObject()` aus dem Speicher löschen können.

Unter AmigaOS und kompatiblen Betriebssystemen erfordert diese Fähigkeit mindestens MUI 4.0.

EINGABEN

<code>id</code>	ID des Pageview-Objekts
<code>obj</code>	ID der zu entfernenden Seite

35 Popcolor-Klasse (Farbdialog)

35.1 Übersicht

Die Popcolor-Klasse (Farbdialog) leitet sich von der Area-Klasse ab und erstellt ein Widget, mit dem der Benutzer eine Farbe auswählen kann. Farbdialog-Widgets enthalten in der Regel eine Schaltfläche, die beim Anklicken einen Dialog öffnet, der den Benutzer zur Auswahl einer Farbe auffordert. Dies alles wird automatisch vom Widget verwaltet und Sie müssen nichts tun, außer das Widget irgendwo in Ihrem Fensterlayout zu platzieren.

Sie können den Titel des Farbdialog mit dem Attribut `Popcolor.Title` auf dem Popcolor-Objekt festlegen. Um die aktuelle Farbe eines Popcolor-Objekts zu erhalten oder zu setzen, können Sie das Attribut `Popcolor.RGB` verwenden.

35.2 Popcolor.RGB

BEZEICHNUNG

`Popcolor.RGB` – setzt die Farbe oder gibt sie zurück

BESCHREIBUNG

Setzen oder holen Sie die Farbdialog-Farbe. Wenn Sie eine Benachrichtigung zu diesem Attribut einrichten, werden Sie benachrichtigt, wenn sich die Farbe ändert. Aus dem Hollywood-Skript wird die Farbe als einfacher Zahlenwert angegeben, der für jede Komponente 8 Bit enthält. Wenn Sie die Farbe in der XML-Datei angeben, muss sie als 6-stellige Zeichenkette mit dem #-Zeichen als Präfix übergeben werden (genau wie in HTML).

TYP

Zahl

ANWENDBARKEIT

ISGN

35.3 Popcolor.Title

BEZEICHNUNG

`Popcolor.Title` – setzt den Farbdialog-Fenstertitel

BESCHREIBUNG

Setzt den Titel für das Farbdialog-Fenster.

TYP

Zeichenkette

ANWENDBARKEIT

I

36 Popfile-Klasse (Dateidialog)

36.1 Übersicht

Popfile-Klasse (Dateidialog) leitet sich von der Area-Klasse ab und erstellt ein Widget, mit dem der Benutzer eine Datei auswählen kann. Dateidialog-Widgets enthalten normalerweise eine Schaltfläche, die beim Anklicken einen System-Dialog öffnet, in dem der Benutzer aufgefordert wird, eine Datei auszuwählen. Dies alles wird automatisch vom Widget verwaltet und Sie müssen nichts anderes tun, als das Widget irgendwo in Ihrem Fensterlayout zu platzieren.

Sie können den Titel des Popup-Dialogs mit dem Attribut `Popfile.Title` auf dem Popfile-Objekt setzen und die Auswahl des Benutzers mit dem Attribut `Popfile.File` ermitteln.

36.2 Popfile.File

BEZEICHNUNG

`Popfile.File` – setzt oder ermittelt den aktuellen Dateidialog-Pfad

BESCHREIBUNG

Ermitteln oder setzen Sie den aktuellen Pfad von diesem Popfile-Objekt.

Sie können für dieses Attribut auch eine Benachrichtigung einrichten, um benachrichtigt zu werden, wenn der Benutzer eine neue Datei für dieses Popfile-Objekt auswählt.

TYP

Zeichenkette

ANWENDBARKEIT

ISGN

36.3 Popfile.Pattern

BEZEICHNUNG

`Popfile.Pattern` – setzt das Filtermuster für den Datei-System-Dialog

BESCHREIBUNG

Hiermit setzen Sie das Filtermuster für den Datei-System-Dialog. Nur Dateien, die diesem Filtermuster entsprechen, können im System-Dialog ausgewählt werden. Das Filtermuster ist ein Zeichenkette, der eine Reihe von Dateierweiterungen enthält. Diese Endungen müssen durch das Zeichen '|' getrennt werden. Zum Beispiel: "voc|wav|8svx|16sv|iff|aiff" zeigt nur Dateien an, die eine dieser Erweiterungen haben.

Wenn Sie dieses Attribut nicht setzen, sind alle Dateien auswählbar.

TYP

Zeichenkette

ANWENDBARKEIT

I

36.4 Popfile.SaveMode

BEZEICHNUNG

Popfile.SaveMode – aktiviert den Speichermodus

BESCHREIBUNG

Setzen Sie diesen Tag auf **True**, um den Datei-System-Dialog in den Speichermodus zu versetzen.

TYP

Boolesch

ANWENDBARKEIT

I

36.5 Popfile.Title

BEZEICHNUNG

Popfile.Title – setzt den Titel des System-Dialog

BESCHREIBUNG

Legen Sie den Titel für den Datei-System-Dialog fest.

TYP

Zeichenkette

ANWENDBARKEIT

I

37 Popfont-Klasse (Schriftdialog)

37.1 Übersicht

Die Popfont-Klasse (Schriftdialog) leitet sich von der Area-Klasse ab und erstellt ein Widget, mit dem der Benutzer eine Schriftart auswählen kann. Schriftdialog-Widgets enthalten normalerweise eine Schaltfläche, die beim Anklicken einen System-Dialog öffnet, in dem der Benutzer eine Schriftart auswählen kann. Dies alles wird automatisch vom Widget verwaltet und Sie müssen nichts anderes tun, als das Widget irgendwo in Ihrem Fensterlayout zu platzieren.

Sie können den Titel des Popup-Dialogs mit dem Attribut `Popfile.Title` auf dem Popfile-Objekt setzen und die Auswahl des Benutzers mit dem Attribut `Popfont.Font` ermitteln.

37.2 Popfont.Font

BEZEICHNUNG

`Popfont.Font` – setzt/ermittelt die aktuelle Schriftauswahl

BESCHREIBUNG

Ermittelt oder setzt die aktuelle Schriftartenauswahl des Benutzers. Die Popfont-Klasse verwendet einen Schrägstrich zur Trennung von Schriftname und -größe, z.B. `Arial/48`.

Sie können auch eine Benachrichtigung für dieses Attribut einrichten, wenn der Benutzer eine neue Schriftart mit diesem Popfont-Objekt auswählt.

TYP

Zeichenkette

ANWENDBARKEIT

ISGN

37.3 Popfont.MaxSize

BEZEICHNUNG

`Popfont.MaxSize` – stellt die maximale Schrifthöhe ein

BESCHREIBUNG

Zeigt nur Schriften an, die gleich oder kleiner als diese Größe sind.

TYP

Zahl

ANWENDBARKEIT

I

37.4 Popfont.MinSize

BEZEICHNUNG

Popfont.MinSize – stellt die minimale Schrifthöhe ein

BESCHREIBUNG

Zeigt nur Schriften an, die gleich oder größer als diese Größe sind.

TYP

Zahl

ANWENDBARKEIT

I

37.5 Popfont.Title

BEZEICHNUNG

Popfont.Title – setzt den Titel des Schriftdialog-Fensters

BESCHREIBUNG

Setzt den Titel für das Schriftdialog-Fenster.

TYP

Zeichenkette

ANWENDBARKEIT

I

38 Popath-Klasse (Verzeichnisdialog)

38.1 Übersicht

Die Popath-Klasse (Verzeichnisdialog) leitet sich von der Area-Klasse ab und erstellt ein Widget, mit dem der Benutzer einen Pfad/ein Verzeichnis auswählen kann. Verzeichnisdialog-Widgets enthalten normalerweise eine Schaltfläche, die beim Anklicken einen Dialog öffnet, in dem der Benutzer einen Pfad auswählen kann. Dies alles wird automatisch vom Widget verwaltet und Sie müssen nichts anderes tun, als das Widget irgendwo in Ihrem Fensterlayout zu platzieren.

Sie können den Titel des Verzeichnisdialog mit dem Attribut `Poppath.Title` setzen und die Auswahl des Benutzers mit dem Attribut `Poppath.Path` ermitteln.

38.2 Popath.Path

BEZEICHNUNG

`Poppath.Path` – setzt den aktuellen Verzeichnisdialog-Pfad oder gibt ihn zurück

BESCHREIBUNG

Liefert oder setzt den aktuellen Pfad dieses Popath-Objekts.

Sie können auch eine Benachrichtigung für dieses Attribut einrichten, die ausgelöst wird, sobald der Benutzer einen neuen Pfad für dieses Popath-Objekt auswählt.

TYP

Zeichenkette

ANWENDBARKEIT

ISGN

38.3 Popath.Title

BEZEICHNUNG

`Poppath.Title` – setzt den Titel des Dialogfensters

BESCHREIBUNG

Setzen Sie die Titelzeichenfolge für das Dialogfenster.

TYP

Zeichenkette

ANWENDBARKEIT

I

39 Progressbar-Klasse (Fortschrittsbalken)

39.1 Übersicht

Die Progressbar-Klasse (Fortschrittsbalken) leitet sich von der Area-Klasse ab und ist ein Widget, welches Aufgaben oder Prozesse visualisiert, die einige Zeit benötigen. Normalerweise wird der Rest der Anwendung blockiert, während ein Fortschrittsbalken angezeigt wird.

Hier ist ein Beispiel, wie in XML ein Fortschrittsbalken definiert wird:

```
<progressbar/>
```

39.2 Progressbar.Horiz

BEZEICHNUNG

Progressbar.Horiz – stellt die Ausrichtung des Fortschrittsbalken ein

BESCHREIBUNG

Für einen vertikalen Fortschrittsbalken setzen Sie dies auf `False`. Standardmäßig wird ein horizontaler Fortschrittsbalken erstellt.

TYP

Boolesch

ANWENDBARKEIT

I

39.3 Progressbar.Level

BEZEICHNUNG

Progressbar.Level – stellt den aktuellen Pegel ein oder gibt ihn zurück

BESCHREIBUNG

Stellen Sie den aktuellen Pegel des Fortschrittsbalkens ein. Der neue Pegel des Fortschritts muss zwischen 0 und `Progressbar.Max` liegen. Sie können auch eine Benachrichtigung für dieses Attribut einrichten. Dies kann nützlich sein, wenn Sie ein anderes Widget aktualisieren möchten, wenn sich der aktuelle Pegel ändert.

TYP

Zahl

ANWENDBARKEIT

ISGN

39.4 Progressbar.Max

BEZEICHNUNG

Progressbar.Max – legt den Maximalen Bereich vom Fortschrittsbalken fest

BESCHREIBUNG

Stellen Sie den maximalen Bereich für den Fortschrittsbalken ein. Der Standardwert ist 100.

TYP

Zahl

ANWENDBARKEIT

ISG

40 Radio-Klasse

40.1 Übersicht

Die Radio-Klasse leitet sich von der Area-Klasse ab und erstellt eine Reihe von sich gegenseitig ausschließenden Schaltflächen (Buttons), die dem Benutzer eine einzige Auswahl ermöglicht. Die Schaltflächen sind innerhalb einer Gruppe mit einem Rahmen und einem optionalen Rahmentitel eingebettet.

Wenn Sie ein Radio-Widget in XML definieren, müssen Sie den Tag `<item>` verwenden, um das Radio-Widget mit Elementen zu füllen. Jedes Radio-Widget muss mindestens ein Element enthalten.

Hier ein Beispiel für einen XML-Auszug zum Erstellen eines Radio-Widgets:

```
<radio id="printer">
  <item>HP Deskjet</item>
  <item>NEC P6</item>
  <item>Okimate 20</item>
</radio>
```

Sie können das Attribut `Radio.Columns` verwenden, um das Layout der Radio-Schaltflächen zu optimieren. Standardmäßig werden sie in einem einspaltigen vertikalen Layout angezeigt. Wenn Sie `Radio.Columns` einstellen, können Sie dies beispielsweise in ein horizontales Layout ändern.

40.2 Radio.Active

BEZEICHNUNG

`Radio.Active` – setzt oder ermittelt die aktive Radio-Schaltfläche

BESCHREIBUNG

Legen Sie die aktive Schaltfläche im Radio-Widget fest oder rufen Sie sie ab, beginnend mit dem Index 0 für die erste Schaltfläche bis zur Anzahl aller Schaltflächen-1 für die letzte Schaltfläche.

Sie können auch eine Benachrichtigung für dieses Attribut einrichten, um zu erfahren, dass der Benutzer eine neue Schaltfläche auswählt.

TYP

Zahl

ANWENDBARKEIT

ISGN

40.3 Radio.Columns

BEZEICHNUNG

`Radio.Columns` – setzt die Anzahl der Radio-Spalten (V2.0)

BESCHREIBUNG

Legt die Anzahl der Spalten fest, die das Radio-Widget verwenden soll. Der Standardwert ist 1, was bedeutet, dass alle Radio-Schaltflächen in einer Spalte angezeigt werden.

Beachten Sie, dass Sie unter Android keine komplexen Radio-Layouts verwenden können, die Spalten und Zeilen verwenden. Sie können alle Radio-Schaltflächen entweder in einer Zeile (horizontales Layout) oder in einer Spalte (vertikales Layout) anzeigen lassen.

TYP

Zahl

ANWENDBARKEIT

I

40.4 Radio.GetItem

BEZEICHNUNG

Radio.GetItem – ermittelt die Beschriftung der Radio-Schaltfläche (V2.0)

ÜBERSICHT

```
e$ = moai.DoMethod(id, "GetItem", pos)
```

BESCHREIBUNG

Ruft die Beschriftung einer Radio-Schaltfläche ab. Sie können entweder einen absoluten Index in `pos` oder den speziellen Wert `Active` übergeben, um die Beschriftung der aktiven Schaltfläche zu ermitteln. `Radio.GetItem` gibt dann die Beschriftung als Zeichenkette zurück.

EINGABEN

<code>id</code>	ID des Radio-Objekts
<code>pos</code>	Index der Schaltfläche oder "Active"

RÜCKGABEWERTE

<code>e\$</code>	Beschriftung der Radio-Schaltfläche am angegebenen Index
------------------	--

40.5 Radio.SetItem

BEZEICHNUNG

Radio.SetItem – setzt die Beschriftung der Radio-Schaltfläche (V2.0)

ÜBERSICHT

```
moai.DoMethod(id, "SetItem", pos, newname$)
```

BESCHREIBUNG

Setzt die Beschriftung der Radio-Schaltfläche auf den in `newname$` angegebenen Namen. Der Index der zu verwendenden Radio-Schaltfläche wird im Argument `pos` angegeben. Dies kann eine absolute Indexposition beginnend bei 0 für die erste Schaltfläche oder einer der folgenden Sonderwerte sein:

<code>Active</code>	Ändert die Beschriftung der aktiven Schaltfläche.
---------------------	---

EINGABEN

<code>id</code>	ID des Radio-Objekts
<code>pos</code>	Schaltflächenposition als absolute Zahl oder spezieller Wert (siehe oben)
<code>newname\$</code>	neuer Beschriftungstext

40.6 Radio.Title**BEZEICHNUNG**

`Radio.Title` – legt den Titel für den Radio-Rahmen fest

BESCHREIBUNG

Der Text, den Sie hier angeben, wird als Titeltext in dem Rahmen angezeigt, der um die Radio-Schaltflächen gezeichnet wird.

Beachten Sie, dass das Ändern des Rahmentitels MUI 4.0 oder höher auf AmigaOS und kompatiblen Systemen erfordert.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

41 Rectangle-Klasse (Rechteck)

41.1 Übersicht

Die Rectangle-Klasse (Rechteck) leitet sich von der Area-Klasse ab und erstellt leere Rechteck-Objekte, die frei skalierbar sind. Dies mag auf den ersten Blick nicht sehr nützlich erscheinen, aber in der Tat werden häufig Rechteck-Objekte benötigt, um das GUI-Layout zu verfeinern. Sie können für alle Arten von Layouts-Aufgaben verwendet werden, z.B. zur Steuerung der Ausrichtung von Widgets mit fester Größe und vieles mehr.

Zusätzlich werden oft Rechteck-Objekte als Füllfläche neben Objekten benötigt, die selbst nicht größenveränderbar sind. Das Objekt `<radio>` ist beispielsweise nicht größenveränderbar. Um zu verhindern, dass diese statischen Radio-Objekte die Größenänderung Ihrer gesamte GUI blockiert, können Sie sie einfach mit `<rectangle>`-Objekten auffüllen. Hier ist ein Beispiel für ein Radio-Objekt, das mit einem unsichtbaren Rechteck aufgefüllt ist:

```
<hgroup>
  <rectangle/>
  <radio id="printer">
    <item>HP Deskjet</item>
    <item>NEC P6</item>
    <item>Okimate 20</item>
  </radio>
  <rectangle/>
</hgroup>
```

Durch die Verwendung von Rechtecken auf beiden Seiten des Radio-Box-Objekts wird die Radio-Box automatisch innerhalb des verfügbaren GUI-Raums zentriert. Wenn Sie nur ein Rechteck-Objekt vor dem Radio-Objekt verwenden, wird das Radio-Objekt automatisch rechtsbündig ausgerichtet. Ein Rechteck-Objekt nach dem Radio-Objekt würde zu einer linken Ausrichtung des Radio-Objekts führen.

Wenn Sie Objekte mit fester Größe benötigen, verwenden Sie die HSpace- oder VSpace-Klasse (HorizontalAbstand oder VertikalAbstand). Siehe [Abschnitt 21.1 \[HSpace-Klasse\]](#), [Seite 127](#), für Details. Siehe [Abschnitt 60.1 \[VSpace-Klasse\]](#), [Seite 313](#), für Details.

Die Rechteckklasse definiert keine Attribute.

42 Scrollbar-Klasse (Bildlaufleiste)

42.1 Übersicht

Die Scrollbar-Klasse (Bildlaufleiste) leitet sich von der Area-Klasse ab und erstellt ein Widget, das eine horizontale oder vertikale Bildlaufleiste darstellt. Ein isoliertes Scrollbar-Objekt macht nicht viel Sinn. Das ist der Grund, warum Scrollbar-Objekte normalerweise mit einem anderen Widget verbunden sind, indem das Attribut `Scrollbar.Target` verwendet wird. Beispielsweise können Sie Bildlaufleisten mit einem Objekt der Hollywood-Klasse verbinden, um ein benutzerdefiniertes Widget zu erstellen, welches Grafiken abhängig von der aktuellen Position der Bildlaufleiste anzeigt.

42.2 Scrollbar.AutoScale

BEZEICHNUNG

`Scrollbar.AutoScale` – aktiviert/deaktiviert die automatische Skalierung (V2.0)

BESCHREIBUNG

Standardmäßig verwenden alle Größen- und Positionswerte in der Scrollbar-Klasse geräteunabhängige Pixel. Wenn Sie dies nicht möchten, setzen Sie dieses Attribut auf `False`. In diesem Fall verwenden alle Werte stattdessen physische Pixel. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), [Seite 21](#), für Details.

Dieses Attribut entspricht standardmäßig dem, was im Tag `ScaleGUI` festgelegt wurde, wenn `@REQUIRE` mit dem RapaGUI-Plugin aufgerufen wird. Beachten Sie, dass `ScaleGUI` standardmäßig auf `True` gesetzt ist. Siehe [Abschnitt 3.4 \[Initialisierung von RapaGUI\]](#), [Seite 11](#), für Details.

TYP

Boolesch

ANWENDBARKEIT

I

42.3 Scrollbar.Horiz

BEZEICHNUNG

`Scrollbar.Horiz` – stellt die Richtung der Bildlaufleiste ein

BESCHREIBUNG

Legen Sie fest, ob Sie eine horizontale (`True`) oder vertikale (`False`) Bildlaufleiste wünschen. Standardmäßig wird eine vertikale Bildlaufleiste erstellt.

Wenn `Scrollbar.AutoScale` nicht auf `False` gesetzt ist, wird dieser Wert in geräteunabhängige Pixel angezeigt. Wenn Sie stattdessen physische Pixel haben wollen, müssen Sie `Scrollbar.AutoScale` auf `False` setzen.

TYP

Boolesch

ANWENDBARKEIT

IG

42.4 Scrollbar.Level**BEZEICHNUNG**

Scrollbar.Level – setzt/ermittelt die aktuelle Position des Schiebers

BESCHREIBUNG

Setzt oder ermittelt die aktuelle Position des Schiebers. In der Regel möchten Sie eine Benachrichtigung für dieses Attribut einrichten, wenn sich die Position des Schiebers ändert. So können Sie dynamisch auf diese Änderungen reagieren und ein weiteres Widget aktualisieren, das Ihre Bildlaufleiste steuern soll.

Wenn `Scrollbar.AutoScale` nicht auf `False` gesetzt ist, wird dieser Wert in geräteunabhängige Pixel angezeigt. Wenn Sie stattdessen physische Pixel haben wollen, müssen Sie `Scrollbar.AutoScale` auf `False` setzen.

TYP

Zahl

ANWENDBARKEIT

ISGN

42.5 Scrollbar.Range**BEZEICHNUNG**

Scrollbar.Range – setzt/ermittelt den Bereich der Bildlaufleiste

BESCHREIBUNG

Setzt oder ermittelt den Bereich der Bildlaufleiste.

Wenn `Scrollbar.AutoScale` nicht auf `False` gesetzt ist, wird dieser Wert in geräteunabhängige Pixel angezeigt. Wenn Sie stattdessen physische Pixel haben wollen, müssen Sie `Scrollbar.AutoScale` auf `False` setzen.

TYP

Zahl

ANWENDBARKEIT

ISG

42.6 Scrollbar.StepSize**BEZEICHNUNG**

Scrollbar.StepSize – setzt/ermittelt die Schrittweite der Bildlaufleiste

BESCHREIBUNG

Stellen Sie die Anzahl der zu scrollenden Pixel ein, wenn der Benutzer auf eine der Schaltflächen der Bildlaufleiste klickt.

Wenn `Scrollbar.AutoScale` nicht auf `False` gesetzt ist, wird dieser Wert in geräteunabhängige Pixel angezeigt. Wenn Sie stattdessen physische Pixel haben wollen, müssen Sie `Scrollbar.AutoScale` auf `False` setzen.

TYP

Zahl

ANWENDBARKEIT

ISG

42.7 Scrollbar.Target

BEZEICHNUNG`Scrollbar.Target` – setzt das Ziel-Widget für die Bildlaufleiste**BESCHREIBUNG**

Verwenden Sie dieses Attribut, um das Widget zu definieren, das von dieser Bildlaufleiste gesteuert werden soll. Dies sollte immer angegeben werden, da sich die Größe des Schiebers in der Bildlaufleiste verändert, wenn sich die Größe des Ziel-Widgets ändert. Wenn Sie hier ein Ziel-Widget angeben, aktualisiert RapaGUI automatisch die Größe des Schiebers in der Bildlaufleiste anhand der Größenänderung des Ziel-Widget. Andernfalls müssen Sie dies manuell vornehmen, indem Sie `Scrollbar.Visible` setzen.

TYP

MOAI-Objekt

ANWENDBARKEIT

I

42.8 Scrollbar.UseWinBorder

BEZEICHNUNG`Scrollbar.UseWinBorder` – fügt eine Bildlaufleiste in den Fensterrahmen ein**PLATTFORMEN**

Nur AmigaOS und kompatible Betriebssysteme

BESCHREIBUNG

Setzen Sie dieses Attribut, damit RapaGUI die Bildlaufleiste in den Fensterrahmen setzt, anstatt ein Widget zu erstellen, welches in ein Gruppenobjekt eingefügt werden kann.

Vor der Verwendung von `Scrollbar.UseWinBorder` müssen Sie zunächst die Randbildlaufleisten für das übergeordnete Fenster aktivieren, indem Sie die entsprechenden Attribute der Window-Klasse (Fenster) setzen. Wenn Sie beispielsweise die Bildlaufleiste in den unteren Fensterrand setzen möchten, müssen Sie zuerst das Attribut `Window.UseBottomBorderScroller` setzen. Siehe [Abschnitt 62.1 \[Window-Klasse\]](#), Seite 319, für Details.

Beachten Sie auch, dass es natürlich nur zwei Bildlaufleisten im Fensterrand geben kann: Einen am linken oder rechten sowie einen am unteren Fensterrand.

Die folgenden Werte werden durch dieses Attribut erkannt:

Left Benutzt den linken Fensterrand.
Right Benutzt den rechten Fensterrand.
Bottom Benutzt den unteren Fensterrand.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

I

42.9 Scrollbar.Visible

BEZEICHNUNG

Scrollbar.Visible – setzt/ermittelt die Anzahl der sichtbaren Einträge

BESCHREIBUNG

Setzt oder ermittelt die Anzahl der sichtbaren Einträge und aktualisiert die Größe des Schiebers in der Bildlaufleiste entsprechend. Wenn Sie kein Ziel-Widget mit **Scrollbar.Target** definiert haben, müssen Sie die Größe des Schiebers mit diesem Attribut aktualisieren, sobald sich die Größe des Ziel-Widgets ändert.

Wenn **Scrollbar.AutoScale** nicht auf **False** gesetzt ist, wird dieser Wert in geräteunabhängige Pixel angezeigt. Wenn Sie stattdessen physische Pixel haben wollen, müssen Sie **Scrollbar.AutoScale** auf **False** setzen.

TYP

Zahl

ANWENDBARKEIT

ISG

43 Scrollcanvas-Klasse (Bildlaufleinwand)

43.1 Übersicht

Die Scrollcanvas-Klasse (Bildlaufleinwand) leitet sich von der Area-Klasse ab und erstellt eine Zeichnungsfläche mit angehängten Bildlaufleisten. Sie können benutzerdefinierte Grafiken über eine Zeichnungs-Callback-Funktion auf dieser Zeichnungsfläche darstellen, die automatisch aufgerufen wird, wenn Inhalte gezeichnet werden sollen. Sie müssen nur eine Benachrichtigung für das Attribut `Scrollcanvas.Paint` einrichten, und Ihre Zeichnungs-Callback-Funktion wird immer dann aufgerufen, wenn Inhalte gezeichnet werden müssen.

Dasselbe kann auch erreicht werden, indem ein Widget, das von Hollywood-Klasse abgeleitet ist, in eine Bildlaufleisten-Gruppe integriert wird oder ein solches Hollywood-Widget mit Bildlaufleisten verbunden wird. Scrollcanvas-Klasse ist jedoch in einigen Fällen vorzuziehen, da sie speziell für Verschiebeinhalte optimiert ist. Sie versucht, das Zeichnen zu minimieren, indem OS-Widgets verwendet werden, die speziell zum Anzeigen von verschiebbaren Inhalt entwickelt wurden. Daher ist diese Klasse normalerweise schneller als die beiden oben beschriebenen Lösungen.

Sie können die Dimensionen der Zeichnungsfläche mithilfe von `Scrollcanvas.VirtWidth` und `Scrollcanvas.VirtHeight` festlegen. Da die Scrollcanvas-Klasse Bildlaufleisten verwendet, können die Abmessungen der Zeichnungsfläche natürlich viel größer sein als die physischen Abmessungen des Bildlaufleinwand-Widgets. Wie bei allen anderen Klassen können Sie diese physischen Dimensionen mit den generischen Attributen `Area.Width` und `Area.Height` festlegen.

Um ein vollständiges Neuzeichnen Ihres Widgets zu erzwingen, führen Sie einfach die Methode `Area.Redraw` auf Ihrem Objekt aus. Wenn Sie `Area.Redraw` für Ihr Objekt ausführen, wird Ihre Zeichnungs-Callback-Funktion aufgerufen, so dass Sie die Zeichnungsfläche entsprechend aktualisieren können.

Mit Scrollcanvas-Klasse sind die Abmessungen, die Sie in `Scrollcanvas.VirtWidth` und `Scrollcanvas.VirtHeight` angeben, standardmäßig in geräteunabhängigen Pixeln und RapaGUI wird automatisch den Skalierungsfaktor des Systems auf den von `Scrollcanvas.Paint` gezeichneten Inhalt anwenden. Wenn Sie eine genauere Steuerung wünschen, können Sie das Attribut `Scrollcanvas.AutoScale` auf `False` setzen. In diesem Fall werden `Scrollcanvas.VirtWidth` und `Scrollcanvas.VirtHeight` als physikalische Pixel interpretiert und es erfolgt keine automatische Skalierung, damit Ihre Zeichnungs-Funktion hochauflösende Grafiken ohne Qualitätsverlust aufgrund der Skalierung zeichnen kann. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), [Seite 21](#), für Details.

43.2 Scrollcanvas.AutoBars

BEZEICHNUNG

`Scrollcanvas.AutoBars` – stellt die Sichtbarkeit der Bildlaufleiste ein

BESCHREIBUNG

Standardmäßig werden die Bildlaufleisten automatisch ausgeblendet, wenn sie nicht benötigt werden. Wenn Sie dieses Verhalten nicht wünschen, setzen Sie dieses Attribut

auf `False`. In diesem Fall sind die Bildlaufleisten immer sichtbar, auch wenn sie nicht benötigt werden.

TYP

Boolesch

ANWENDBARKEIT

I

43.3 Scrollcanvas.AutoScale

BEZEICHNUNG

Scrollcanvas.AutoScale – aktiviert/deaktiviert die automatische Skalierung (V2.0)

BESCHREIBUNG

Standardmäßig verwendet die Scrollcanvas-Klasse geräteunabhängige Pixel und skaliert dann ihre Bildlaufleinwand, um sie an die DPI-Einstellungen des aktuellen Monitors anzupassen. Wenn Sie dies nicht möchten, setzen Sie dieses Attribut auf `False`. In diesem Fall arbeitet Ihr Scrollcanvas-Widget vollständig im physischen Pixelmodus und es wird keine automatische Skalierung durchgeführt. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), Seite 21, für Details.

Dieses Attribut entspricht standardmäßig dem, was im Tag `ScaleGUI` festgelegt wurde, wenn `@REQUIRE` mit dem RapaGUI-Plugin aufgerufen wird. Beachten Sie, dass `ScaleGUI` standardmäßig auf `True` gesetzt ist. Siehe [Abschnitt 3.4 \[Initialisierung von RapaGUI\]](#), Seite 11, für Details.

TYP

Boolesch

ANWENDBARKEIT

I

43.4 Scrollcanvas.Paint

BEZEICHNUNG

Scrollcanvas.Paint – benachrichtigt, wenn in die Zeichnungsfläche gemalt wird

BESCHREIBUNG

Richten Sie für dieses Attribut eine Überwachung ein, damit die Callback-Funktion immer dann aufgerufen wird, wenn Inhalte auf die Zeichnungsfläche gemalt werden müssen. Ihre Callback-Funktion kann dann den Zeichnungsflächen-Inhalt abhängig von der aktuellen Position der Bildlaufleiste zeichnen.

RapaGUI übergibt den Identifikator eines Hollywood-Pinsels, dessen Größe genau so groß ist wie der sichtbare Bereich Ihres Bildlaufleinwand-Widgets, an Ihre Callback-Funktion. Sie müssen dann den gewünschten Inhalt auf diesen Pinsel zeichnen. Sie können also nur in dem Rechteck zeichnen, das durch die vier Koordinaten `X`, `Y`, `Width` und `Height` definiert ist, die auch an Ihre Callback-Funktion übergeben werden. Diese vier Koordinaten beschreiben einen rechteckigen Bereich innerhalb der Dimensionen des Pinsels. Wenn ein

vollständiges Neuzeichnen benötigt wird, sind **X** sowie **Y** gleich 0 und **Width** sowie **Height** entsprechen den Abmessungen des Pinsels. Meistens wird jedoch nur ein partielles Neuzeichnen benötigt und dann müssen Sie nur auf den Teil des Pinsels zeichnen, der durch diese Koordinaten definiert ist.

Die folgenden zusätzlichen Argumente werden an Ihre Callback-Funktion übergeben:

Brush: Enthält den Identifikator eines Pinsels, auf den Sie zeichnen müssen. Verwenden Sie den Befehl `SelectBrush()` von Hollywood, um diesen Pinsel als Ausgabegerät in Ihrer Callback-Funktion auszuwählen. Vergessen Sie nicht, `EndSelect()` aufzurufen, wenn Sie fertig sind!

ViewWidth: Enthält die sichtbare Breite des Widgets. Die ist auch identisch mit der Breite des Pinsels, der an Ihre Callback-Funktion übergeben wird.

ViewHeight: Enthält die sichtbare Höhe des Widgets. Auch die ist identisch mit der Höhe des Pinsels, der an Ihre Callback-Funktion übergeben wird.

ScrollX: Enthält die Position des horizontalen Schiebers.

ScrollY: Enthält die Position des vertikalen Schiebers.

VirtWidth: Enthält die virtuelle Breite Ihres Widgets. Dies ist der mit `Scrollcanvas.VirtWidth` gesetzte Wert.

VirtHeight: Enthält die virtuelle Höhe Ihres Widgets. Dies ist der mit `Scrollcanvas.VirtHeight` gesetzte Wert.

X: Enthält die x-Position innerhalb des Pinsels, an der Sie mit dem Zeichnen beginnen sollen. Siehe oben für Details.

Y: Enthält die y-Position innerhalb des Pinsels, an der Sie mit dem Zeichnen beginnen sollen. Siehe oben für Details.

Width: Enthält die Anzahl der Spalten, die Sie auf den Pinsel malen sollten (ab **X**). Siehe oben für Details.

Height: Enthält die Anzahl der Zeilen, die Sie auf den Pinsel malen sollten (ab **Y**). Siehe oben für Details.

Um die absolute Position des Inhalts zu berechnen, der auf die Zeichnungsfläche gezeichnet werden soll, fügen Sie einfach die Koordinaten `ScrollX+X` sowie `ScrollY+Y` hinzu und Sie sind fertig.

Siehe [Abschnitt 3.7 \[Benachrichtigungen der Attribute\]](#), Seite 14, für Details.

TYP

Boolesch

ANWENDBARKEIT

N

43.5 Scrollcanvas.Scroll

BEZEICHNUNG

Scrollcanvas.Scroll – scrollt die Zeichnungsfläche

ÜBERSICHT

```
moai.DoMethod(id, "Scroll", x, y)
```

BESCHREIBUNG

Scrollt die Zeichnungsfläche an die in `x` und `y` angegebene Position.

EINGABEN

<code>id</code>	ID des Scrollcanvas-Objekts
<code>x</code>	gewünschte neue x-Position
<code>y</code>	gewünschte neue y-Position

43.6 Scrollcanvas.StepSize

BEZEICHNUNG

Scrollcanvas.StepSize – setzt/ermittelt die Schrittweite der Bildlaufleisten

BESCHREIBUNG

Legen Sie die Anzahl der zu scrollenden Pixel fest, wenn der Benutzer auf eine der Schaltflächen der Bildlaufleiste klickt. Ausserdem können Sie die zu scrollenden Pixel auch ermitteln.

TYP

Zahl

ANWENDBARKEIT

ISG

43.7 Scrollcanvas.UseLeftBorder

BEZEICHNUNG

Scrollcanvas.UseLeftBorder – verwendet linke statt rechte Bildlaufleiste im Fensterrahmen

PLATTFORMEN

Nur AmigaOS und kompatible Betriebssysteme

BESCHREIBUNG

Bei der Verwendung von `Scrollcanvas.UseWinBorder` werden die Bildlaufleisten standardmäßig in den rechten und unteren Fensterrahmen eingefügt. Wenn Sie die vertikale Bildlaufleiste stattdessen im linken Fensterrahmen haben möchten, legen Sie das Attribut `Scrollcanvas.UseLeftBorder` fest.

Vergessen Sie nicht, das Attribut `Window.UseLeftBorderScroller` ebenfalls festzulegen.

TYP

Boolesch

ANWENDBARKEIT

I

43.8 Scrollcanvas.UseWinBorder

BEZEICHNUNG

Scrollcanvas.UseWinBorder – verwendet die Bildlaufleisten im Fensterrahmen

PLATTFORMEN

Nur AmigaOS und kompatible Betriebssysteme

BESCHREIBUNG

Setzen Sie dieses Attribut, damit RapaGUI die Bildlaufleisten der Zeichnungsfläche in den Fensterrahmen einfügt, anstatt sie als normale Widgets zu erstellen.

Standardmäßig werden die Bildlaufleisten in den rechten und unteren Fensterrand gesetzt. Wenn Sie stattdessen die vertikale Bildlaufleiste im linken Fensterrand haben möchten, setzen Sie das Attribut `Scrollcanvas.UseLeftBorder`.

Bevor Sie `Scrollcanvas.UseWinBorder` verwenden, müssen Sie zunächst die Rahmen-Bildlaufleisten für das übergeordnete Fenster aktivieren, indem Sie die entsprechenden Fensterklassenattribute festlegen. Für die Standardkonfiguration (rechte und untere Rahmen-Bildlaufleisten) müssten Sie die Attribute `Window.UseBottomBorderScroller` und `Window.UseRightBorderScroller` festlegen. Siehe [Abschnitt 62.1 \[Window-Klasse\]](#), [Seite 319](#), für Details.

TYP

Boolesch

ANWENDBARKEIT

I

43.9 Scrollcanvas.VirtHeight

BEZEICHNUNG

Scrollcanvas.VirtHeight – setzt/ermittelt die Höhe der Zeichnungsfläche

BESCHREIBUNG

Setzt oder ermittelt die Höhe der Zeichnungsfläche. Normalerweise muss dieser Wert in geräteunabhängige Pixel angegeben werden. Wenn Sie möchten, dass Ihre Bildlaufleinwand anstelle von geräteunabhängige physische Pixel verwendet, müssen Sie das Attribut `Scrollcanvas.AutoScale` auf `False` setzen. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), [Seite 21](#), für Details.

Dies wird als "virtuelle Höhe" bezeichnet, da normalerweise nur ein Teil davon tatsächlich sichtbar ist. Der Benutzer kann die Bildlaufleisten verwenden, um den gesamten Zeichnungsflächeninhalt zu verschieben/durchzublättern.

Dieser Wert muss angegeben werden.

TYP

Zahl

ANWENDBARKEIT

ISG

43.10 Scrollcanvas.VirtWidth

BEZEICHNUNG

Scrollcanvas.VirtWidth – setzt/ermittelt die Breite der Zeichnungsfläche

BESCHREIBUNG

Setzt oder ermittelt die Breite der Zeichnungsfläche. Normalerweise muss dieser Wert in geräteunabhängige Pixel angegeben werden. Wenn Sie möchten, dass Ihre Bildlaufleinwand anstelle von geräteunabhängige physische Pixel verwendet, müssen Sie das Attribut `Scrollcanvas.AutoScale` auf `False` setzen. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), Seite 21, für Details.

Dies wird als "virtuelle Breite" bezeichnet, da normalerweise nur ein Teil davon tatsächlich sichtbar ist. Der Benutzer kann die Bildlaufleisten verwenden, um den gesamten Zeichnungsflächeninhalt zu verschieben/durchzublättern.

Dieser Wert muss angegeben werden.

TYP

Zahl

ANWENDBARKEIT

ISG

44 Scrollgroup-Klasse (Bildlaufgruppe)

44.1 Übersicht

Die Scrollgroup-Klasse (Bildlaufgruppe) ist eine spezielle Variante der Group-Klasse, die Bildlaufleisten zu Gruppen hinzufügt, um es zu ermöglichen, Gruppen zu verwenden, die größer als der verfügbare GUI-Bereich sind. Der Benutzer kann einfach durch die Gruppe mit den an die Gruppe angehängten Bildlaufleiste blättern. Wenn genügend Platz für die gesamte Gruppe vorhanden ist, werden die Bildlaufleisten automatisch ausgeblendet, es sei denn, sie sind explizit als sichtbar konfiguriert.

Scrollgruppen können ähnlich wie normale Gruppen erstellt werden. Hier ist ein XML-Beispiel:

```
<scrollgroup>
  <radio>
    <item>Amiga 500</item>
    <item>Amiga 1200</item>
    <item>Amiga 4000</item>
  </radio>
  <listview>
    <column/>
  </listview>
</scrollgroup>
```

Der obige XML-Code bettet ein Radio- und ein Listview-Objekt (Listenansicht) in eine Scrollgruppe ein.

44.2 Scrollgroup.AutoBars

BEZEICHNUNG

Scrollgroup.AutoBars – stellt die Sichtbarkeit der Bildlaufleisten ein

BESCHREIBUNG

Standardmäßig werden die Bildlaufleisten automatisch ausgeblendet, wenn sie nicht benötigt werden. Wenn Sie dieses Verhalten nicht wünschen, setzen Sie dieses Attribut auf `False`. In diesem Fall sind die Bildlaufleisten immer sichtbar, auch wenn sie nicht benötigt werden.

TYP

Boolesch

ANWENDBARKEIT

I

44.3 Scrollgroup.Horiz

BEZEICHNUNG

Scrollgroup.Horiz – setzt die Ausrichtung der Scrollgruppen

BESCHREIBUNG

Setzen Sie dies auf `True`, um eine horizontale Scrollgruppe zu erstellen. Standardmäßig verwenden Scrollgruppen die vertikale Ausrichtung.

TYP

Boolesch

ANWENDBARKEIT

I

44.4 Scrollgroup.UseWinBorder

BEZEICHNUNG

`Scrollgroup.UseWinBorder` – verwendet Fenster-Bildlaufleisten für diese Scrollgruppe

PLATTFORMEN

Nur AmigaOS und kompatible Betriebssysteme

BESCHREIBUNG

Setzen Sie dieses Attribut, damit RapaGUI die Bildlaufleisten der Scrollgruppe in den Fensterrand setzt, anstatt sie als normale Widgets zu erstellen.

Vor der Verwendung von `Scrollgroup.UseWinBorder` müssen Sie zunächst die Rand-Bildlaufleisten für das übergeordnete Fenster aktivieren, indem Sie die entsprechenden Fensterklassenattribute setzen. Wenn Sie beispielsweise eine Bildlaufleiste in den unteren Fensterrand einfügen möchten, müssen Sie zuerst das Attribut `Window.UseBottomBorderScroller` setzen. Siehe [Abschnitt 62.1 \[Window-Klasse\]](#), [Seite 319](#), für Details.

TYP

Boolesch

ANWENDBARKEIT

I

45 Slider-Klasse (Schieberegler)

45.1 Übersicht

Die Slider-Klasse (Schieberegler) leitet sich von der Area-Klasse ab und erstellt ein Widget, mit dem der Benutzer einen numerischen Wert mithilfe eines Schiebers anpassen kann, indem er vor und zurück gezogen wird, um den Wert zu ändern. Schieberegler können abhängig vom Attribut `Slider.Horiz` horizontale oder vertikale Ausrichtung verwenden. Standardmäßig wird ein horizontaler Schieberegler erstellt.

Hier ist ein XML-Beispiel für einen Schieberegler, mit dem der Benutzer eine Zahl zwischen 0 und 100 konfigurieren kann:

```
<slider min="0" max="100"/>
```

45.2 Slider.Drag

BEZEICHNUNG

Slider.Drag – benachrichtigt, wenn der Schieber gezogen wird (V2.0)

BESCHREIBUNG

Dieses Attribut wird immer dann gesetzt, wenn der Benutzer beginnt, den Schieber zu ziehen. Sie können für dieses Attribut eine Benachrichtigung einrichten, um informiert zu werden, wenn der Benutzer beginnt, den Schieber des Schiebereglers zu ziehen.

TYP

Boolesch

ANWENDBARKEIT

N

45.3 Slider.Horiz

BEZEICHNUNG

Slider.Horiz – setzt/ermittelt die Ausrichtung des Schiebereglers

BESCHREIBUNG

Geben Sie an, ob Sie einen horizontalen (`True`) oder vertikalen (`False`) Schieberegler wünschen. Standardwert ist `True`, was bedeutet, dass ein horizontaler Schieberegler erstellt wird.

TYP

Boolesch

ANWENDBARKEIT

I

45.4 Slider.Level

BEZEICHNUNG

Slider.Level – setzt/ermittelt die aktuelle Position des Schiebers

BESCHREIBUNG

Setzt/ermittelt die aktuelle Position des Schiebers im Schieberegler. Dieser Wert liegt immer zwischen `Slider.Min` und `Slider.Max`.

Sie können auch eine Benachrichtigung über dieses Attribut einrichten, um zu erfahren, wenn sich die Position des Schiebers ändert.

TYP

Zahl

ANWENDBARKEIT

ISGN

45.5 Slider.Max

BEZEICHNUNG

Slider.Max – setzt/ermittelt den Maximalwert des Schiebereglers

BESCHREIBUNG

Setzt oder ermittelt den Maximalwert des Schieberegler-Objekts.

TYP

Zahl

ANWENDBARKEIT

ISG

45.6 Slider.Min

BEZEICHNUNG

Slider.Min – setzt/ermittelt den Minimalwert des Schiebereglers

BESCHREIBUNG

Setzt oder ermittelt den Minimalwert des Schieberegler-Objekts. Dieser kann auch kleiner als 0 sein.

TYP

Zahl

ANWENDBARKEIT

ISG

45.7 Slider.Quiet

BEZEICHNUNG

Slider.Quiet – zeigt den aktuellen Schieberegler-Wert nicht an

BESCHREIBUNG

Stellen Sie dies auf `True`, damit der Schieberegler seinen aktuellen Wert ausblendet. Normalerweise wird der aktuelle Wert entweder im Schieber des Schiebereglers oder in einem Text-Widget neben dem Schieberegler angezeigt.

TYP

Boolesch

ANWENDBARKEIT

I

45.8 Slider.Release

BEZEICHNUNG

Slider.Release – benachrichtigt, wenn der Benutzer den Schieber loslässt

BESCHREIBUNG

Dieses Attribut wird immer dann gesetzt, wenn der Benutzer den Schieber loslässt. Sie können für dieses Attribut eine Benachrichtigung einrichten, um zu erfahren, wenn der Benutzer den Schieber loslässt.

TYP

Boolesch

ANWENDBARKEIT

N

45.9 Slider.Reverse

BEZEICHNUNG

Slider.Reverse – kehrt die Richtung des Schiebereglers um

BESCHREIBUNG

Setzen Sie dieses Attribut auf `True`, um die Richtung des Schiebereglers umzukehren.

TYP

Boolesch

ANWENDBARKEIT

I

45.10 Slider.StepSize

BEZEICHNUNG

Slider.StepSize – setzt/ermittelt die Schrittgröße des Schiebereglers (V2.0)

PLATTFORMEN

Windows, Linux, macOS

BESCHREIBUNG

Stellt die Schrittgröße des Schiebereglers ein oder ruft sie ab. Dieser Wert gibt an, um wie viel der Pegel des Schiebereglers erhöht oder verringert werden soll, wenn außerhalb des Schiebers geklickt wird. Die Standardschrittgröße ist 1.

TYP

Zahl

ANWENDBARKEIT

ISG

46 Statusbar-Klasse (Statusleiste)

46.1 Übersicht

Die Statusbar-Klasse (Statusleiste) leitet sich von der Area-Klasse ab und erstellt ein Widget am unteren Rand des Fensters, um einige Statusinformationen anzuzeigen. Statusleisten enthalten ein oder mehrere Elemente der Statusbaritem-Klasse (Statusleistenelement), die alle entweder feste oder variable Längen haben können.

Wenn Sie eine Statusleiste in XML erstellen, müssen Sie angeben, wie viele Felder sie enthalten soll, indem Sie den Tag `<item>` verwenden, um Elemente der Statusbaritem-Klasse zu Ihrer Statusleiste hinzuzufügen. Hier ist ein Beispiel für eine Statusleiste mit drei Feldern:

```
<statusbar>
  <item id="first">Welcome to my application</item>
  <item id="second"/>
  <item id="third" width="20"/>
</statusbar>
```

Die Breite der einzelnen Felder können Sie mit dem Attribut `Statusbaritem.Width` einstellen. Oben weisen wir nur dem dritten Feld eine feste Breite zu. Die anderen beiden verwenden variable Breiten.

Der Text von Statusleisten-Elementen kann später durch Setzen des Attributs `Statusbaritem.Text` geändert werden. Beachten Sie jedoch, dass dies oft nicht notwendig ist, da der in den Attributen `MenuItem.Help` und `ToolBarbutton.Help` enthaltene Text automatisch in der Statusleiste angezeigt wird und keinen zusätzlichen Code von Ihrer Seite benötigt.

Es kann nur eine Statusleiste pro Fenster geben und sie muss immer das letzte Element der Wurzelgruppe des Fensters sein. RapaGUI akzeptiert keine Statusleisten an beliebigen Positionen im GUI-Layout. Sie befinden sich also immer am unteren Rand des Fensterlayouts. Beachten Sie auch, dass es nicht möglich ist, eigenständige Instanzen dieser Klasse mit `moai.CreateObject()` zu erzeugen. Statusleisten müssen immer im Kontext eines Fensters erstellt werden. Wenn Sie also mit `moai.CreateObject()` Statusleisten erstellen wollen, müssen Sie immer ein komplettes Fenster erstellen und die Statusleiste in diese Fensterdefinition einbetten.

Die Statusbar-Klasse definiert selbst keine Attribute oder Methoden. Siehe [Abschnitt 47.1 \[Statusbaritem-Klasse\]](#), [Seite 241](#), für alle notwendigen Informationen.

47 Statusbaritem-Klasse (Statusleistenelement)

47.1 Übersicht

Die Statusbaritem-Klasse (Statusleistenelement) wird verwendet, um Textfelder zu erstellen, die als untergeordnete Elemente für Statusbar-Klasse (Statusleiste) verwendet werden. Sie können keine unabhängigen Instanzen der Statusbaritem-Klasse erstellen. Sie müssen immer in die Statusbar-Klasse eingebettet werden. Siehe [Abschnitt 46.1 \[Statusbar-Klasse\]](#), [Seite 239](#), für Details.

47.2 Statusbaritem.Text

BEZEICHNUNG

Statusbaritem.Text – setzt oder ermittelt den Text des Statusleisteneintrags

BESCHREIBUNG

Setzt den Text des Statusleisteneintrags oder gibt ihn zurück.

TYP

Zeichenkette

ANWENDBARKEIT

SG

47.3 Statusbaritem.Width

BEZEICHNUNG

Statusbaritem.Width – stellt die Breite des Statusleisteneintrags ein

BESCHREIBUNG

Stellen Sie die gewünschte Breite der Statusleisteneintrags ein. Dies kann entweder ein absoluter Wert in geräteunabhängige Pixel für ein Feld mit fester Breite oder ein negativer Wert sein, der einen Anteil für ein Feld mit variabler Breite angibt. Der verbleibende Platz für alle Einträge mit variabler Breite wird entsprechend dem Absolutwert dieser Zahl zwischen ihnen aufgeteilt (Absolutwert ist die Zahl ohne Vorzeichen). Ein variabler Eintrag mit einer Breite von -2 erhält doppelt so viel Platz wie ein Eintrag mit einer Breite von -1 und so weiter (Absolutwert von -1 ist 1, von -2 ist 2).

Um z.B. ein Eintrag mit fester Breite der Breite 100 im rechten Teil der Statusleiste und zwei weitere Einträge zu erstellen, die entsprechend 66% bzw. 33% des verbleibenden Platzes erhalten, würden Sie die Breitenfelder der Elemente auf -2, -1 bzw. 100 setzen.

Voreingestellt ist -1.

TYP

Zahl

ANWENDBARKEIT

I

48 Text-Klasse

48.1 Übersicht

Die Text-Klasse leitet sich von der Area-Klasse ab und erzeugt Widgets, die eine oder mehrere Zeilen Text anzeigen, ähnlich wie die Widgets der Label-Klasse (Beschriftung). Ein Unterschied besteht darin, dass die Text-Klasse mehrere Textzeilen sowie Rahmen um den Text unterstützt. Widgets, die von der Text-Klasse erzeugt werden, sind in der Größe veränderbar, während die Label-Klasse horizontal nicht skalierbar ist. Wie Widgets der Label-Klasse können auch Widgets der Text-Klasse ein Unterstrichzeichen enthalten, um ein Tastaturkürzel für den Zugriff über die Tastatur festzulegen.

Bitte beachten Sie, dass die Text-Klasse weder automatischen Zeilenumbruch noch Textformatierung unterstützt. Wenn Sie diese Funktionen haben wollen, müssen Sie die Textview-Klasse (Textanzeige) verwenden. Siehe [Abschnitt 51.1 \[Textview-Klasse\]](#), [Seite 267](#), für Details.

Hier ist ein Beispiel für die Verwendung des Tags `<text>` in XML:

```
<text>Hello World</text>
```

48.2 Text.Align

BEZEICHNUNG

Text.Align – setzt die Textausrichtung oder gibt sie zurück

BESCHREIBUNG

Stellen Sie die gewünschte Ausrichtung für den Text ein. Dies kann einer der folgenden Werte sein:

Left Links ausgerichtet. Dies ist voreingestellt.

Right Rechts ausgerichtet.

Center Zentriert ausgerichtet.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

ISG

48.3 Text.Frame

BEZEICHNUNG

Text.Frame – zeichnet einen Rahmen um das Text-Widget

BESCHREIBUNG

Setzen Sie dies auf `True`, um einen Rahmen um das Text-Widget zu ziehen.

TYP

Boolesch

ANWENDBARKEIT

I

48.4 Text.Text**BEZEICHNUNG**

Text.Text – setzt die Textzeichenfolge oder gibt sie zurück

BESCHREIBUNG

Setzen oder ermitteln Sie die Textzeichenfolge. Der Text kann Zeilenumbrüche enthalten.

TYP

Zeichenkette

ANWENDBARKEIT

SG

49 Texteditor-Klasse

49.1 Übersicht

Die Texteditor-Klasse leitet sich von der Area-Klasse ab und erstellt mehrzeilige Texteingabe-Widgets mit den meisten Funktionen eines normalen Texteditors. Es unterstützt auch die Textformatierung über bestimmte Zeichencodes, wenn `Texteditor.Styled` festgelegt wurde. Dadurch können Sie bestimmte Stile (Fett, Kursiv, Unterstrichen) für Ihren Text aktivieren sowie die Farbe Ihres Textes ändern. Siehe [Abschnitt 3.14 \[Textformatierungscodes\]](#), Seite 24, für Details. Der Inhalt des XML-Tags wird als anfänglicher Inhalt des Texteditor-Widgets verwendet.

Hier ist ein XML-Beispiel, wie Sie ein Texteditor-Objekt in Ihre GUI einbinden können:

```
<texteditor>Enter your text here!</texteditor>
```

Beachten Sie, dass es einen kleinen Unterschied zwischen dem Texteditor-Widget unter Windows und allen anderen Plattformen gibt: Windows' natives Texteditor-Widget verwendet zwei Zeichen für einen Zeilenumbruch (CRLF, d.h. Wagenrücklauf und Zeilenvorschub), während auf allen anderen Plattformen nur ein Zeilenvorschubzeichen verwendet wird. Dies führt zu dem Problem, dass fest codierte Index- oder Bereichspositionen nicht vollständig portierbar sind, da Windows immer zwei Zeichen zum Starten einer neuen Zeile verwendet. Dieses Problem müssen Sie beachten.

Beachten Sie auch, dass diese Klasse auf AmigaOS und kompatiblen Systemen die Erweiterung `TextEditor.mcc` benötigt.

49.2 Texteditor.Align

BEZEICHNUNG

`Texteditor.Align` – setzt/ermittelt die Textausrichtung

BESCHREIBUNG

Setzt oder ermittelt die Textausrichtung. Folgende Werte sind möglich:

<code>Left</code>	Links ausgerichtet.
<code>Right</code>	Rechts ausgerichtet.
<code>Center</code>	Zentriert ausgerichtet.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

ISG

49.3 Texteditor.AreaMarked

BEZEICHNUNG

`Texteditor.AreaMarked` – benachrichtigt, wenn Text makriert ist

BESCHREIBUNG

Dieser Tag wird auf `True` gesetzt, wenn Text markiert wurde, und zurück auf `False`, wenn nichts (mehr) markiert ist. Sie können eine Benachrichtigung mit diesem Tag erstellen und Ihre Schaltflächen wie Ausschneiden und Kopieren deaktiviert lassen, wenn nichts markiert ist.

TYP

Boolesch

ANWENDBARKEIT

GN

49.4 Texteditor.Bold

BEZEICHNUNG

`Texteditor.Bold` – setzt/ermittelt den Schriftstil Fett

BESCHREIBUNG

Dieser Tag zeigt an, ob der Cursor gerade Fett schreibt oder der Block fett formatierter Text ist oder nicht. Sie können eine Benachrichtigung für diesen Tag einrichten, um sich über Stiländerungen zu informieren. Sie können dieses Tag auf `True` für Fett oder `False` für die originale Schriftdicke setzen, wenn Sie den Stil ändern möchten.

Beachten Sie, dass `Texteditor.Styled` auf `True` gesetzt werden muss, um diese Fähigkeit zu benutzen.

TYP

Boolesch

ANWENDBARKEIT

SGN

49.5 Texteditor.Clear

BEZEICHNUNG

`Texteditor.Clear` – löscht den gesamten Text

ÜBERSICHT

```
moai.DoMethod(id, "Clear")
```

BESCHREIBUNG

Dadurch wird der gesamte Text im Widget gelöscht.

Beachten Sie, dass obwohl das Setzen dieses Attributs keine `Texteditor.HasChanged`-Benachrichtigung auslöst, es `Texteditor.HasChanged` auf `False` setzt.

EINGABEN

`id` ID des Texteditor-Objekts

49.6 Texteditor.Color

BEZEICHNUNG

Texteditor.Color – setzt/ermittelt die aktuelle Textfarbe

BESCHREIBUNG

Mit diesem Attribut kann die aktuelle Textfarbe eingestellt werden. Jeder Text, der nach dem Setzen dieses Attributs eingegeben wird, erscheint in der angegebenen Farbe. Um die Farbe des vorhandenen Textes zu ändern, verwenden Sie stattdessen die Methode `Texteditor.SetColor`. Die Farbe muss als 24-Bit-RGB-Wert übergeben werden.

Sie können auch eine Benachrichtigung über dieses Attribut einrichten, um zu erfahren, wenn der Cursor über Text in einer anderen Farbe bewegt wurde.

Beachten Sie, dass `Texteditor.Styled` auf `True` gesetzt werden muss, um diese Fähigkeit zu benutzen.

TYP

Zahl

ANWENDBARKEIT

SGN

49.7 Texteditor.Copy

BEZEICHNUNG

Texteditor.Copy – kopiert den markierten Text

ÜBERSICHT

```
moai.DoMethod(id, "Copy")
```

BESCHREIBUNG

Kopiert den aktuell ausgewählten Text in die Zwischenablage.

EINGABEN

id ID des Texteditor-Objekts

49.8 Texteditor.CursorPos

BEZEICHNUNG

Texteditor.CursorPos – setzt/ermittelt die Cursorposition

BESCHREIBUNG

Mit diesem Tag können Sie die Position des Cursors setzen oder ermitteln. Das Anfangszeichen beginnt an Position 0.

Sie können auch eine Benachrichtigung für diesem Tag einrichten, um zu erfahren, wann der Cursor bewegt wird.

TYP

Zahl

ANWENDBARKEIT

SGN

49.9 Texteditor.Cut**BEZEICHNUNG**

Texteditor.Cut – schneidet den markierten Text aus

ÜBERSICHT`moai.DoMethod(id, "Cut")`**BESCHREIBUNG**

Schneidet den aktuell markierten Text aus und legt ihn in die Zwischenablage.

EINGABEN`id` ID des Texteditor-Objekts**49.10 Texteditor.GetLineLength****BEZEICHNUNG**

Texteditor.GetLineLength – ermittelt die Anzahl der Zeichen in der Zeile (V2.0)

ÜBERSICHT`len = moai.DoMethod(id, "GetLineLength", line)`**BESCHREIBUNG**

Diese Methode gibt die Anzahl der Zeichen in der durch `line` angegebenen Zeile zurück. Nachfolgende Zeichen wie Wagenrücklauf oder Zeilenvorschub werden nicht mitgezählt. Zeilenindizes werden ab 0 gezählt.

EINGABEN`id` ID des Texteditor-Objekts`line` Index der abzufragenden Zeile (beginnend mit 0)**RÜCKGABEWERTE**`len` Anzahl der Zeichen in der Zeile**49.11 Texteditor.GetPosition****BEZEICHNUNG**

Texteditor.GetPosition – ermittelt die Indexposition aus Spalte und Zeile (V2.0)

ÜBERSICHT`pos = moai.DoMethod(id, "GetPosition", x, y)`**BESCHREIBUNG**

Diese Methode wandelt die durch `x` und `y` angegebene Spalten- und Zeilenposition in eine Indexposition um. Alle Werte beginnen bei Index 0. Bei ungültigen Spalten- und Zeilenpositionen wird -1 zurückgegeben.

EINGABEN

id	ID des Texteditor-Objekts
x	Spaltenposition
y	Zeilenposition

RÜCKGABEWERTE

pos	Index der angegebenen Spalten- und Zeilenposition
-----	---

49.12 Texteditor.GetSelection**BEZEICHNUNG**

Texteditor.GetSelection – ermittelt die Textauswahl

ÜBERSICHT

```
start, end = moai.DoMethod(id, "GetSelection")
```

BESCHREIBUNG

Diese Methode gibt die Start- und Endposition des aktuell ausgewählten Textes zurück. Das erste Zeichen des Gesamttextes steht an Position 0. Ist kein Bereich markiert, wird für beide Werte -1 zurückgegeben.

EINGABEN

id	ID des Texteditor-Objekts
----	---------------------------

RÜCKGABEWERTE

start	Startversatz des markierten Textes
end	Endversatz des markierten Textes

49.13 Texteditor.GetText**BEZEICHNUNG**

Texteditor.GetText – exportiert einen Teil des aktuellen Textes

ÜBERSICHT

```
t$ = moai.DoMethod(id, "GetText", start, end)
```

BESCHREIBUNG

Diese Methode exportiert den Teil des aktuellen Textes zwischen `start` und `end` und gibt ihn zurück. Positionen werden im Gesamttext ab 0 gezählt.

EINGABEN

id	ID des Texteditor-Objekts
start	Startposition des gewünschten Blocks
end	Endposition des gewünschten Blocks

RÜCKGABEWERTE

t\$	Text der angegebenen Blockkoordinaten
-----	---------------------------------------

49.14 Texteditor.GetXY

BEZEICHNUNG

Texteditor.GetXY – wandelt die Indexposition in Spalte und Zeile um

ÜBERSICHT

```
x, y = moai.DoMethod(id, "GetXY", pos)
```

BESCHREIBUNG

Diese Methode konvertiert die angegebene Indexposition in ihre Spalten- und Zeilenkoordinaten. Sowohl Spalten- als auch Zeilenzähler beginnen bei Index 0. Bei ungültigen Indizes wird -1 zurückgegeben.

EINGABEN

id ID des Texteditor-Objekts
pos zu konvertierende Indexposition

RÜCKGABEWERTE

x Spaltenoffset des angegebenen Indexes
y Zeilenoffset des angegebenen Indexes

49.15 Texteditor.HasChanged

BEZEICHNUNG

Texteditor.HasChanged – benachrichtigt, wenn sich der Textinhalt ändert

BESCHREIBUNG

Dieses Attribut wird gesetzt, wenn der Benutzer den Inhalt des Textes ändert. Sie können eine Benachrichtigung zu diesem Attribut einrichten, um sich über diese Änderungen zu informieren.

Beachten Sie, dass obwohl `Texteditor.HasChanged` nicht ausgelöst wird, wenn `Texteditor.Text` gesetzt oder `Texteditor.Clear` aufgerufen wird, diese beiden immer noch dazu führen, dass `Texteditor.HasChanged` auf `False` gesetzt wird.

TYP

Boolesch

ANWENDBARKEIT

SGN

49.16 Texteditor.Hint

BEZEICHNUNG

Texteditor.Hint – setzt/ermittelt den Texthinweis (V2.0)

BESCHREIBUNG

Legt den Hinweis eines Texteditor-Widgets fest oder ruft ihn ab. Der Hinweis wird angezeigt, wenn kein Text im Widget vorhanden ist. Er kann verwendet werden, um dem Benutzer einen Hinweis darauf zu geben, was er in das Widget eingeben soll.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

49.17 Texteditor.Insert

BEZEICHNUNG

Texteditor.Insert – fügt Text ein

ÜBERSICHT

```
moai.DoMethod(id, "Insert", t$, pos$)
```

BESCHREIBUNG

Dadurch wird der in `t$` angegebene Text an der in `pos$` angegebenen Position eingefügt. Die Position des eingefügten Textes kann eine der folgenden Werte sein:

Cursor Fügt den Text an der aktuellen Cursorposition ein.

Top Fügt den Text ganz am Anfang ein.

Bottom Hängt den Text am Schluss an.

EINGABEN

`id` ID des Texteditor-Objekts

`t$` der einzufügende Text

`pos$` Einfügeposition (siehe oben für mögliche Werte)

49.18 Texteditor.Italic

BEZEICHNUNG

Texteditor.Italic – setzt/ermittelt den Schriftstil Italic

BESCHREIBUNG

Dieser Tag zeigt an, ob der Cursor gerade kursiv schreibt oder der Block kursiv formatierter Text ist oder nicht. Sie können eine Benachrichtigung für diesen Tag einrichten, um sich über Stiländerungen zu informieren. Wenn Sie den Stil ändern möchten, können Sie diesen Tag auf `True` für Kursiv setzen oder `False`, um Kursiv aufzuheben.

Beachten Sie, dass `Texteditor.Styled` auf `True` gesetzt werden muss, um diese Fähigkeit zu benutzen.

TYP

Boolesch

ANWENDBARKEIT

SGN

49.19 Texteditor.Mark

BEZEICHNUNG

Texteditor.Mark – markiert den Text

ÜBERSICHT

```
moai.DoMethod(id, "Mark", start, end)
```

BESCHREIBUNG

Diese Methode markiert den Text in dem durch `start` und `end` begrenzten Bereich. Das erste Zeichen des Gesamttextes beginnt bei Position 0.

EINGABEN

<code>id</code>	ID des Texteditor-Objekts
<code>start</code>	Startposition
<code>end</code>	Endposition

49.20 Texteditor.MarkAll

BEZEICHNUNG

Texteditor.MarkAll – markiert den gesamten Text

ÜBERSICHT

```
moai.DoMethod(id, "MarkAll")
```

BESCHREIBUNG

Markiert den gesamten Text im Widget.

EINGABEN

<code>id</code>	ID des Texteditor-Objekts
-----------------	---------------------------

49.21 Texteditor.MarkNone

BEZEICHNUNG

Texteditor.MarkNone – löscht den ausgewählten Text

ÜBERSICHT

```
moai.DoMethod(id, "MarkNone")
```

BESCHREIBUNG

Löscht den ausgewählten Text.

EINGABEN

<code>id</code>	ID des Texteditor-Objekts
-----------------	---------------------------

49.22 Texteditor.NoWrap

BEZEICHNUNG

Texteditor.NoWrap – deaktiviert den Wortumbruch

BESCHREIBUNG

Standardmäßig verwenden Texteditor-Widgets automatischen Wortumbruch, wenn Wörter über den verfügbaren Widget-Platz hinaus laufen. Setzen Sie dieses Tag auf `True`, wenn Sie das nicht wollen. In diesem Fall verwendet das Texteditor-Widget eine horizontale Bildlaufleiste, anstatt Wörter in die nächste Zeile einzufügen.

TYP

Boolesch

ANWENDBARKEIT

I

49.23 Texteditor.Paste

BEZEICHNUNG

Texteditor.Paste – fügt den Text aus der Zwischenablage ein

ÜBERSICHT

```
moai.DoMethod(id, "Paste")
```

BESCHREIBUNG

Fügt den Text aus der Zwischenablage in das Texteditor-Widget ein.

EINGABEN

`id` ID des Texteditor-Objekts

49.24 Texteditor.ReadOnly

BEZEICHNUNG

Texteditor.ReadOnly – setzt den Texteditor in den Nur-Lese-Modus

BESCHREIBUNG

Um das Widget in den Nur-Lese-Modus zu versetzen, setzen Sie diesen Tag auf `True`.

Dies ist wahrscheinlich nicht sehr nützlich, da es die Textview-Klasse (Textanzeige) für die Anzeige von nicht editierbarem Text gibt. Siehe [Abschnitt 51.1 \[Textview-Klasse\]](#), [Seite 267](#), für Details.

TYP

Boolesch

ANWENDBARKEIT

I

49.25 Texteditor.Redo

BEZEICHNUNG

Texteditor.Redo – hebt ein vorangehendes Undo auf

ÜBERSICHT

```
moai.DoMethod(id, "Redo")
```

BESCHREIBUNG

Mit Redo können Sie ein vorangegangenes Undo des Texteditor-Widgets wieder aufheben.

EINGABEN

`id` ID des Texteditor-Objekts

49.26 Texteditor.RedoAvailable

BEZEICHNUNG

Texteditor.RedoAvailable – benachrichtigt, wenn Redo verfügbar ist

BESCHREIBUNG

Dieser Tag wird auf `True` gesetzt, wenn der Benutzer ein vorangehendes Undo aufheben kann. Sie können eine Benachrichtigung für diesem Tag erstellen und Ihre Redo-Schaltfläche deaktivieren, wenn es kein Undo zum aufheben hat.

TYP

Boolesch

ANWENDBARKEIT

GN

49.27 Texteditor.ScrollToLine

BEZEICHNUNG

Texteditor.ScrollToLine – verschiebt die Zeile in die Ansicht (V2.0)

ÜBERSICHT

```
moai.DoMethod(id, "ScrollToLine", line)
```

BESCHREIBUNG

Scrollt die durch `line` angegebene Zeile in den sichtbaren Bereich. Linienindizes beginnen bei 0.

EINGABEN

`id` ID des Texteditor-Objekts

`line` Zeile, die angezeigt werden soll (beginnend bei 0)

49.28 Texteditor.SetBold

BEZEICHNUNG

Texteditor.SetBold – aktiviert/deaktiviert den Stil Fett des Textblocks

ÜBERSICHT

```
moai.DoMethod(id, "SetBold", start, end, flag)
```

BESCHREIBUNG

Diese Methode aktiviert oder deaktiviert den fettgedruckten Stil des Textblocks, der durch die Koordinaten `start` und `end` definiert ist. Wenn das Argument `flag` auf `True` gesetzt ist, wird der angegebenen Textblock fett ausgegeben, andernfalls wird der fette Stil aus dem Textblock entfernt.

Beachten Sie, dass `Texteditor.Styled` auf `True` gesetzt werden muss, um diese Fähigkeit zu benutzen.

EINGABEN

<code>id</code>	ID des Texteditor-Objekts
<code>start</code>	Startposition (Gesamttext beginnt bei 0)
<code>end</code>	Endposition
<code>flag</code>	mit <code>True</code> wird der Textblock fett ausgegeben, mit <code>False</code> hingegen aufgehoben

49.29 Texteditor.SetColor

BEZEICHNUNG

Texteditor.SetColor – ändert die Farbe des Textblocks

ÜBERSICHT

```
moai.DoMethod(id, "SetColor", start, end, color)
```

BESCHREIBUNG

Diese Methode ändert die Farbe eines Textblocks, der durch die Koordinaten `start` und `end` definiert ist. Die Farbe muss als 24-Bit-RGB-Wert in `color` übergeben werden.

Um die Farbe zu ändern, die für neu eingefügten Text verwendet werden soll, verwenden Sie stattdessen das Attribut `Texteditor.Color`.

Beachten Sie, dass `Texteditor.Styled` auf `True` gesetzt werden muss, um diese Fähigkeit zu benutzen.

EINGABEN

<code>id</code>	ID des Texteditor-Objekts
<code>start</code>	Startposition (Gesamttext beginnt bei 0)
<code>end</code>	Endposition
<code>color</code>	gewünschte Farbe für den Text

49.30 Texteditor.SetItalic

BEZEICHNUNG

Texteditor.SetItalic – aktiviert/deaktiviert den Stil Italic des Textblocks

ÜBERSICHT

```
moai.DoMethod(id, "SetItalic", start, end, flag)
```

BESCHREIBUNG

Diese Methode aktiviert oder deaktiviert den Kursivdruck des Textblocks, der durch die Koordinaten `start` und `end` definiert ist. Wenn das Argument `flag` auf `True` gesetzt ist, wird der angegebenen Textblock kursiv ausgegeben, andernfalls wird der kursive Stil aus dem Textblock entfernt.

Beachten Sie, dass `Texteditor.Styled` auf `True` gesetzt werden muss, um diese Fähigkeit zu benutzen.

EINGABEN

<code>id</code>	ID des Texteditor-Objekts
<code>start</code>	Startposition (Gesamttext beginnt bei 0)
<code>end</code>	Endposition
<code>flag</code>	mit <code>True</code> wird der Textblock kursiv ausgegeben, mit <code>False</code> hingegen aufgehoben

49.31 Texteditor.SetUnderline

BEZEICHNUNG

Texteditor.SetUnderline – aktiviert/deaktiviert den Stil Unterstrichen des Textblocks

ÜBERSICHT

```
moai.DoMethod(id, "SetUnderline", start, end, flag)
```

BESCHREIBUNG

Diese Methode aktiviert oder deaktiviert den Unterstreichungsstil des Textblocks, der durch die Koordinaten `start` und `end` definiert ist. Wenn das Argument `flag` auf `True` gesetzt ist, wird der angegebenen Textblock unterstrichen ausgegeben, andernfalls wird der Unterstreichungsstil aus dem Textblock entfernt.

Beachten Sie, dass `Texteditor.Styled` auf `True` gesetzt werden muss, um diese Fähigkeit zu benutzen.

EINGABEN

<code>id</code>	ID des Texteditor-Objekts
<code>start</code>	Startposition (Gesamttext beginnt bei 0)
<code>end</code>	Endposition
<code>flag</code>	mit <code>True</code> wird der Textblock unterstrichen ausgegeben, mit <code>False</code> hingegen aufgehoben

49.32 Texteditor.Styled

BEZEICHNUNG

Texteditor.Styled – aktiviert die Textformatierung

BESCHREIBUNG

Setzen Sie dieses Attribut auf `True`, um die Textformatierung für dieses Widget zu aktivieren. Wenn diese Option auf `True` gesetzt ist, können Sie in dem Text, den Sie an dieses Widget übergeben, spezielle Textformatierungscodes verwenden. Siehe [Abschnitt 3.14 \[Textformatierungscodes\]](#), Seite 24, für Details.

TYP

Boolesch

ANWENDBARKEIT

I

49.33 Texteditor.Text

BEZEICHNUNG

Texteditor.Text – setzt/ermittelt den Inhalt des Texteditors

BESCHREIBUNG

Verwenden Sie dieses Attribut, um den Inhalt des Texteditor-Objekts zu setzen oder zu ermitteln.

Die hier angegebene Zeichenkette kann Textformatierungscodes verwenden, wenn `Texteditor.Styled` auf `True` gesetzt wurde. Siehe [Abschnitt 3.14 \[Textformatierungscodes\]](#), Seite 24, für Details.

Beachten Sie, dass obwohl das Setzen dieses Attributs keine `Texteditor.HasChanged`-Benachrichtigung auslöst, es `Texteditor.HasChanged` auf `False` setzt.

TYP

Zeichenkette

ANWENDBARKEIT

SG

49.34 Texteditor.Underline

BEZEICHNUNG

Texteditor.Underline – setzt/ermittelt den Schriftstil Unterstrichen

BESCHREIBUNG

Dieser Tag zeigt an, ob der Cursor oder Block über unterstrichenem Text steht oder nicht. Sie können eine Benachrichtigung für diesen Tag einrichten, um sich über Stiländerungen zu informieren. Sie können diesen Tag auf `True` (unterstrichen) oder `False` (unterstrichen aufgehoben) setzen, wenn Sie den Stil ändern möchten.

Beachten Sie, dass `Texteditor.Styled` auf `True` gesetzt werden muss, um diese Fähigkeit zu benutzen.

TYP

Boolesch

ANWENDBARKEIT

SGN

49.35 Texteditor.Undo

BEZEICHNUNG

Texteditor.Undo – Undo macht die letzte Operation rückgängig

ÜBERSICHT

```
moai.DoMethod(id, "Undo")
```

BESCHREIBUNG

Mit Undo wird die letzte Operation des Texteditor-Widgets rückgängig gemacht.

EINGABEN

id ID des Texteditor-Objekts

49.36 Texteditor.UndoAvailable

BEZEICHNUNG

Texteditor.UndoAvailable – benachrichtigt, wenn Undo verfügbar ist

BESCHREIBUNG

Dieser Tag wird auf `True` gesetzt, wenn der Benutzer seine Aktion(en) rückgängig machen kann. Sie können eine Benachrichtigung für diesen Tag erstellen und Ihre Undo-Schaltfläche deaktivieren, wenn es nichts rückgängig zu machen gibt.

TYP

Boolesch

ANWENDBARKEIT

GN

50 Textentry-Klasse (Texteingabe)

50.1 Übersicht

Die Textentry-Klasse (Texteingabe) leitet sich von der Area-Klasse ab und generiert die einzeiligen Texteingabe-Widgets, die zur Anzeige und Eingabe einer Text-Zeichenkette verwendet werden können.

Standardmäßig haben Texteingabe-Widgets keine Beschriftung (Label) neben sich. Wenn Sie eine Beschriftung neben Ihrem Texteingabe-Widget haben möchten, müssen Sie es in eine `<hgroup>` einfügen und dann die Label-Klasse verwenden, um eine Beschriftung daneben zu setzen.

Der Inhalt des XML-Tags wird als Anfangsinhalt des Texteingabe-Widgets verwendet.

Hier ist ein XML-Beispiel für ein Texteingabe-Widget:

```
<textentry id="mytextentry"/>
```

Beachten Sie, dass die Textentry-Klasse nur einzeilige Widgets unterstützt. Wenn Sie ein mehrzeiliges Texteingabe-Widget benötigen, müssen Sie stattdessen die Texteditor-Klasse verwenden. Siehe [Abschnitt 49.1 \[Texteditor-Klasse\]](#), [Seite 245](#), für Details.

50.2 Textentry.Accept

BEZEICHNUNG

`Textentry.Accept` – setzt die Zeichen, die vom Texteingabe-Widget akzeptiert werden

BESCHREIBUNG

Setzen Sie dies auf eine Zeichenkette, die Zeichen enthält, welche vom Widget akzeptiert werden. Nützlich zum Beispiel, wenn Sie nur die Eingabe von Zahlen erlauben wollen. In diesem Fall würden Sie `Textentry.Accept` auf `"0123456789"` setzen.

Sie können auch `Textentry.Reject` verwenden, um Zeichen selektiv abzulehnen.

TYP

Zeichenkette

ANWENDBARKEIT

I

50.3 Textentry.Acknowledge

BEZEICHNUNG

`Textentry.Acknowledge` – benachrichtigt, wenn der Benutzer RETURN drückt

BESCHREIBUNG

Immer wenn der Benutzer die Taste Return drückt, wird dieses Attribut auf `True` gesetzt. Sie können diese Benachrichtigung überwachen und die entsprechenden Maßnahmen ergreifen.

Das Drücken der TAB-Taste oder das Klicken mit der Maus zum Deaktivieren des Widgets löst `Textentry.Acknowledge` nicht aus.

TYP

Boolesch

ANWENDBARKEIT

N

50.4 Textentry.AdvanceOnCR**BEZEICHNUNG**

Textentry.AdvanceOnCR – aktiviert das nächste Objekt, wenn RETURN gedrückt wird

BESCHREIBUNG

Wenn Sie dies auf **True** setzen, verhält sich das Drücken von RETURN wie das Drücken der TAB-Taste, d.h. es gibt den Fokus auf das nächste Widget im Fenster.

TYP

Boolesch

ANWENDBARKEIT

I

50.5 Textentry.Copy**BEZEICHNUNG**

Textentry.Copy – kopiert den markierten Text

ÜBERSICHT

```
moai.DoMethod(id, "Copy")
```

BESCHREIBUNG

Kopiert den aktuell ausgewählten Text in die Zwischenablage.

Unter AmigaOS und kompatiblen Betriebssystemen erfordert diese Fähigkeit mindestens MUI 4.0.

EINGABEN

id ID des Textentry-Objekts

50.6 Textentry.CursorPos**BEZEICHNUNG**

Textentry.CursorPos – setzt/ermittelt die Cursorposition

BESCHREIBUNG

Setzt oder ermittelt die aktuelle Position des Cursor.

Unter AmigaOS und kompatiblen Betriebssystemen erfordert diese Fähigkeit mindestens MUI 4.0.

TYP

Zahl

ANWENDBARKEIT

SG

50.7 Textentry.Cut**BEZEICHNUNG**

Textentry.Cut – schneidet den markierten Text aus

ÜBERSICHT

```
moai.DoMethod(id, "Cut")
```

BESCHREIBUNG

Diese Methode schneidet den aktuell markierten Text aus und legt ihn in der Zwischenablage ab.

Unter AmigaOS und kompatiblen Betriebssystemen erfordert diese Fähigkeit mindestens MUI 4.0.

EINGABEN

`id` ID des Textentry-Objekts

50.8 Textentry.GetSelection**BEZEICHNUNG**

Textentry.GetSelection – ermittelt den Bereich der Textauswahl

ÜBERSICHT

```
start, end = moai.DoMethod(id, "GetSelection")
```

BESCHREIBUNG

Gibt den Bereich des aktuell ausgewählten Textes zurück, der bei `start` anfängt und bei `end` aufhört. Der Versatz vom gesamten Text beginnt bei 0, wobei mit 0 das erste Zeichen angegeben wird. Wenn kein Text markiert ist, wird für beide Werte -1 zurückgegeben.

EINGABEN

`id` ID des Textentry-Objekts

RÜCKGABEWERTE

`start` der Beginn des Bereichs

`end` das Ende des Bereichs

50.9 Textentry.Hint**BEZEICHNUNG**

Textentry.Hint – setzt/ermittelt den Texthinweis (V2.0)

BESCHREIBUNG

Legt den Hinweis eines Texteingabe-Widgets fest oder ermittelt ihn. Der Hinweis wird angezeigt, wenn kein Text im Widget vorhanden ist. Er kann verwendet werden, um dem Benutzer einen Hinweis darauf zu geben, was er in das Widget eingeben soll.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

50.10 Textentry.Insert**BEZEICHNUNG**

Textentry.Insert – fügt den Text ein

ÜBERSICHT

```
moai.DoMethod(id, "Insert", t$)
```

BESCHREIBUNG

Dadurch wird der in `t$` angegebener Text an der aktuellen Cursorposition eingefügt.

Unter AmigaOS und kompatiblen Betriebssystemen erfordert diese Fähigkeit mindestens MUI 4.0.

EINGABEN

<code>id</code>	ID des Textentry-Objekts
<code>t\$</code>	Text, der eingefügt wird

50.11 Textentry.Mark**BEZEICHNUNG**

Textentry.Mark – markiert den Text

ÜBERSICHT

```
moai.DoMethod(id, "Mark", start, end)
```

BESCHREIBUNG

Markiert den Text von der ersten Position `start` bis zum Zeichen an der letzten Position `end`. Das erste Zeichen vom gesamten Text steht an der Position 0.

Unter AmigaOS und kompatiblen Betriebssystemen erfordert diese Fähigkeit mindestens MUI 4.0.

EINGABEN

<code>id</code>	ID des Textentry-Objekts
<code>start</code>	Position des ersten zu markierenden Zeichens
<code>end</code>	Position des letzten zu markierenden Zeichens

50.12 Textentry.MarkAll

BEZEICHNUNG

Textentry.MarkAll – markiert den gesamten Text

ÜBERSICHT

```
moai.DoMethod(id, "MarkAll")
```

BESCHREIBUNG

Mit dieser Methode markieren Sie den gesamten Text im Widget.

Unter AmigaOS und kompatiblen Betriebssystemen erfordert diese Fähigkeit mindestens MUI 4.0.

EINGABEN

id ID des Textentry-Objekts

50.13 Textentry.MarkNone

BEZEICHNUNG

Textentry.MarkNone – entfernt die Markierung

ÜBERSICHT

```
moai.DoMethod(id, "MarkNone")
```

BESCHREIBUNG

Diese Methode entfernt jede Textauswahl. Der Text wird komplett unmarkiert, wenn diese Methode beendet ist.

Unter AmigaOS und kompatiblen Betriebssystemen erfordert diese Fähigkeit mindestens MUI 4.0.

EINGABEN

id ID des Textentry-Objekts

50.14 Textentry.MaxLen

BEZEICHNUNG

Textentry.MaxLen – stellt die maximale Eintragslänge ein

BESCHREIBUNG

Legt die maximale Anzahl von Zeichen fest, die eingegeben werden kann.

TYP

Zahl

ANWENDBARKEIT

IG

50.15 Textentry.Password

BEZEICHNUNG

Textentry.Password – setzt das Widget in den Passwortmodus

BESCHREIBUNG

Wenn Sie dieses Attribut setzen, wird die Benutzereingabe ausgeblendet. Nützlich vor allem bei der Eingabe von Passwörtern.

TYP

Boolesch

ANWENDBARKEIT

I

50.16 Textentry.Paste

BEZEICHNUNG

Textentry.Paste – fügt den Text aus der Zwischenablage ein

ÜBERSICHT

```
moai.DoMethod(id, "Paste")
```

BESCHREIBUNG

Fügt den Text aus der Zwischenablage in das Texteingabe-Widget ein.

Unter AmigaOS und kompatiblen Betriebssystemen erfordert diese Fähigkeit mindestens MUI 4.0.

EINGABEN

id ID des Textentry-Objekts

50.17 Textentry.ReadOnly

BEZEICHNUNG

Textentry.ReadOnly – deaktiviert die Textbearbeitung (V2.0)

BESCHREIBUNG

Setzen Sie diesen Tag auf `True`, um das Widget in den schreibgeschützten Modus zu versetzen.

Dies nützt wahrscheinlich nicht viel, da es die Text-Klasse zum Anzeigen von nicht bearbeitbarem Text gibt. Siehe [Abschnitt 48.1 \[Text-Klasse\]](#), Seite 243, für Details.

TYP

Boolesch

ANWENDBARKEIT

I

50.18 Textentry.Redo

BEZEICHNUNG

Textentry.Redo – hebt ein vorangehendes Undo auf

ÜBERSICHT

```
moai.DoMethod(id, "Redo")
```

BESCHREIBUNG

Diese Methode hebt ein vorangehendes Undo in einem Texteingabe-Widget auf.

Unter AmigaOS und kompatiblen Betriebssystemen erfordert diese Fähigkeit mindestens MUI 4.0.

EINGABEN

id ID des Textentry-Objekts

50.19 Textentry.Reject

BEZEICHNUNG

Textentry.Reject – setzt die Zeichen, welche vom Texteingabe-Widget abgelehnt werden

BESCHREIBUNG

Setzen Sie dieses Attribut auf eine Zeichenkette, die Zeichen enthält, die vom Widget abgelehnt werden sollen. Wenn Sie beispielsweise die Eingabe von Zahlen nicht zulassen möchten, setzen Sie `Textentry.Reject` auf "0123456789".

Sie können auch `Textentry.Accept` verwenden, um Zeichen selektiv zu akzeptieren.

TYP

Zeichenkette

ANWENDBARKEIT

I

50.20 Textentry.Text

BEZEICHNUNG

Textentry.Text – setzt/ermittelt den Inhalt vom Texteingabe-Widget

BESCHREIBUNG

Setzen oder ermitteln Sie den Inhalt eines Texteintrags-Widgets.

Sie können auch eine Benachrichtigung für dieses Attribut einrichten, um benachrichtigt zu werden, wenn sich der Inhalt des Widgets ändert. Beachten Sie, dass dies dazu führt, dass bei jedem Tastendruck ein Ereignis ausgelöst wird.

TYP

Zeichenkette

ANWENDBARKEIT

SGN

50.21 Textentry.Undo

BEZEICHNUNG

Textentry.Undo – Undo macht die letzte Operation rückgängig

ÜBERSICHT

```
moai.DoMethod(id, "Undo")
```

BESCHREIBUNG

Mit Undo nehmen Sie die letzte Operation des Texteingabe-Widgets zurück.

Unter AmigaOS und kompatiblen Betriebssystemen erfordert diese Fähigkeit mindestens MUI 4.0.

EINGABEN

id ID des Textentry-Objekts

51 Textview-Klasse (Textanzeige)

51.1 Übersicht

Die Textview-Klasse (Textanzeige) leitet sich von der Area-Klasse ab und kann verwendet werden, um größere Textmengen mit Unterstützung für Zeilenumbruch und Textformatierung in einem Widget anzuzeigen. Der Inhalt des XML-Tags wird als Anfangsinhalt des Textanzeige-Widgets verwendet.

Hier ist ein Beispiel für die Verwendung des XML-Tags `<textview>`:

```
<textview>Hello World</textview>
```

Hier ist das gleiche Beispiel in fett ausgegeben:

```
<textview styled="true">\33bHello World</textview>
```

Die hier angegebene Zeichenkette kann Textformatierungscodes verwenden, wenn `Textview.Styled` auf `True` gesetzt wurde. Siehe [Abschnitt 3.14 \[Textformatierungscodes\]](#), [Seite 24](#), für Details.

51.2 Textview.Align

BEZEICHNUNG

`Textview.Align` – setzt die Textausrichtung oder gibt sie zurück

BESCHREIBUNG

Legen Sie die gewünschte Textausrichtung für mehrzeiligen Text fest. Dies kann einer der folgenden Werte sein:

<code>Left</code>	Links ausgerichtet.
<code>Right</code>	Rechts ausgerichtet.
<code>Center</code>	Zentriert ausgerichtet.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

ISG

51.3 Textview.Append

BEZEICHNUNG

`Textview.Append` – hängt den Text an das Ende der Textanzeige (V2.0)

ÜBERSICHT

```
moai.DoMethod(id, "Append", s$)
```

BESCHREIBUNG

Hängt den Text in der durch `s$` angegebenen Zeichenkette an das Ende der Textanzeige an.

Unter AmigaOS und kompatiblen Systemen erfordert diese Methode MUI 4.0 oder besser.

EINGABEN

<code>id</code>	ID des Textview-Objekts
<code>s\$</code>	Text, der angehängt wird

51.4 Textview.Styled

BEZEICHNUNG

`Textview.Styled` – aktiviert die Textformatierung

BESCHREIBUNG

Setzen Sie dieses Attribut auf `True`, um die Textformatierung für dieses Widget zu aktivieren. Wenn auf `True` gesetzt, können Sie spezielle Textformatierungs-codes in dem Text verwenden, die Sie an dieses Widget übergeben. Siehe [Abschnitt 3.14 \[Textformatierungs-codes\]](#), Seite 24, für Details.

TYP

Boolesch

ANWENDBARKEIT

I

51.5 Textview.Text

BEZEICHNUNG

`Textview.Text` – setzt den Inhalt des Textview-Objekts oder gibt ihn zurück

BESCHREIBUNG

Hiermit wird der Text gesetzt, der vom Textanzeige-Widget angezeigt werden soll. Aber Sie können auch den Text aus dem Textanzeige-Widget auslesen. Der Text kann Zeilenumbrüche enthalten und wird automatisch umgebrochen, sobald er die Grenzen des Widgets erreicht. Mehrzeiliger Text wird entsprechend der mit `Textview.Align` eingestellten Ausrichtung ausgegeben.

Die hier angegebene Zeichenkette kann Textformatierungs-codes verwenden, wenn `Textview.Styled` auf `True` gesetzt wurde.

TYP

Zeichenkette

ANWENDBARKEIT

SG

52 Toolbar-Klasse (Symbolleiste)

52.1 Übersicht

Die Toolbar-Klasse (Symbolleiste) leitet sich von der Area-Klasse ab und erstellt eine Leiste mit Bildschaltflächen, die normalerweise am oberen Rand eines Fensters platziert wird. Die Schaltflächen können in verschiedenen Ansichtsmodi angezeigt werden: Entweder als Bild, Bild und Text oder nur als Text. Normalerweise werden Schaltflächen in der Symbolleiste nur als Bilder angezeigt.

Wenn Sie ein Symbolleisten-Widget im XML-Code definieren, müssen Sie immer mindestens eine Symbolleistenschaltfläche hinzufügen. Dies geschieht mit der `Toolbarbutton`-Klasse (Symbolleistenschaltflächen). Hier ist eine Beispiel-XML-Definition einer Symbolleiste mit sechs Schaltflächen:

```
<toolbar>
  <button icon="1">Open</button>
  <button icon="2">Save</button>
  <button/>
  <button icon="3">Cut</button>
  <button icon="4">Copy</button>
  <button icon="5">Paste</button>
  <button/>
  <button icon="6">Help</button>
</toolbar>
```

In der obigen XML-Definition verwendet die Schaltfläche 1 in der Symbolleiste den Hollywood-Pinsel 1 als Bild, Schaltfläche 2 den Pinsel 2 und so weiter. Beachten Sie die leeren `<button/>`-Definitionen: Diese erzeugen Füll-Elemente, um eine zusammengehörende Gruppe von Schaltflächen, optisch vom Rest der Schaltflächen zu trennen. Symbolleisten-Schaltflächen können viele weitere Optionen wie spezielle Bilder für ausgewählte und deaktivierte Zustände, Tooltips und mehr verwenden. Siehe [Abschnitt 53.1 \[Toolbarbutton-Klasse\], Seite 271](#), für Details.

Es kann nur eine Symbolleiste pro Fenster geben und sie muss immer das erste Element der Wurzelgruppe des Fensters sein. RapaGUI akzeptiert keine Symbolleisten an beliebigen Positionen im GUI-Layout. Sie befinden sich also entweder oben (horizontale Symbolleisten) oder links vom Fenster (vertikale Symbolleisten).

Beachten Sie, dass es nicht möglich ist, eigenständige Instanzen dieser Klasse mit `moai.CreateObject()` zu erzeugen. Symbolleisten müssen immer im Kontext eines Fensters erstellt werden. Wenn Sie also Symbolleisten mit `moai.CreateObject()` erzeugen wollen, müssen Sie immer ein komplettes Fenster erstellen und die Symbolleiste in diese Fensterdefinition einbetten.

AmigaOS-Benutzer beachten bitte auch, dass diese Klasse die Erweiterung `TheBar.mcc` benötigt.

52.2 Toolbar.Horiz

BEZEICHNUNG

Toolbar.Horiz – setzt die Ausrichtung der Symbolleiste

BESCHREIBUNG

Boolesch-Wert, um anzugeben, ob die Schaltflächen der Symbolleiste horizontal (**True**) oder vertikal (**False**) angeordnet werden sollen. Standardwert ist **True**.

TYP

Boolesch

ANWENDBARKEIT

I

52.3 Toolbar.ViewMode

BEZEICHNUNG

Toolbar.ViewMode – stellt den Ansichtsmodus der Symbolleiste ein

BESCHREIBUNG

Legt den Ansichtsmodus der Symbolleiste fest. Die folgenden Modi werden unterstützt:

TextGfx Schaltflächen in der Symbolleiste werden als Text und Bilder angezeigt.

Gfx Schaltflächen in der Symbolleiste werden nur als Bilder angezeigt. Dies ist auch die Voreinstellung.

Text Schaltflächen in der Symbolleiste werden nur als Text angezeigt.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

I

53 Toolbarbutton-Klasse (Symbolleistenschaltfläche)

53.1 Übersicht

Die Toolbarbutton-Klasse (Symbolleistenschaltflächen) ist eine Unterklasse der Toolbar-Klasse (Symbolleiste). Sie kann nicht allein verwendet werden, sondern muss immer innerhalb einer Toolbar-Klassendefinition gekapselt werden. Toolbarbutton-Klasse erstellt eine einzelne Schaltfläche für seine Toolbar-Klasse.

Bitte beachten Sie, dass das XML-Tag für diese Klasse nur `<button>` ist, nicht `<toolbarbutton>`. Siehe [Abschnitt 52.1 \[Toolbar-Klasse\]](#), Seite 269, für ein Beispiel.

53.2 Toolbarbutton.Disabled

BEZEICHNUNG

Toolbarbutton.Disabled – aktiviert/deaktiviert die Schaltfläche

BESCHREIBUNG

Aktiviert oder deaktiviert eine Symbolleisten-Schaltfläche.

TYP

Boolesch

ANWENDBARKEIT

ISG

53.3 Toolbarbutton.Help

BEZEICHNUNG

Toolbarbutton.Help – setzt den Hilfetext für die Symbolleisten-Schaltfläche

BESCHREIBUNG

Legen Sie den Hilfetext der Symbolleisten-Schaltfläche fest. Dieser Text wird automatisch in der Statusleiste des Fensters angezeigt, in dem sich die Symbolleiste befindet, wenn die Maus über die Schaltfläche in der Symbolleiste bewegt wird. Dies ist für den Anwender sehr komfortabel, da es die Funktion der einzelnen Schaltflächen der Symbolleiste auf elegante Weise erklärt. Siehe [Abschnitt 46.1 \[Statusbar-Klasse \(Statusleiste\)\]](#), Seite 239, für Details.

Beachten Sie, dass dies nicht dasselbe wie `Toolbarbutton.ToolTip` ist. Kurzinfos (Tool-tips) werden als kleine Fenster direkt neben der Symbolleisten-Schaltfläche angezeigt, sobald Sie mit der Maus darüber fahren. Der Hilfetext wird dagegen sofort in der Statusleiste des Fensters angezeigt, sobald sich der Mauszeiger über einer Schaltfläche in der Symbolleiste befindet.

TYP

Zeichenkette

ANWENDBARKEIT

I

53.4 `Toolbarbutton.Icon`

BEZEICHNUNG

`Toolbarbutton.Icon` – setzt das Bild für die Symbolleisten-Schaltfläche

BESCHREIBUNG

Setzen Sie dieses Attribut auf den Identifikator eines Hollywood-Pinsels oder -Icons, um ein Bild zu Ihrer Symbolleisten-Schaltfläche hinzuzufügen. Ob dieses Attribut einen Hollywood-Pinsel oder ein Hollywood-Icon erwartet, hängt davon ab, was Sie im Attribut `Toolbarbutton.IconType` angeben. Standardmäßig erwartet `Toolbarbutton.Icon` einen Hollywood-Pinsel.

Beachten Sie, dass RapaGUI das Bild möglicherweise skaliert, um es an die DPI-Einstellung des aktuellen Monitors anzupassen. Bitte lesen Sie das Kapitel über High-DPI-Unterstützung für weitere Informationen. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), [Seite 21](#), für Details.

Bitte lesen Sie auch das Kapitel Bilder-Cache von RapaGUI, um mehr über die Unterstützung von Icons in RapaGUI zu erfahren. Siehe [Abschnitt 3.20 \[Bilder-Cache\]](#), [Seite 32](#), für Details.

TYP

Zahl

ANWENDBARKEIT

I

53.5 `Toolbarbutton.IconScale`

BEZEICHNUNG

`Toolbarbutton.IconScale` – konfiguriert die automatische Bildskalierung (V2.0)

BESCHREIBUNG

Wenn `Toolbarbutton.Icon` auf einen Rasterpinsel gesetzt wurde und RapaGUI auf einem High-DPI-Display ausgeführt wird, skaliert RapaGUI automatisch die Rastergrafiken des Pinsels, um sie an die DPI-Einstellung des aktuellen Monitors anzupassen. Wenn Sie dies nicht möchten, setzen Sie diesen Tag auf `False`.

Alternativ können Sie die automatische Bildskalierung auch global deaktivieren, indem Sie beim Aufrufen vom Befehl `@REQUIRE` mit RapaGUI den Tag `ScaleGUI` auf `False` setzen. Siehe [Abschnitt 3.4 \[Initialisierung von RapaGUI\]](#), [Seite 11](#), für Details.

Bitte lesen Sie auch das Kapitel über High-DPI-Unterstützung in RapaGUI, um mehr über die Unterstützung von High-DPI-Displays zu erfahren. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), [Seite 21](#), für Details.

TYP

Boolesch

ANWENDBARKEIT

I

53.6 Toolbarbutton.IconType

BEZEICHNUNG

Toolbarbutton.IconType – legt den zu verwendenden Icon-Typ fest (V2.0)

BESCHREIBUNG

Mit diesem Attribut können Sie den Typ des Hollywood-Bildobjekts festlegen, das im Attribut `Toolbarbutton.Icon` übergeben wird. Standardmäßig erwartet `Toolbarbutton.Icon` einen Hollywood-Pinsel. Durch Setzen von `Toolbarbutton.IconType` können Sie jedoch einen anderen Hollywood-Bildtyp verwenden.

Die folgenden Bildtypen sind derzeit verfügbar:

Brush Verwendet einen Hollywood-Pinsel. Dies ist der Standardtyp. Sie können entweder Raster- oder Vektorpinsel verwenden. Vektorpinsel haben den Vorteil, dass sie ohne Qualitätsverlust auf jede beliebige Auflösung skaliert werden können. Dies ist sehr nützlich, wenn Sie Programme entwerfen, die mit High-DPI-Monitoren kompatibel sein sollen. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), [Seite 21](#), für Details.

Icon Verwendet ein Hollywood-Icon. Dieser Bildtyp hat den Vorteil, dass er mehrere Teilbilder unterschiedlicher Größe enthalten kann. Dadurch ist es möglich, Bilder in unterschiedlichen Auflösungen bereitzustellen, was bei der Entwicklung von Programmen, die mit High-DPI-Monitoren kompatibel sein sollen, sehr nützlich sein kann. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), [Seite 21](#), für Details.

Beachten Sie, dass Sie den Standardwert dieses Attributs global auf Hollywood-Icon ändern können, indem Sie den Tag `Application.UseIcons` setzen. Siehe [Abschnitt 9.13 \[Application.UseIcons\]](#), [Seite 70](#), für Details.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

I

53.7 Toolbarbutton.Pressed

BEZEICHNUNG

Toolbarbutton.Pressed – benachrichtigt, wenn eine Schaltfläche gedrückt wird

BESCHREIBUNG

Dieses Attribut wird ausgelöst, wenn der Benutzer eine Schaltfläche drückt. RapaGUI überwacht automatisch dieses Attribut für alle Schaltflächen der Symbolleiste, so dass Sie nicht explizit eine Benachrichtigung mit dem Attribut `MOAI.Notify` anfordern müssen.

TYP

Boolesch

ANWENDBARKEIT

N

53.8 Toolbarbutton.Selected**BEZEICHNUNG**

Toolbarbutton.Selected – Auswahlzustand umschalten/benachrichtigen

BESCHREIBUNG

Hiermit können Sie den Auswahlzustand einer Umschalt- sowie Radio-Schaltfläche umschalten oder den aktuellen Zustand überwachen.

RapaGUI überwacht automatisch dieses Attribut für alle Schaltflächen der Symbolleiste, so dass Sie nicht explizit eine Benachrichtigung mit dem Attribut MOAI.Notify anfordern müssen.

TYP

Boolesch

ANWENDBARKEIT

ISGN

53.9 Toolbarbutton.Tooltip**BEZEICHNUNG**

Toolbarbutton.Tooltip – setzt die Kurzinfo für Symbolleisten-Schaltfläche

BESCHREIBUNG

Legt die Kurzinfo der Symbolleisten-Schaltfläche fest.

Beachten Sie, dass dies nicht dasselbe wie `Toolbarbutton.Help` ist. Kurzinfos werden als kleine Fenster direkt neben der Symbolleisten-Schaltfläche angezeigt, wenn Sie mit der Maus darüber fahren. Der Hilfetext wird dagegen sofort in der Statusleiste des Fensters angezeigt, sobald sich der Mauszeiger über einer Schaltfläche in der Symbolleiste befindet.

TYP

Zeichenkette

ANWENDBARKEIT

I

53.10 Toolbarbutton.Type**BEZEICHNUNG**

Toolbarbutton.Type – legt den Typ der zu erstellenden Schaltfläche fest

BESCHREIBUNG

Dieses Attribut definiert die Art der Schaltfläche in der Symbolleiste, die Sie haben möchten. Folgende Typen sind derzeit möglich:

`Normal` Normale Schaltfläche in der Symbolleiste.

- Toggle** Schaltfläche in der Symbolleiste, die zwischen zwei Zuständen umgeschaltet werden kann.
- Radio** Schaltfläche in der Symbolleiste, die mit den benachbarten Schaltflächen eine Gruppe bildet. Es kann jeweils nur eine Schaltfläche der Gruppe ausgewählt werden.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

I

54 Treeview-Klasse (Baumansicht)

54.1 Übersicht

Die Treeview-Klasse (Baumansicht) leitet sich von der Area-Klasse ab und kann verwendet werden, um komplexe Datenstrukturen mithilfe einer Baumstruktur zu visualisieren. Die Daten werden als Hierarchie dargestellt, in der jedes Element eine unendliche Anzahl von Unterelementen enthalten kann. Elemente, die Unterelemente enthalten, werden Knoten genannt, während Elemente ohne Unterelemente Blätter heißen. RapaGUIs Treeview-Klasse ist sehr mächtig und unterstützt mehrspaltige Bäume, Auswahlkästchen (Checkboxes), bearbeitbare Knoten und Blätter sowie Symbole für die einzelnen Baumansicht-Elemente.

Beim Erstellen einer Baumansicht im XML-Code müssen Sie immer mindestens eine Spalte hinzufügen. Dies geschieht mit Treeviewcolumn-Klasse (Baumansichtspalten). Hier ist ein Beispiel für eine minimale Baumansicht-Definition mit nur einer einzigen Spalte:

```
<treeview>
  <column/>
</treeview>
```

Es ist auch möglich, einige Einträge zum Zeitpunkt der XML-Definition in die Baumansicht einzufügen. Dies kann mit den Tags `<node>` und `<leaf>` geschehen, die sich auf Treeviewnode-Klasse (Baumansichtknoten) bzw. Treeviewleaf-Klasse (Baumansichtblätter) beziehen. Ein Knoten der Baumansicht enthält immer nur einen einzigen Eintrag, auch bei mehrspaltigen Baumansichten. Blätter hingegen müssen jedoch so viele Einträge definieren, wie es Spalten in der Baumansicht gibt. Sie müssen also ein `<item>`-Tag pro Spalte innerhalb des `<leaf>` verwenden, um diese einzelnen Elemente zu erstellen. Hier ist eine Beispieldefinition:

```
<treeview>
  <column/>
  <node name="CPU">
    <leaf><item>Model: Motorola MPC 7447 Apollo V1.1</item></leaf>
    <leaf><item>CPU speed: 999 Mhz</item></leaf>
    <leaf><item>FSB speed: 133 Mhz</item></leaf>
    <leaf><item>Extensions: performancemonitor altivec</item></leaf>
  </node>
  <node name="Machine">
    <leaf><item>Machine name: Pegasos II</item></leaf>
    <leaf><item>Memory: 524288 KB</item></leaf>
    <leaf><item>Extensions: bus.pci bus.agp</item></leaf>
  </node>
  <node name="Expansion buses">
    <node name="PCI/AGP">
      <leaf><item>Vendor 0x11AB Device 0x6460</item></leaf>
    </node>
  </node>
  <node name="Libraries">
    <leaf><item>0x6c7d4a58: exec.library V53.34</item></leaf>
```

```

</node>
<node name="Devices">
  <leaf><item>0x6ff8fba4: ramdrive.device V52.6</item></leaf>
</node>
<node name="Tasks">
  <node name="input.device">
    <leaf><item>Stack: 0x6ff4b000 - 0x6ff5b000</item></leaf>
    <leaf><item>Signals: SigWait 0x00000000</item></leaf>
    <leaf><item>State: Task (Waiting)</item></leaf>
  </node>
</node>
</treeview>

```

Wie Sie sehen können, haben wir eine einspaltige Baumansicht mit dem obigen XML-Code erstellt. So müssen wir `<item>` nur einmal pro `<leaf>`-Definition verwenden. Für mehrspaltige Bäume müssten Sie so viele `<item>`-Tags verwenden, wie es Spalten in Ihrer Baumansicht gibt. Siehe [Abschnitt 56.1 \[Treeviewleaf-Klasse\], Seite 299](#), für Details. Siehe [Abschnitt 57.1 \[Treeviewleafitem-Klasse\], Seite 305](#), für Details.

Im Beispiel haben wir auch das Attribut `Treeviewnode.Name` verwendet, um jedem unserer Knoten einen Namen hinzuzufügen. Es gibt noch einige weitere Attribute, mit denen Sie das Aussehen Ihrer Knoten anpassen können. Siehe [Abschnitt 58.1 \[Treeviewnode-Klasse\], Seite 307](#), für Details. Um beispielsweise auf ein Element (Knoten oder Blatt) in einem Treeview-Objekt zu verweisen, müssen Sie ihm eine eindeutige ID-Zeichenfolge zuweisen. Sie können dann auf dieses Element verweisen, indem Sie einfach diesen ID-Zeichenfolge verwenden. IDs werden entweder bei der Objekterstellung im XML-Code oder beim Aufruf von `Treeview.InsertNode` oder `Treeview.InsertLeaf` vergeben.

Im obigen Beispiel definieren wir keine Attribute für die Spalte der Baumansicht. Es ist jedoch möglich, das Aussehen von Baumspalten anzupassen. Sie können beispielsweise das Attribut `Treeviewcolumn.Title` verwenden, um jeder Ihrer Spalten eine Titelleiste hinzuzufügen. Darüber hinaus können Sie Ihren Spalten auch Auswahlkästchen (Checkboxes) hinzufügen und die Bearbeitung von Spaltenelementen erlauben. Siehe [Abschnitt 55.1 \[Treeviewcolumn-Klasse\], Seite 293](#), für Details.

Beachten Sie, dass RapaGUI möglicherweise zwei verschiedene Arten von Widgets für diese Klasse verwendet: Wenn Sie eine Strukturansicht erstellen, die keine erweiterten Funktionen (z.B. mehrere Spalten, Kontrollkästchen, bearbeitbare Elemente, Dekorationen) verwendet, erstellt RapaGUI eventuell ein einfacheres Baumansichts-Widget für Sie, weil einige Betriebssysteme zwei verschiedene Baumansicht-basierte Widgets anbieten. RapaGUI wird das Light-Widget verwenden, falls Ihre Baumansicht keine der erweiterten Funktionen verwendet, da es schneller ist. Wenn Sie das nicht möchten, können Sie RapaGUI zwingen, immer eine vollständige Baumansicht zu verwenden, indem Sie das Attribut `Treeview.ForceMode` auf das entsprechende Tag setzen. Siehe [Abschnitt 54.12 \[Treeview.ForceMode\], Seite 283](#), für Details.

54.2 Treeview.AbortEditing

BEZEICHNUNG

Treeview.AbortEditing – benachrichtigt, wenn der Benutzer die Bearbeitung abbricht (V1.1)

BESCHREIBUNG

Wenn Sie eine Benachrichtigung für dieses Attribut einrichten, führt RapaGUI Ihre Callback-Funktion aus, wenn der Benutzer einen Bearbeitungsvorgang an einem Element abbricht, z.B. durch Drücken der Escape-Taste oder durch Klicken außerhalb des Elementbearbeitungs-Widgets.

Beachten Sie, dass Sie `Treeviewcolumn.Editable` auf `True` setzen müssen, wenn Blätter in der jeweiligen Spalte editierbar sein sollen. Um Knoten editierbar zu machen, müssen Sie `Treeview.EditableNodes` auf `True` setzen.

Ihre Callback-Funktion wird mit den folgenden zusätzlichen Argumenten aufgerufen:

Item: ID des Knotens oder Blattes, den der Benutzer bearbeitet hat, als er den Vorgang abgebrochen hat.

Column: Spaltenindex des Elements, welches der Benutzer bearbeitet hat, als er die Bearbeitung abgebrochen hat. Für Knoten ist dies immer 0.

Siehe [Abschnitt 3.7 \[Benachrichtigungen der Attribute\]](#), Seite 14, für Details.

TYP

Boolesch

ANWENDBARKEIT

N

54.3 Treeview.Active

BEZEICHNUNG

Treeview.Active – setzt/ermittelt das aktive Bauelement

BESCHREIBUNG

Setzen Sie dieses Attribut, um das angegebene Bauelement zu aktivieren. An dieses Attribut muss die ID des zu aktivierenden Eintrags übergeben werden. IDs werden entweder bei der Objekterstellung im XML-Code oder beim Aufruf von `Treeview.InsertNode` oder `Treeview.InsertLeaf` vergeben.

Um die Markierung des aktiven Eintrags aufzuheben, übergeben Sie einfach den speziellen Wert `Off`. Wenn dieses Attribut abgerufen wird, gibt es die ID des aktiven Bauelements und `Off` zurück, falls es keinen aktiven Eintrag gibt.

Sie können auch eine Benachrichtigung für `Treeview.Active` einrichten, um über Änderungen der Auswahl informiert zu werden. Die ID des aktiven Bauelements wird dann in `TriggerValue` zurückgegeben.

TYP

Zeichenkette

ANWENDBARKEIT

SGN

54.4 Treeview.Alternate**BEZEICHNUNG**

Treeview.Alternate – verwendet wechselnde Zeilenfarben

BESCHREIBUNG

Setzen Sie dieses Attribut auf `True`, damit die Baumansicht mit wechselnden Zeilenfarben erscheint.

Beachten Sie, dass dieses Attribut unter AmigaOS und kompatiblen Systemen mindestens MUI 5.0 erfordert.

TYP

Boolesch

ANWENDBARKEIT

I

54.5 Treeview.Clear**BEZEICHNUNG**

Treeview.Clear – löscht die Baumansicht (V2.0)

ÜBERSICHT

```
moai.DoMethod(id, "Clear")
```

BESCHREIBUNG

Entfernt alle Einträge aus der Baumansicht.

EINGABEN

<code>id</code>	ID des Baumansicht-Objekts
-----------------	----------------------------

54.6 Treeview.ClickColumn**BEZEICHNUNG**

Treeview.ClickColumn – benachrichtigt bei Spaltenklicks (V2.0)

BESCHREIBUNG

In Baumansichten mit mehreren Spalten zeichnet dieses Attribut die Nummer der Spalte auf, auf die der Benutzer zuletzt geklickt hat.

TYP

Zahl

ANWENDBARKEIT

GN

54.7 Treeview.Close

BEZEICHNUNG

Treeview.Close – schließt den Knoten in der Baumansicht

ÜBERSICHT

```
moai.DoMethod(id, "Close", node$)
```

BESCHREIBUNG

Diese Methode schliesst den angegebenen Knoten und alle seine Unterknoten in der Baumansicht.

Sie können die folgenden speziellen Werte für `node$` übergeben:

Root Der Wurzelknoten. Die Übergabe dieses speziellen Wertes schließt alle Knoten.

Active Der aktive Knoten.

EINGABEN

id ID des Treeview-Objekts

node\$ ID des zu verwendenden Baumknotens oder spezieller Wert (siehe oben)

54.8 Treeview.DoubleClick

BEZEICHNUNG

Treeview.DoubleClick – benachrichtigt, wenn Doppelklicks auf Baum-Elemente ausgeführt werden

BESCHREIBUNG

Sie können eine Benachrichtigung für dieses Attribut einrichten, um bei Doppelklicks auf Baum-Elemente zu reagieren. Das Baum-Element, auf das der Benutzer doppelt geklickt hat, wird in `TriggerValue` übergeben.

TYP

Boolesch

ANWENDBARKEIT

N

54.9 Treeview.DropFile

BEZEICHNUNG

Treeview.DropFile – benachrichtigt, wenn Datei(en) über dem Widget abgelegt werden (V1.1)

BESCHREIBUNG

Wenn Sie eine Benachrichtigung für dieses Attribut einrichten, führt RapaGUI Ihre Callback-Funktion aus, sobald eine oder mehrere Dateien auf das Baumansicht-Widget gezogen wurden. Das Feld `TriggerValue` der Nachrichtentabelle wird auf eine Tabelle gesetzt, die eine Liste aller Dateien enthält, die auf das Widget gezogen wurden.

Außerdem enthält die Meldungstabelle die folgenden zwei zusätzlichen Felder:

- X:** Die x-Position, an der der Benutzer die Datei(en) abgelegt hat. Dies ist relativ zur linken Ecke des Widgets.
- Y:** Die y-Position, an der der Benutzer die Datei(en) abgelegt hat. Dies ist relativ zur oberen Ecke des Widgets.

Siehe [Abschnitt 3.7 \[Benachrichtigungen der Attribute\]](#), Seite 14, für Details.

Beachten Sie, dass `Treeview.DropFile` nur dann ausgelöst wird, wenn `Treeview.DropTarget` zuerst auf `True` gesetzt wurde.

TYP

Boolesch

ANWENDBARKEIT

N

54.10 Treeview.DropTarget

BEZEICHNUNG

`Treeview.DropTarget` – konfiguriert, ob Dateien abgelegt werden können (V1.1)

BESCHREIBUNG

Setzen Sie dies auf `True`, wenn Dateien auf diesem Baumansicht-Widget abgelegt werden können. Sie können das Attribut `Treeview.DropFile` überwachen, um zu erfahren, wenn der Benutzer eine oder mehrere Dateien darauf ablegt.

TYP

Boolesch

ANWENDBARKEIT

ISG

54.11 Treeview.EditableNodes

BEZEICHNUNG

`Treeview.EditableNodes` – erlaubt die Bearbeitung von Baum-Knoten

BESCHREIBUNG

Setzen Sie dies auf `True`, um die Bearbeitung von Baum-Knoten durch den Benutzer zu ermöglichen. Der Benutzer kann dann alle Knoten im Baum bearbeiten, indem er einen langsamen Doppelklick ausführt, d.h. die linke Maustaste zweimal hintereinander langsam drückt.

Wenn Sie nur die Bearbeitung einiger Knoten im Baum erlauben möchten, müssen Sie dieses Attribut auf `True` setzen und das Attribut `Treeview.StartEditing` überwachen. `Treeview.StartEditing` wird immer dann ausgelöst, wenn der Benutzer versucht, einen Knoten zu bearbeiten und Ihr Callback-Funktion kann `False` zurückgeben, um die Bearbeitung bestimmter Elemente zu verbieten. Siehe [Abschnitt 54.19 \[Treeview.StartEditing\]](#), Seite 289, für Details.

Um benachrichtigt zu werden, wenn sich der Wert eines Baum-Knotens ändert, weil der Benutzer ihn bearbeitet hat, müssen Sie das Attribut `TreeView.ValueChange` überwachen.

Um die Bearbeitung eines Baum-Knotens manuell zu starten, rufen Sie die Methode `TreeViewnode.Edit` auf.

Beachten Sie, dass dieses Attribut nur für Baum-Knoten gilt. Wenn Sie möchten, dass auch Beschriftungen der Blätter bearbeitet werden können, müssen Sie das Attribut `TreeViewcolumn.Editable` auf `True` setzen.

Beachten Sie auch, dass unter AmigaOS und kompatiblen Betriebssystemen diese Fähigkeit mindestens MUI 4.0 erfordert.

TYP

Boolesch

ANWENDBARKEIT

I

54.12 Treeview.ForceMode**BEZEICHNUNG**

`TreeView.ForceMode` – überschiebt den Standardmodus der Baumansicht

BESCHREIBUNG

RapaGUI kann abhängig von Ihren Einstellungen zwei verschiedene Widgets für die Treeview-Klasse verwenden. Wenn beispielsweise eine einspaltige Baumansicht ohne Überschriften verwendet wird, benutzt RapaGUI möglicherweise aus Gründen der Effizienz ein anderes Widget, falls das Host-Betriebssystem ein solches Widget bereitstellt. Unter Windows beispielsweise wird RapaGUI in diesen Fällen das Treeview-Control anstelle einer voll funktionsfähigen Baumansicht verwenden. Wenn Sie das nicht möchten, setzen Sie dieses Attribut auf das gewünschte Widget und RapaGUI wird versuchen, es zu verwenden.

Die folgenden Modi werden derzeit erkannt:

- Normal** Wählt automatisch das Widget aus, das am besten passt. Dies ist die Voreinstellung.
- TreeView** Verwendet ein Treeview-Widget. Treeview-Widgets unterstützen nur einspaltige Bäume ohne editierbare Elemente und ohne Auswahlkästchen (Checkboxen), ohne Titel und ohne spezielle Dekorationen.
- Dataview** Verwenden Sie ein Dataview-Widget. Unterstützt alles, verwendet aber derzeit eine generische Implementierung unter Windows.

Beachten Sie, dass dieses Attribut unter AmigaOS und Kompatiblen keine Auswirkung hat, da RapaGUI auf diesen Plattformen immer das gleiche Widget verwendet.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

I

54.13 Treeview.GetEntry**BEZEICHNUNG**

Treeview.GetEntry – ermittelt Informationen über den Baumeintrag

ÜBERSICHT

```
found, table = moai.DoMethod(id, "GetEntry", item, position)
```

BESCHREIBUNG

Ermittelt Informationen über einen Baumeintrag. Der Baumeintrag kann entweder als absoluter Index oder auch relativ zu einem bestimmten Knoten oder Blatt angegeben werden. So können Sie den gesamten Baum durchqueren. Siehe unten für ein Beispiel.

`item` gibt das Bauelement an, das als Referenzpunkt verwendet werden soll. Dies kann ein Knoten oder ein Blatt sein. Beachten Sie, dass `item` im Gegensatz zu allen anderen Methoden oder Attributen der Treeview-Klasse kein Zeichenketten-Identifikator, sondern ein spezieller Wert sein muss, der von dieser Methode im Feld `UID` (siehe unten) zurückgegeben oder durch Abfragen von `Treeviewleaf.UID` oder `Treeviewnode.UID` für ein Objekt abgerufen wird. Alternativ kann es einer der folgenden speziellen Werte sein:

Root Verwendet den Wurzelknoten.
Active Verwendet das aktive Element.

`position` kann entweder eine Zahl oder ein spezieller Wert sein. Die Übergabe einer Nummer wird nur unterstützt, wenn das angegebene `item` ein Knoten ist. In diesem Fall gibt die Zahl den Index des Elementes an, über das Sie Informationen erhalten möchten, beginnend bei 0, d.h. die Übergabe von 5 würde hier Informationen über das sechste Element des in `item` übergebenen Knotens zurückgeben. Alternativ können Sie die folgenden Sonderwerte in `position` übergeben:

Head Gibt Informationen über das erste Element des Knotens zurück.
Tail Gibt Informationen über das letzte Element des Knotens zurück.
Active Gibt Informationen über das aktive Element zurück.
Next Gibt den nächsten Eintrag im Baum nach `item` zurück.
Previous Gibt den vorherigen Eintrag im Baum vor `item` zurück.
Parent Gibt Informationen über den übergeordneten Objektteil von `item` zurück.

Diese Methode gibt zwei Werte zurück: Der erste Rückgabewert `found` ist ein boolesches Flag, das angibt, ob ein Element gefunden wurde oder nicht. Wenn der erste Rückgabewert `True` ist, ist der zweite Rückgabewert `table` eine Tabelle, bei dem folgende Felder initialisiert sind:

Items Dies ist eine Tabelle, die die Einträge für alle Spalten dieses Baum-Eintrags enthält. Beachten Sie, dass wenn der Eintrag ein Knoten ist, diese Tabelle nur einen Eintrag enthält, da sich Knoten nicht über mehrere Spalten erstrecken können.

Node	True , wenn der gefundene Eintrag ein Knoten ist, False , wenn es ein Blatt ist.
ID	Zeichenkette-Objekt-Identifikator dieses Baumeintrags.
UID	Interne ID dieses Baumeintrags. Dies ist die einzige ID, die Sie im Argument node dieser Methode übergeben dürfen. Die Übergabe von Standard-Zeichenketten-Objekt-Identifikatoren ist bei dieser Methode nicht erlaubt. Sie können diesen Wert für nachfolgende Aufrufe von Treeview.GetEntry im Argument node verwenden. Siehe oben für weitere Informationen und unten für ein Beispiel. Sie können UIDs auch erhalten, indem Sie das Attribut Treeviewleaf.UID oder Treeviewnode.UID ermitteln.

EINGABEN

id	ID des Treeview-Objekts
item	spezieller Identifikator, der von dieser Methode zurückgegeben wird oder ein spezieller Wert (siehe oben)
position	Index des Eintrags, der ermittelt wird oder ein spezieller Wert (siehe oben)

RÜCKGABEWERTE

found	boolesches Flag, das angibt, ob ein Eintrag gefunden wurde oder nicht
table	Tabelle mit Informationen zum gefundenen Eintrag

BEISPIEL

```
Function p_DumpListTree(id$, node, indent)
  Local found, t = moai.DoMethod(id$, "GetEntry", node, "Head")
  While found = True
    If indent > 0
      DebugPrint(RepeatStr(" ", indent) ..
        IIf(t.Node = True, "+", ""), Unpack(t.items))
    Else
      DebugPrint(IIf(t.Node = True, "+", ""), Unpack(t.items))
    EndIf
    If t.Node = True Then p_DumpListTree(id$, t.uid, indent + 4)
    found, t = moai.DoMethod(id$, "GetEntry", t.uid, "Next")
  Wend
EndFunction
```

```
p_DumpListTree("mytreeview", "root", 0)
```

Der obige Code zeigt, wie man den kompletten Inhalt einer Baumansicht unter Beibehaltung seiner Struktur ausgibt.

54.14 Treeview.HRules**BEZEICHNUNG**

Treeview.HRules – zeichnet horizontale Linien zwischen den Zeilen

PLATTFORMEN

Windows, Linux, macOS

BESCHREIBUNG

Setzen Sie dies auf `True`, um für die Baumansicht horizontale Linien zwischen den Zeilen zu aktivieren.

TYP

Boolesch

ANWENDBARKEIT

I

54.15 Treeview.InsertLeaf

BEZEICHNUNG

`Treeview.InsertLeaf` – fügt ein neues Blatt in den Baum ein

ÜBERSICHT

```
moai.DoMethod(id, "InsertLeaf", id$, node$, pred$, [icon1,] entry1$, ...)
```

BESCHREIBUNG

Fügt ein neues Blatt an der durch `node$` und `pred$` definierten Stelle in die Baumansicht ein. Ein Blatt ist ein Bauelement, das keine Elemente hat. `id$` muss ein eindeutiger Zeichenketten-Identifikator sein, den Sie verwenden werden, um auf das neu eingefügte Baum-Blatt zu verweisen.

Die Eingabedaten für das neue Blatt müssen in den letzten Parametern übergeben werden. Wenn die Baumansicht mehrere Spalten hat, müssen Sie für alle Spalten der Baumansicht individuelle Eingabedaten übergeben. Die Eingabedaten bestehen aus einer Zeichenkette und, wenn für die Spalte das Attribut `Treeviewcolumn.Icon` gesetzt ist, einem Symbol für jede Spalte. Das Symbol muss vor der Zeichenkette übergeben werden und es muss ein Identifikator eines Hollywood-Pinsels/-Icons sein, der dann als Symbol für den Eintrag verwendet werden soll. Wenn `Treeviewcolumn.Icon` nicht gesetzt ist, müssen Sie das Attribut `icon` weglassen und nur Textdaten für den Baum-Eintrag übergeben. Wenn Sie `Treeviewcolumn.Icon` auf `True` gesetzt haben und kein Symbol in dieser bestimmten Zeile und Spalte anzeigen möchten, können Sie auch den speziellen Wert `-1` übergeben. In diesem Fall zeigt RapaGUI kein Symbol an, obwohl `Treeviewcolumn.Icon` auf `True` gesetzt wurde. Beachten Sie, dass automatisch generierte IDs nicht verwendet werden können. Bitte lesen Sie auch das Kapitel über den Bilder-Cache von RapaGUI, um mehr über die Unterstützung von Icons in RapaGUI zu erfahren. Siehe [Abschnitt 3.20 \[Bilder-Cache\]](#), [Seite 32](#), für Details.

Wenn eine Spalte ein Auswahlkästchen (Checkbox) enthält, müssen Sie `"On"`, `"True"` oder `"1"` übergeben, um das Auswahlkästchen zu aktivieren und irgend einen anderen Text, um das Auswahlkästchen zu deaktivieren.

In `node$` müssen Sie den Knoten übergeben, in dessen Liste das neue Blatt eingefügt wird. Dies kann der Zeichenketten-Identifikator eines Knotens oder einer der folgenden Sonderwerte sein:

`Root` Der Wurzelknoten.

In `pred$` müssen Sie den Knoten oder das Blatt angeben, der zum Vorgänger des einzufügenden Blattes wird, d.h. das neue Blatt wird nach dem in `pred$` angegebenen Element eingefügt. Dies kann der Zeichenketten-Identifikator eines Knotens oder eines Blattes oder einer der folgenden speziellen Werte sein:

Head	Als erstes Element des Knotens einfügen.
Tail	Als letztes Element des Knotens einfügen.
Active	Fügt das neue Blatt nach dem aktiven Element ein. Wenn es keinen aktiven Eintrag gibt, wird das Objekt als letztes untergeordnetes Element eingefügt.

Beachten Sie, dass unter AmigaOS und kompatiblen Betriebssystemen diese Fähigkeit mindestens MUI 4.0 erfordert.

EINGABEN

<code>id</code>	ID des Treeview-Objekts
<code>id\$</code>	eindeutiger Zeichenketten-Identifikator für das neue Baum-Blatt
<code>node\$</code>	Zeichenketten-ID des einzufügenden Knotens oder spezieller Wert (siehe oben)
<code>pred\$</code>	Zeichenketten-ID des Vorgängerknotens/-Blattes oder speziellen Wertes (siehe oben)
<code>icon1</code>	optional: Symbol für den Eintrag in der ersten Spalte; dieses muss nur übergeben werden, wenn die Spalte das Icon-Attribut gesetzt hat; beachten Sie, dass dies ein numerischer Identifikator sein muss und automatisch generierte IDs in diesem Fall nicht gültig sind
<code>entry1\$</code>	Eintrag zum Einfügen in die erste Spalte
<code>...</code>	mehr Einträge, wenn der Baum mehrere Spalten hat

54.16 Treeview.InsertNode

BEZEICHNUNG

`Treeview.InsertNode` – fügt einen neuen Knoten in den Baum ein

ÜBERSICHT

```
moai.DoMethod(id, "InsertNode", id$, node$, pred$, entry$[, icon])
```

BESCHREIBUNG

Fügt einen neuen Knoten in die Baumansicht an der Position ein, die durch `node$` und `pred$` definiert ist. Ein Knoten ist ein Baumansichtselement, das untergeordnete Elemente haben und vom Benutzer geöffnet werden kann. `id$` muss ein eindeutiger Zeichenketten-Identifikator sein, den Sie verwenden werden, um auf den neu eingefügten Baum-Knoten zu verweisen.

`entry$` gibt die gewünschte Beschriftung für den neuen Knoten an. Sie können auch die ID eines Hollywood-Pinsels/-Icons im optionalen Argument `icon` übergeben, wenn neben der Knotenbeschriftung ein Symbol angezeigt werden soll. Beachten Sie, dass Knotensymbole immer unterstützt werden. Sie müssen `Treeviewcolumn.Icon` nicht auf `True`

setzen, um Symbole neben den Knotenbeschriftungen anzuzeigen. `Treeviewcolumn.Icon` gilt nur für Blätter; Knoten unterstützen immer Symbole. Beachten Sie, dass automatisch generierte IDs nicht verwendet werden können. Bitte lesen Sie auch den Bilder-Cache von RapaGUI, um mehr über die Unterstützung von Symbolen in RapaGUI zu erfahren. Siehe [Abschnitt 3.20 \[Bilder-Cache\]](#), [Seite 32](#), für Details.

In `node$` müssen Sie den Knoten übergeben, dessen Liste zum Einfügen des neuen Knotens verwendet wird. Dies kann der Zeichenketten-Identifikator eines Knotens oder einer der folgenden speziellen Werte sein:

Root Der Wurzelknoten.

In `pred$` müssen Sie den Knoten oder das Blatt angeben, der zum Vorgänger des einzufügenden Knotens wird, d.h. der neue Knoten wird nach dem in `pred$` angegebenen Element eingefügt. Dies kann der Zeichenketten-Identifikator eines Knotens oder eines Blattes oder einer der folgenden speziellen Werte sein:

Head Als erstes Element des Knotens einfügen.

Tail Als letztes Element des Knotens einfügen.

Active Fügt das neue Blatt nach dem aktiven Element ein. Wenn es keinen aktiven Eintrag gibt, wird das Objekt als letztes untergeordnetes Element eingefügt.

Beachten Sie, dass unter AmigaOS und kompatiblen Betriebssystemen diese Fähigkeit mindestens MUI 4.0 erfordert.

EINGABEN

<code>id</code>	ID des Treeview-Objekts
<code>id\$</code>	eindeutiger Zeichenketten-Identifikator für den neuen Baum-Knoten
<code>node\$</code>	Zeichenketten-ID des einzufügenden Knotens oder spezieller Wert (siehe oben)
<code>pred\$</code>	Zeichenketten-ID des Vorgängerknotens/-Blattes oder speziellen Wertes (siehe oben)
<code>entry\$</code>	gewünschte Knotenbeschriftung
<code>icon</code>	optional: gewünschtes Knotensymbol (standardmäßig -1, d.h. kein Symbol); beachten Sie, dass dies ein numerischer Identifikator sein muss und automatisch generierte IDs in diesem Fall nicht gültig sind

54.17 Treeview.Open

BEZEICHNUNG

Treeview.Open – öffnet einen Baumknoten

ÜBERSICHT

```
moai.DoMethod(id, "Open", node$[, all])
```

BESCHREIBUNG

Öffnet den angegebenen Baum-Knoten. Wenn das optionale Argument `all` auf `True` gesetzt ist, werden auch alle Elemente des angegebenen Knotens geöffnet.

Sie können folgende spezielle Werte für `node$` übergeben:

<code>Root</code>	Der Wurzelknoten. Da der Wurzelknoten nie sichtbar ist, werden automatisch alle Elemente des unsichtbaren Wurzelknotens geöffnet.
<code>Active</code>	Der aktive Knoten.

EINGABEN

<code>id</code>	ID des Treeview-Objekts
<code>node\$</code>	Zeichenketten-ID des zu öffnenden Knotens oder spezieller Wert (siehe oben)
<code>all</code>	optional: <code>True</code> , um auch alle Knotenelemente zu öffnen (Voreingestellt: <code>False</code>)

54.18 Treeview.Remove

BEZEICHNUNG

`Treeview.Remove` – entfernt ein Bauelement

ÜBERSICHT

```
moai.DoMethod(id, "Remove", item$)
```

BESCHREIBUNG

Entfernt ein Element aus einer Baumansicht (entweder ein Blatt oder einen kompletten Knoten). Sie müssen die Zeichenketten-ID des zu entfernenden Eintrags in `item$` übergeben. Alternativ können Sie auch einen der folgenden Sonderwerte in `item$` übergeben:

<code>Active</code>	Entfernt das aktive Element.
<code>Root</code>	Entfernt alle Elemente in der Baumansicht. Dies entspricht der Methode <code>Treeview.Clear</code> . (V2.0)

EINGABEN

<code>id</code>	ID des Treeview-Objekts
<code>item\$</code>	ID des zu entfernenden Baueintrags oder spezieller Wert (siehe oben)

54.19 Treeview.StartEditing

BEZEICHNUNG

`Treeview.StartEditing` – benachrichtigt, wenn ein Element bearbeitet wird

BESCHREIBUNG

Wenn Sie eine Benachrichtigung für dieses Attribut einrichten, führt RapaGUI die Callback-Funktion immer dann aus, wenn der Benutzer ein Baumansichtselement bearbeiten möchte, indem er langsam darauf doppelklickt oder wenn das Skript die Methoden `Treeviewleaf.Edit` oder `Treeviewnode.Edit` ausführt. Ihre Callback-Funktion kann dann die Bearbeitungsanfrage des Benutzers erlauben oder verbieten.

Um die Anfrage zu verbieten, muss Ihre Callback-Funktion **False** zurückgeben. Um die Anfrage zuzulassen, muss sie hingegen **True** zurückgeben.

Beachten Sie, dass Sie `Treeviewcolumn.Editable` auf **True** setzen müssen, wenn Blätter in der entsprechenden Spalte bearbeitbar sein sollen. Um Knoten editierbar zu machen, müssen Sie `Treeview.EditableNodes` auf **True** setzen.

Ihre Callback-Funktion wird mit folgenden zusätzlichen Argumenten aufgerufen:

- Item:** ID des Knotens oder Blattes, den der Benutzer bearbeiten möchte.
- Column:** Spaltenindex des Eintrags, den der Benutzer bearbeiten möchte. Für Knoten ist dies immer 0.

Siehe [Abschnitt 3.7 \[Benachrichtigungen der Attribute\]](#), Seite 14, für Details.

TYP

Boolesch

ANWENDBARKEIT

N

54.20 `Treeview.ValueChange`

BEZEICHNUNG

`Treeview.ValueChange` – benachrichtigt, wenn sich der Wert des Baumansicht-Elements ändert

BESCHREIBUNG

Wenn Sie eine Benachrichtigung für dieses Attribut einrichten, führt RapaGUI die Callback-Funktion aus, sobald sich der Wert eines Baumansichts-Elementes geändert hat, weil der Benutzer das Auswahlkästchen (Checkbox) aktiviert oder das Element bearbeitet hat. Beachten Sie, dass `Treeview.ValueChange` nicht ausgelöst wird, wenn der Wert des Elements mit den Methoden `Treeviewleaf.SetItem` oder `Treeviewleaf.SetState` geändert wurde. Es wird auch nicht ausgelöst, wenn `Treeviewnode.Name` zum Ändern der Beschriftung des Elements verwendet wurde.

Für Elemente in Auswahlkästchen-Spalten wird `TriggerValue` entweder auf **True** oder **False** gesetzt, was den neuen Auswahlkästchen-Status widerspiegelt. Für Elemente in Textspalten enthält `TriggerValue` den neuen Elementtext.

Außerdem wird Ihre Callback-Funktion mit folgenden zusätzlichen Argumenten aufgerufen:

- Item:** ID des Eintrags (Knoten oder Blatt), dessen Wert sich geändert hat.
- Column:** Spaltenindex des Elements, dessen Wert sich geändert hat. Für Knoten ist dies immer 0.

Siehe [Abschnitt 3.7 \[Benachrichtigungen der Attribute\]](#), Seite 14, für Details.

TYP

Boolesch oder Zeichenkette (abhängig vom Spaltentyp)

ANWENDBARKEIT

N

54.21 Treeview.VRules

BEZEICHNUNG

TreeView.VRules – zeichnet vertikale Linien zwischen den Spalten

BESCHREIBUNG

Setzen Sie dies auf `True`, um vertikale Linien zwischen den Spalten für die Baumansicht zu aktivieren.

TYP

Boolesch

ANWENDBARKEIT

I

55 Treeviewcolumn-Klasse (Baumansichtspalten)

55.1 Übersicht

Die Treeviewcolumn-Klasse (Baumansichtspalten) wird beim Erstellen von Baumansichten benötigt. Sie erlaubt Ihnen, verschiedene Attribute für die Spalten Ihrer Baumansichten anzugeben.

Die Treeviewcolumn-Klasse muss immer in eine `<treeview>`-Definition eingebettet sein. Ihr XML-Tag ist `<column>`. Siehe [Abschnitt 54.1 \[TreeView-Klasse\], Seite 277](#), für Details.

Beachten Sie, dass Sie keine Instanzen dieser Klasse mit `moai.CreateObject()` erzeugen können. Die Spaltennummern der Baumansicht sind derzeit statisch, d.h. Sie können zur Laufzeit keine Spalten hinzufügen oder entfernen.

55.2 Treeviewcolumn.Align

BEZEICHNUNG

Treeviewcolumn.Align – setzt/ermittelt die Spaltenausrichtung

BESCHREIBUNG

Stellen Sie die Spaltenausrichtung ein oder rufen Sie sie ab. Dies kann einer der folgenden Werte sein:

Left Links ausgerichtet. Dies ist auch voreingestellt.

Right Rechts ausgerichtet.

Center Zentriert ausgerichtet.

Auf Nicht-AmigaOS-Systemen wird dieses Attribut nur für das Dataview-Widget unterstützt. Wenn Sie eine Baumansicht erstellen und `Treeviewcolumn.Align` auf einen anderen Wert als `Left` gesetzt ist, wird RapaGUI automatisch zum Dataview-Widget wechseln. Wenn Sie dieses Attribut nicht zum Zeitpunkt der Erstellung angeben, sondern es später mithilfe von `moai.Set()` festlegen möchten, müssen Sie explizit ein Dataview-Widget anfordern, indem Sie das Attribut `TreeView.ForceMode` setzen.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

ISG

55.3 Treeviewcolumn.Checkbox

BEZEICHNUNG

Treeviewcolumn.Checkbox – setzt die Spalte in den Auswahlkästchen-Modus

BESCHREIBUNG

Setzen Sie dies auf `True`, um diese Spalte als Auswahlkästchen-Spalte (Checkbox) zu markieren. Auswahlkästchen-Spalten zeigen Auswahlkästchen anstelle von Text an. Immer wenn der Text eines Eintrags in einer Auswahlkästchen-Spalte auf "On", "True"

oder "1" gesetzt ist, wird das Auswahlkästchen ausgewählt. Alle anderen Texte führen zu einem nicht markierten Auswahlkästchen.

Sie können die Zustände der Auswahlkästchen mit der Methode `Treeviewleaf.SetState` ändern. Ebenso ist es möglich, den Status eines Auswahlkästchen über die Methode `Treeviewleaf.GetState` abzurufen.

Um benachrichtigt zu werden, wenn der Benutzer den Status eines Auswahlkästchen umschaltet, müssen Sie das Attribut `Treeview.ValueChange` überwachen.

Beachten Sie auch, dass `Treeviewcolumn.Checkbox`, `Treeviewcolumn.Editable` und `Treeviewcolumn.Icon` sich gegenseitig ausschließen. Sie können keine Auswahlkästchen-Spalten erstellen, die editierbar sind oder Symbole anzeigen.

Auswahlkästchen in der ersten Spalte der Baumansicht werden derzeit unter Windows, macOS und Linux nicht unterstützt.

TYP

Boolesch

ANWENDBARKEIT

I

55.4 Treeviewcolumn.Editable

BEZEICHNUNG

`Treeviewcolumn.Editable` – erlaubt das Bearbeiten von Spaltenblättern

BESCHREIBUNG

Setzen Sie dies auf `True`, um den Benutzern die Bearbeitung der Blätter in dieser Spalte zu erlauben. Der Benutzer wird dann in der Lage sein, alle Blätter in dieser Spalte zu bearbeiten, indem er einen langsamen Doppelklick ausführt, d.h. langsam die linke Maustaste zweimal hintereinander drückt.

Wenn Sie nur einige Blätter in der Spalte bearbeiten möchten, müssen Sie dieses Attribut auf `True` setzen und das Attribut `Treeview.StartEditing` überwachen. `Treeview.StartEditing` wird dann ausgelöst, wenn der Benutzer versucht, ein Blatt zu bearbeiten, und Ihre Callback-Funktion kann `False` zurückgeben, um die Bearbeitung bestimmter Elemente zu verbieten. Siehe [Abschnitt 54.19 \[Treeview.StartEditing\]](#), [Seite 289](#), für Details.

Um benachrichtigt zu werden, wenn sich der Wert eines Baumansicht-Blattes ändert, weil der Benutzer es bearbeitet hat, müssen Sie das Attribut `Treeview.ValueChange` überwachen.

Um die Bearbeitung eines Baumansicht-Blattes manuell zu starten, rufen Sie die Methode `Treeviewleaf.Edit` auf.

Bitte beachten Sie, dass dieses Attribut nur für Baumansicht-Blätter gilt. Wenn Sie möchten, dass auch Knotenbeschriftungen editierbar sind, müssen Sie das Attribut `Treeview.EditableNodes` auf `True` setzen.

Außerdem schließen sich `Treeviewcolumn.Checkbox` und `Treeviewcolumn.Editable` gegenseitig aus. Sie können keine editierbaren Auswahlkästchen-Spalten erstellen.

Beachten Sie, dass unter AmigaOS und kompatiblen Betriebssystemen diese Fähigkeit mindestens MUI 4.0 erfordert.

TYP

Boolesch

ANWENDBARKEIT

I

55.5 Treeviewcolumn.Hide

BEZEICHNUNG

Treeviewcolumn.Hide – blendet die Baumansicht ein oder aus

BESCHREIBUNG

Mit diesem Attribut können Sie einzelne Baumspalten ein- oder ausblenden. Beachten Sie, dass die Spalten immer noch da sind, sie sind nur unsichtbar. So dürfen Sie beim Hinzufügen von Baumansicht-Blättern mit `Treeview.InsertLeaf` versteckte Spalten nicht vergessen.

Auf Nicht-AmigaOS-Systemen wird dieses Attribut nur für das Dataview-Widget unterstützt. Wenn Sie eine Baumansicht erstellen und `Treeviewcolumn.Hide` auf `True` gesetzt ist, benutzt RapaGUI automatisch die Dataview-Widget. Wenn Sie dieses Attribut bei der Erstellung nicht angeben, es aber später mit `moai.Set()` setzen wollen, müssen Sie explizit ein Dataview-Widget anfordern, indem Sie das Attribut `Treeview.ForceMode` setzen.

TYP

Boolesch

ANWENDBARKEIT

ISG

55.6 Treeviewcolumn.Icon

BEZEICHNUNG

Treeviewcolumn.Icon – aktiviert Blattsymbole für diese Spalte

BESCHREIBUNG

Setzen Sie dies auf `True`, wenn Baumansicht-Blätter in dieser Spalte Symbole verwenden. In diesem Fall müssen Sie der Methode `Treeview.InsertLeaf` ein Hollywood-Pinsel/-Icons als `Symbol` übergeben.

Beachten Sie, dass dieses Attribut nur für Baumansicht-Blätter gilt. Baumansicht-Knoten können Symbole verwenden, ohne ein bestimmtes Attribut zu setzen. Symbole für Baumansicht-Knoten sind immer verfügbar, aber für Blätter müssen Sie dieses Attribut zuerst auf `True` setzen.

Beachten Sie, dass RapaGUI das Bild möglicherweise skaliert, um es an die DPI-Einstellung des aktuellen Monitors anzupassen. Bitte lesen Sie das Kapitel

über High-DPI-Unterstützung für weitere Informationen. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), [Seite 21](#), für Details.

Bitte lesen Sie auch den Bilder-Cache von RapaGUI, um mehr über die Unterstützung von Symbolen in RapaGUI zu erfahren. Siehe [Abschnitt 3.20 \[Bilder-Cache\]](#), [Seite 32](#), für Details.

Beachten Sie auch, dass unter AmigaOS und kompatiblen Betriebssystemen diese Fähigkeit mindestens MUI 4.0 erfordert.

TYP

Boolesch

ANWENDBARKEIT

I

55.7 Treeviewcolumn.IconScale

BEZEICHNUNG

Treeviewcolumn.IconScale – konfiguriert die automatische Bildskalierung (V2.0)

BESCHREIBUNG

Wenn `Treeviewcolumn.Icon` auf einen Rasterpinsel gesetzt wurde und RapaGUI auf einem Display mit hohem DPI ausgeführt wird, skaliert RapaGUI automatisch die Rastergrafiken des Pinsels, um sie an die DPI-Einstellung des aktuellen Monitors anzupassen. Wenn Sie dies nicht möchten, setzen Sie diesen Tag auf `False`.

Alternativ können Sie die automatische Bildskalierung auch global deaktivieren, indem Sie beim Aufrufen vom Befehl `@REQUIRE` mit RapaGUI den Tag `ScaleGUI` auf `False` setzen. Siehe [Abschnitt 3.4 \[Initialisierung von RapaGUI\]](#), [Seite 11](#), für Details.

Bitte lesen Sie auch das Kapitel über High-DPI-Unterstützung in RapaGUI, um mehr über die Unterstützung von High-DPI-Displays zu erfahren. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), [Seite 21](#), für Details.

TYP

Boolesch

ANWENDBARKEIT

I

55.8 Treeviewcolumn.IconType

BEZEICHNUNG

Treeviewcolumn.IconType – legt den zu verwendenden Icon-Typ fest (V2.0)

BESCHREIBUNG

Mit diesem Attribut können Sie den Typ des Hollywood-Bildobjekts festlegen, das im Attribut `Treeviewcolumn.Icon` übergeben wird. Standardmäßig erwartet `Treeviewcolumn.Icon` einen Hollywood-Pinsel. Durch Setzen von `Treeviewcolumn.IconType` können Sie jedoch einen anderen Hollywood-Bildtyp verwenden.

Die folgenden Bildtypen sind derzeit verfügbar:

- Brush** Verwendet einen Hollywood-Pinsel. Dies ist der Standardtyp. Sie können entweder Raster- oder Vektorpinsel verwenden. Vektorpinsel haben den Vorteil, dass sie ohne Qualitätsverlust auf jede beliebige Auflösung skaliert werden können. Dies ist sehr nützlich, wenn Sie Programme entwerfen, die mit High-DPI-Monitoren kompatibel sein sollen. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), [Seite 21](#), für Details.
- Icon** Verwendet ein Hollywood-Icon. Dieser Bildtyp hat den Vorteil, dass er mehrere Teilbilder unterschiedlicher Größe enthalten kann. Dadurch ist es möglich, Bilder in unterschiedlichen Auflösungen bereitzustellen, was bei der Entwicklung von Programmen, die mit High-DPI-Monitoren kompatibel sein sollen, sehr nützlich sein kann. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), [Seite 21](#), für Details.

Beachten Sie, dass Sie den Standardwert dieses Attributs global auf Hollywood-Icon ändern können, indem Sie den Tag `Application.UseIcons` setzen. Siehe [Abschnitt 9.13 \[Application.UseIcons\]](#), [Seite 70](#), für Details.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

I

55.9 Treeviewcolumn.Title

BEZEICHNUNG

`Treeviewcolumn.Title` – setzt/ermittelt den Titel der Spalte

BESCHREIBUNG

Setzt oder ermittelt den Titel der Spalte. Der Titel wird immer oben in der Baumansicht angezeigt und verschwindet nicht, wenn die Baumansicht gescrollt wird.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

55.10 Treeviewcolumn.Width

BEZEICHNUNG

`Treeviewcolumn.Width` – setzt/ermittelt die Spaltenbreite

BESCHREIBUNG

Setzt oder ermittelt die Spaltenbreite in geräteunabhängige Pixel. Die Voreinstellung ist -1, was bedeutet, dass die Spalte so groß wie ihr größter Eintrag sein sollte.

TYP

Zahl

ANWENDBARKEIT

ISG

56 Treeviewleaf-Klasse (Baumansichtblätter)

56.1 Übersicht

Die Treeviewleaf-Klasse (Baumansichtblätter) wird beim Erstellen von Baumansichten benötigt. Sie erlaubt Ihnen, die einzelnen Blätter der Baumansicht zu definieren und verschiedene Attribute für sie zu setzen. Jedes Blatt Ihrer Baumansicht muss einen Eintrag für jede Spalte Ihrer Baumansicht enthalten. So müssen Ihre `<leaf>`-Definitionen so viele `<item>`-Definitionen enthalten, wie es Spalten in Ihrer Baumansicht gibt. Siehe [Abschnitt 57.1 \[Treeviewleafitem-Klasse\], Seite 305](#), für Details.

Die Treeviewleaf-Klasse muss immer in eine `<treeview>`-Definition eingebettet werden. Siehe [Abschnitt 54.1 \[TreeView-Klasse \(Baumansicht\)\], Seite 277](#), für Details.

Beachten Sie, dass Sie keine Instanzen dieser Klasse mit `moai.CreateObject()` erzeugen können. Stattdessen müssen Sie `TreeView.InsertLeaf` verwenden, um Blätter zur Laufzeit zu erzeugen.

56.2 Treeviewleaf.Edit

BEZEICHNUNG

Treeviewleaf.Edit – startet die Bearbeitung von Blattelementen durch das Programm

ÜBERSICHT

```
moai.DoMethod(id, "Edit", column)
```

BESCHREIBUNG

Diese Methode kann verwendet werden, um die Bearbeitung von Blattelementen durch das Programm zu starten. Normalerweise wird die Bearbeitung eines Eintrags vom Benutzer durch einen langsamen Doppelklick auf einen Eintrag gestartet. Diese Methode stellt eine Alternative zu diesem Benutzermechanismus dar.

Dies funktioniert nur, wenn `TreeViewcolumn.Editable` für die jeweilige Baumansicht-Spalte auf `True` gesetzt wurde.

Wenn der Benutzer die Bearbeitung beendet hat, wird das Attribut `TreeView.ValueChange` ausgelöst.

Beachten Sie, dass wenn Sie eine Überwachung für das Attribut `TreeView.StartEditing` installiert haben, diese Callback-Funktion zuerst um Erlaubnis fragt, bevor die Bearbeitung tatsächlich gestartet wird.

Um zu erfahren, wann Bearbeitungsvorgänge abgebrochen werden, können Sie das Attribut `TreeView.AbortEditing` überwachen.

Beachten Sie, dass unter AmigaOS und kompatiblen Betriebssystemen diese Fähigkeit mindestens MUI 4.0 erfordert.

EINGABEN

<code>id</code>	ID des Blattes
<code>column</code>	Spaltenindex des zu bearbeitenden Blattelements

56.3 Treeviewleaf.GetDisabled

BEZEICHNUNG

Treeviewleaf.GetDisabled – ermittelt den deaktivierten Status des Auswahlkästchen

ÜBERSICHT

```
state = moai.DoMethod(id, "GetDisabled", column)
```

BESCHREIBUNG

Gibt den deaktivierten Zustand des Auswahlkästchen (Checkbox) in der angegebenen Spalte zurück. Dies ist entweder **True** für deaktiviert oder **False** für aktiviert.

EINGABEN

id ID des Blattes
column Spaltenindex des Auswahlkästchen

RÜCKGABEWERTE

state True wenn das Auswahlkästchen deaktiviert ist, sonst **False**

56.4 Treeviewleaf.GetIcon

BEZEICHNUNG

Treeviewleaf.GetIcon – ermittelt die ID vom Symbol des Blattelements

ÜBERSICHT

```
br = moai.DoMethod(id, "GetIcon", column)
```

BESCHREIBUNG

Gibt die ID des Hollywood-Pinsel/-Icons des in der angegebenen Spalte verwendeten Blattsymbols zurück. Wenn das angegebene Element kein Symbol hat, wird -1 zurückgegeben.

EINGABEN

id ID des Blattes
column Spaltenindex des Elements

RÜCKGABEWERTE

br ID eines Hollywood-Pinsels/-Icons oder -1, wenn es kein Symbol gibt.

56.5 Treeviewleaf.GetItem

BEZEICHNUNG

Treeviewleaf.GetItem – ermittelt die Beschriftung des Blattelements

ÜBERSICHT

```
t$ = moai.DoMethod(id, "GetItem", column)
```

BESCHREIBUNG

Gibt den Beschriftungstext des Elements in der angegebenen Blattspalte zurück.

EINGABEN

`id` ID des Blattes
`column` Spaltenindex des Elements

RÜCKGABEWERTE

`t$` Beschriftung des angegebenen Elements

56.6 Treeviewleaf.GetState

BEZEICHNUNG

`Treeviewleaf.GetState` – ermittelt den Umschaltstatus des Auswahlkästchen

ÜBERSICHT

```
state = moai.DoMethod(id, "GetState", column)
```

BESCHREIBUNG

Gibt den Umschaltstatus des Auswahlkästchen (Checkbox) in der angegebenen Spalte zurück. Dies ist entweder `True`, wenn das Auswahlkästchen aktiviert oder `False`, wenn sie deaktiviert ist.

EINGABEN

`id` ID des Blattes
`column` Spaltenindex des Auswahlkästchen

RÜCKGABEWERTE

`state` `True`, wenn des Auswahlkästchen markiert ist, ansonsten `False`

56.7 Treeviewleaf.SetDisabled

BEZEICHNUNG

`Treeviewleaf.SetDisabled` – setzt den Deaktivierungsstatus des Auswahlkästchen

ÜBERSICHT

```
moai.DoMethod(id, "SetDisabled", column, state)
```

BESCHREIBUNG

Legt den deaktivierten Status des Auswahlkästchen (Checkbox) in der angegebenen Spalte fest. Übergeben Sie `True`, um das Auswahlkästchen zu deaktivieren oder `False`, um sie zu aktivieren.

EINGABEN

`id` ID des Blattes
`column` Spaltenindex des Auswahlkästchen
`state` `True`, wenn des Auswahlkästchen deaktiviert werden soll, andernfalls `False`.

56.8 Treeviewleaf.SetIcon

BEZEICHNUNG

Treeviewleaf.SetIcon – setzt das Symbol für das Blattelement

ÜBERSICHT

```
moai.DoMethod(id, "SetIcon", column, br)
```

BESCHREIBUNG

Fügt dem angegebenen Blatteintrag ein Symbol hinzu. Sie müssen die ID eines Hollywood-Pinsels/-Icons übergeben, der als Symbol in `br` verwendet werden soll. Um ein Symbol aus einem Blatteintrag zu entfernen, übergeben Sie `-1` in `br`.

Bitte lesen Sie auch den Bilder-Cache von RapaGUI, um mehr über die Unterstützung von Symbolen in RapaGUI zu erfahren. Siehe [Abschnitt 3.20 \[Bilder-Cache\]](#), Seite 32, für Details.

EINGABEN

<code>id</code>	ID des Blattes
<code>column</code>	Spaltenindex des Blattelements
<code>br</code>	ID eines Hollywood-Pinsels/-Icons, der als Symbol verwendet werden soll oder <code>-1</code> , um das Element-Symbol zu entfernen.

56.9 Treeviewleaf.SetItem

BEZEICHNUNG

Treeviewleaf.SetItem – setzt die Beschriftung des Blattelements

ÜBERSICHT

```
moai.DoMethod(id, "SetItem", column, l$)
```

BESCHREIBUNG

Diese Methode überschreibt die Beschriftung mit dem in `l$` angegebenen Text im `id` angegebenen Blattelement.

EINGABEN

<code>id</code>	ID des Blattes
<code>column</code>	Spaltenindex des Blattelements
<code>l\$</code>	gewünschte neue Beschriftung für das Blattelement

56.10 Treeviewleaf.SetState

BEZEICHNUNG

Treeviewleaf.SetState – setzt den Umschaltstatus des Auswahlkästchen

ÜBERSICHT

```
moai.DoMethod(id, "SetState", column, state)
```


BESCHREIBUNG

Setzt den Umschaltstatus des Auswahlkästchen (Checkbox) in der angegebenen Spalte. Übergeben Sie `True`, um das Auswahlkästchen anzukreuzen/Häckchen zu setzen oder `False`, um das Kreuz/Häckchen zu entfernen.

EINGABEN

<code>id</code>	ID des Blattes
<code>column</code>	Spaltenindex des Blattelements
<code>state</code>	<code>True</code> , wenn das Auswahlkästchen angekreuzt werden soll, ansonsten <code>False</code>

56.11 Treeviewleaf.UID

BEZEICHNUNG

Treeviewleaf.UID – ermittelt die UID des Blattes

BESCHREIBUNG

Ermittelt die UID des angegebenen Baumblattes ab. Diese UID wird benötigt, wenn Sie die Methode `Treeview.GetEntry` ausführen.

TYP

MOAI-Objekt

ANWENDBARKEIT

G

57 Treeviewleafitem-Klasse (Baumansichtblätterelement)

57.1 Übersicht

Die `Treeviewleafitem`-Klasse (Baumansichtblätterelement) wird beim Erstellen von Baumansichten (`Treeview`) benötigt. Es erlaubt Ihnen, die einzelnen Elemente für jede Spalte eines Baumansicht-Blattes zu definieren. Sie müssen so viele `<item>`-Definitionen verwenden, wie es Spalten in Ihrer Baumansicht gibt. Da diese Klasse eine Unterklasse von der `Treeviewleaf`-Klasse (Baumansichtblätter) ist, müssen Ihre `<item>`-Definitionen in den Tag `<leaf>` eingebettet werden. Blatt-Elemente können mit einer Beschriftung und optional mit einem Symbol versehen werden. Siehe [Abschnitt 54.1 \[Treeview-Klasse\]](#), [Seite 277](#), für ein Beispiel.

Beachten Sie, dass Sie keine Instanzen dieser Klasse mit `moai.CreateObject()` erzeugen können. Stattdessen müssen Sie `Treeview.InsertLeaf` verwenden, um zur Laufzeit Blatt-Elemente zu erstellen.

57.2 Treeviewleafitem.Icon

BEZEICHNUNG

`Treeviewleafitem.Icon` – setzt das Bild

BESCHREIBUNG

Setzen Sie dieses Attribut auf eine Hollywood-ID eines Hollywood-Pinsels oder -Icons, um Ihrem Baumansichtblätterelement ein Bild hinzuzufügen. Ob dieses Attribut einen Hollywood-Pinsel oder ein Hollywood-Icon erwartet, hängt davon ab, was Sie im Attribut `Treeviewcolumn.IconType` angeben. Standardmäßig erwartet `Treeviewleafitem.Icon` einen Hollywood-Pinsel.

Beachten Sie, dass RapaGUI das Bild möglicherweise skaliert, um es an die DPI-Einstellung des aktuellen Monitors anzupassen. Bitte lesen Sie das Kapitel über [High-DPI-Unterstützung](#) für weitere Informationen. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), [Seite 21](#), für Details.

Bitte lesen Sie auch das Kapitel [Bilder-Cache](#) von RapaGUI, um mehr über die Unterstützung von Symbolen/Icons in RapaGUI zu erfahren. Siehe [Abschnitt 3.20 \[Bilder-Cache\]](#), [Seite 32](#), für Details.

TYP

Zahl

ANWENDBARKEIT

I

58 Treeviewnode-Klasse (Baumansichtknoten)

58.1 Übersicht

Die Treeviewnode-Klasse (Baumansichtknoten) wird beim Erstellen von Baumansichten benötigt. Es erlaubt Ihnen, die einzelnen Knoten der Baumansicht zu definieren und verschiedene Attribute für sie zu setzen. Beachten Sie, dass Knoten im Gegensatz zu Baumansicht-Blättern auch bei mehrspaltigen Baumansichten nur einen einzigen Eintrag enthalten.

Die Treeviewnode-Klasse (Baumansichtknoten) muss immer in einer `<treeview>`-Definition eingebettet werden. Siehe [Abschnitt 54.1 \[TreeView-Klasse\]](#), [Seite 277](#), für Details.

Beachten Sie, dass Sie keine Instanzen dieser Klasse mit `moai.CreateObject()` erzeugen können. Stattdessen müssen Sie `TreeView.InsertNode` verwenden, um Knoten während der Laufzeit zu erstellen.

58.2 Treeviewnode.Edit

BEZEICHNUNG

TreeViewnode.Edit – fordert den Benutzer auf, eine Knotenbeschriftung zu bearbeiten

ÜBERSICHT

```
moai.DoMethod(id, "Edit")
```

BESCHREIBUNG

Diese Methode kann verwendet werden, um die Bearbeitung von Knotenbeschriftungen vom Programm aus zu starten. Normalerweise wird die Bearbeitung der Knotenbeschriftung vom Benutzer durch einen langsamen Doppelklick auf ein Element gestartet. Diese Methode stellt eine Alternative zu diesem Benutzermechanismus dar.

Dies funktioniert nur, wenn `TreeView.EditableNodes` in der Treeview-Klasse auf `True` gesetzt wurde.

Wenn der Benutzer die Bearbeitung beendet hat, wird das Attribut `TreeView.ValueChange` ausgelöst.

Bitte beachten Sie, dass, wenn Sie eine Überwachung auf dem Attribut `TreeView.StartEditing` installiert haben, diese Callback-Funktion zuerst um Erlaubnis gefragt wird, bevor die Bearbeitung tatsächlich gestartet wird.

Wenn Sie erfahren möchten, welche Bearbeitungsvorgänge abgebrochen werden, können Sie das Attribut `TreeView.AbortEditing` überwachen.

Beachten Sie auch, dass diese Methode unter AmigaOS und kompatiblen Betriebssystemen nur unter MUI 4.0 oder höher verfügbar ist.

EINGABEN

<code>id</code>	ID des Knotens
-----------------	----------------

58.3 Treeviewnode.Icon

BEZEICHNUNG

Treeviewnode.Icon – setzt das Knotenbild

BESCHREIBUNG

Setzen Sie dieses Attribut auf die ID des Hollywood-Pinsels oder -Icons, um Ihrem Baumansichtsknoten ein Bild hinzuzufügen. Ob dieses Attribut einen Hollywood-Pinsel oder ein Hollywood-Icon erwartet, hängt davon ab, was Sie im Attribut `Treeviewcolumn.IconType` angeben. Standardmäßig erwartet `Treeviewnode.Icon` einen Hollywood-Pinsel. Um das Icon aus einem Baumknoten zu entfernen, setzen Sie dieses Attribut auf `-1`.

Beachten Sie, dass RapaGUI das Bild möglicherweise skaliert, um es an die DPI-Einstellung des aktuellen Monitors anzupassen. Bitte lesen Sie das Kapitel über High-DPI-Unterstützung für weitere Informationen. Siehe [Abschnitt 3.12 \[High-DPI-Unterstützung\]](#), Seite 21, für Details.

Bitte lesen Sie auch das Kapitel über den Bilder-Cache von RapaGUI, um mehr über die Unterstützung von Icons in RapaGUI zu erfahren. Siehe [Abschnitt 3.20 \[Bilder-Cache\]](#), Seite 32, für Details.

TYP

Zahl

ANWENDBARKEIT

ISG

58.4 Treeviewnode.Name

BEZEICHNUNG

Treeviewnode.Name – setzt/ermittelt den Knotennamen

BESCHREIBUNG

Setzt oder ruft die Zeichenkette ab, die als Knotenname verwendet wird.

Beim Erstellen eines neuen Knotens ist dieses Attribut obligatorisch und muss immer festgelegt werden.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

58.5 Treeviewnode.UID

BEZEICHNUNG

Treeviewnode.UID – ermittelt die Knoten-UID

BESCHREIBUNG

Liefert die UID des angegebenen Baumknotens. Diese UID wird beim Ausführen der Methode `Treeview.GetEntry` benötigt.

TYP

MOAI-Objekt

ANWENDBARKEIT

G

59 VLine-Klasse (VertikalLinie)

59.1 Übersicht

Die VLine-Klasse (VertikalLinie) leitet sich von der Area-Klasse ab und erzeugt einfach eine vertikale Trennlinie, die als separate Gruppe von Widgets verwendet werden kann. Es gibt auch eine HLine-Klasse (HorizontalLinie), welche horizontale Trennlinien erzeugt. Siehe [Abschnitt 19.1 \[HLine-Klasse\]](#), [Seite 123](#), für Details.

Die VLine-Klasse hat keine Attribute.

60 VSpace-Klasse (VertikalAbstand)

60.1 Übersicht

Die VSpace-Klasse (VertikalAbstand) erzeugt einfach Objekte mit einer festen geräteunabhängige Pixelgröße. Dies wird typischerweise zur Feinabstimmung des GUI-Layouts verwendet. Um VSpace-Objekte zu erstellen, die frei skalierbar sind, können Sie stattdessen Rectangle-Klasse (Rechteck) verwenden. Siehe [Abschnitt 41.1 \[Rectangle-Klasse\]](#), [Seite 221](#), für Details.

60.2 VSpace.Height

BEZEICHNUNG

VSpace.Height – setzt den vertikalen Abstand

BESCHREIBUNG

Legt den gewünschten vertikalen Abstand für dieses Objekt in geräteunabhängige Pixel fest.

TYP

Zahl

ANWENDBARKEIT

I

61 VSplitter-Klasse

61.1 Übersicht

Die VSplitter-Klasse ist eine spezielle Variante der Group-Klasse. Sie erstellt eine vertikale Gruppe aus zwei untergeordneten Elementen mit einem Schieberahmen dazwischen, der es dem Benutzer ermöglicht, die Größe der beiden Elementen der Gruppe durch Ziehen des Schieberahmens individuell anzupassen.

Beachten Sie, dass VSplitter-Gruppen immer genau zwei untergeordneten Elementen enthalten müssen. Darüber hinaus müssen die Elemente in der Größe angepasst werden, da die VSplitter-Klasse eine Größenänderung ihrer untergeordneten Elemente ermöglicht. Hier ist ein Beispiel-XML-Auszug zum Erstellen eines vertikalen Splitter-Layouts mit zwei Textansichten:

```
<vsplitter>
  <textview>One</textview>
  <textview>Two</textview>
</vsplitter>
```

Die VSplitter-Klasse ist seit RapaGUI 2.0 verfügbar.

61.2 VSplitter.Border

BEZEICHNUNG

VSplitter.Border – stellt den Stil des Schieberahmens ein (V2.0)

BESCHREIBUNG

Setzen Sie diesen Wert auf **False**, damit der Schieberahmen einen anderen Rahmenstil verwendet. Beachten Sie, dass dies nicht auf allen Plattformen unterstützt wird.

TYP

Boolesch

ANWENDBARKEIT

I

61.3 VSplitter.Gravity

BEZEICHNUNG

VSplitter.Gravity – setzt/ermittelt die Anziehungskraft des Schieberahmens (V2.0)

BESCHREIBUNG

Stellt die Anziehungskraft des Schieberahmens ein oder ermittelt sie. Die Anziehungskraft ist ein Wert zwischen 0 und 100, der die Position des Schieberahmens steuert, während die Größe der Splittergruppe geändert wird. Der Anziehungskraftwert teilt der Splittergruppe mit, um wie viel das obere Element während der Größenänderung ändert. Zum Beispiel:

- 0: Nur das untere Element wird automatisch in der Größe angepasst.
- 50: Beide Elemente ändern im gleichen Verhältnis ihre Größe.

- 100: Nur das obere Element wird automatisch in der Größe angepasst.

Die Standardziehungskraft des Schieberahmens ist 0.

TYP

Zahl

ANWENDBARKEIT

SG

61.4 VSplitter.MinPaneSize

BEZEICHNUNG

VSplitter.MinPaneSize – Setzt oder ermittelt die minimale Elementgröße (V2.0)

BESCHREIBUNG

Legt die minimale Elementgröße fest oder ermittelt sie. Die standardmäßige Mindestelementgröße ist 0, was bedeutet, dass jedes Element durch Ziehen des Schieberahmens auf Null reduziert werden kann, wodurch eines der Elemente entfernt wird. Um dieses Verhalten zu verhindern (und das Ziehen des Schieberahmens außerhalb des Bereichs zu unterbinden), legen Sie eine Mindestgröße fest, zum Beispiel 20.

Dieser Wert ist in geräteunabhängige Pixel.

TYP

Zahl

ANWENDBARKEIT

ISG

61.5 VSplitter.Position

BEZEICHNUNG

VSplitter.Position – setzt/ermittelt die Position des Schieberahmens (V2.0)

BESCHREIBUNG

Legt die Position des Schieberahmens fest oder ermittelt sie, der die Elemente der Gruppe trennt. Dieser Wert ist in geräteunabhängige Pixel.

TYP

Zahl

ANWENDBARKEIT

SG

61.6 VSplitter.Split

BEZEICHNUNG

VSplitter.Split – teilt die Gruppe in zwei Elemente auf (V2.0)

ÜBERSICHT

```
moai.DoMethod(id, "Split", pos)
```

BESCHREIBUNG

Hiermit wird die Gruppe in zwei Elemente aufgeteilt. Das Argument `pos` gibt die Position des Schieberahmens in geräteunabhängige Pixel an. Wenn dieser Wert positiv ist, gibt er die Größe des oberen Elements an. Wenn er negativ ist, gibt sein absoluter Wert die Größe des unteren Elements an. Geben Sie schließlich 0 an, um die Standardposition zu wählen (die Hälfte der gesamten Fensterhöhe).

EINGABEN

<code>id</code>	Identifikator des Splitterobjekts
<code>pos</code>	gewünschte Position des Schieberahmens

61.7 VSplitter.Unsplit

BEZEICHNUNG

`VSplitter.Unsplit` – hebt die Teilung der Elemente auf (V2.0)

ÜBERSICHT

```
moai.DoMethod(id, "Unsplit", idx)
```

BESCHREIBUNG

Dadurch wird die Teilung der Elemente aufgehoben und das in `idx` angegebene Element ausgeblendet. Wenn `idx` 0 ist, wird das obere Element ausgeblendet, wenn es 1 ist, wird das untere Element ausgeblendet. Um beide Elemente wieder sichtbar zu machen, verwenden Sie die Methode `VSplitter.Split`.

EINGABEN

<code>id</code>	Identifikator des Splitterobjekts
<code>idx</code>	Index des zu entfernenden Elements

62 Window-Klasse (Fenster)

62.1 Übersicht

Die Window-Klasse erstellt Top-Level-Fenster, die mit Gruppen von Widgets gefüllt werden können. Diese Gruppen sind Elemente der Top-Level-Fenster. Als solche werden Gruppen automatisch weitergeleitet, wenn sich die Größe ihres übergeordneten Objekts ändert. So können Sie flexible GUI-Layouts erstellen, die sich automatisch an den verfügbaren Bildschirmbereich anpassen. Zusätzlich können Sie auch Menüleisten und Symbolleisten an ein Fenster zuweisen.

Fenster sind Elemente der Application-Klasse (Anwendung). Wenn Sie Fensterobjekte dynamisch erstellen, müssen Sie diese zunächst dem Applications-Objekt hinzufügen, indem Sie `Application.AddWindow` aufrufen.

Wenn Sie Dialoge öffnen müssen, können Sie dies tun, indem Sie Objekte der Dialog-Klasse erstellen. Dialoge sind spezielle Top-Level-Fenster, die den Rest vom Programm blockieren, bis sie geschlossen werden. Dialoge werden typischerweise verwendet, wenn Benutzeraktionen erforderlich sind, um eine Aufgabe fortzusetzen oder um anzuzeigen, dass das Programm gerade beschäftigt ist. Ein Dialog könnte dann z.B. einen Fortschrittsbalken anzeigen. Siehe [Abschnitt 16.1 \[Dialog-Klasse\], Seite 99](#), für Details.

Hier ist ein minimales Beispiel für die Erstellung eines Fensters in XML:

```
<window title="Hello World!">
  <vgroup>
    <button id="btn">Hello World!</button>
  </vgroup>
</window>
```

Beachten Sie, dass das Wurzelement eines Fensters immer ein einzelnes Gruppenobjekt sein muss, d.h. eine Instanz von Group-Klasse (Gruppen). Siehe [Abschnitt 18.1 \[Group-Klasse\], Seite 109](#), für Details. In unserem Beispiel verwenden wir eine `<vgroup>` als Wurzelement. Es ist nicht erlaubt, mehrere Elemente auf der Wurzelebene des Fensters zu haben, darum dürfen Sie nur ein einziges Gruppenobjekt als Wurzelement verwenden.

Fenster, die im XML-Code definiert oder von `moai.CreateApp()` erstellt wurden, werden automatisch geöffnet, es sei denn, Sie setzen explizit `Window.Open` auf `False`, damit sie geschlossen bleiben. Fenster, die mit `moai.CreateObject()` erstellt wurden, werden dagegen nicht automatisch geöffnet. Sie müssen sie zunächst der Fensterliste des Programms hinzufügen, indem Sie die Methode `Application.AddWindow` aufrufen und dann `Window.Open` auf `True` setzen.

62.2 Window.Accelerator

BEZEICHNUNG

`Window.Accelerator` – setzt die Tastaturkürzel-Tabelle des Fensters

BESCHREIBUNG

Setzt die Tastaturkürzel-Tabelle (Accelerator-Tabelle) für dieses Fenster. Siehe [Abschnitt 7.1 \[Accelerator-Klasse\], Seite 61](#), für Details.

TYP

MOAI-Objekt

ANWENDBARKEIT

I

62.3 Window.Activate

BEZEICHNUNG

Window.Activate – ändert den Aktivierungszustand des Fensters

BESCHREIBUNG

Setzen Sie dieses Attribut auf `True`, um das Fenster zu aktivieren.

Sie können auch eine Benachrichtigung für dieses Attribut einrichten, um zu erfahren, wann immer das Fenster aktiviert oder deaktiviert wird.

TYP

Boolesch

ANWENDBARKEIT

ISGN

62.4 Window.ActiveObject

BEZEICHNUNG

Window.ActiveObject – setzt/ermittelt das aktive Widget

BESCHREIBUNG

Setzt oder ermittelt das aktive Widget. Das aktive Widget ist das mit dem Tastaturfokus.

Widgets können auch manuell mit der TAB-Taste aktiviert werden.

TYP

MOAI-Objekt

ANWENDBARKEIT

SG

62.5 Window.Borderless

BEZEICHNUNG

Window.Borderless – erzeugt das Fenster randlos (V2.0)

PLATTFORMEN

Amiga, Android

BESCHREIBUNG

Setzen Sie dies auf `True`, um ein Fenster ohne Rahmen zu erstellen.

TYP

Boolesch

ANWENDBARKEIT

I

62.6 Window.CloseGadget**BEZEICHNUNG**

Window.CloseGadget – konfiguriert das Fensterschließsymbol

BESCHREIBUNG

Setzen Sie diese Option auf `False`, um ein Fenster ohne Fensterschließsymbol zu erstellen.

TYP

Boolesch

ANWENDBARKEIT

I

62.7 Window.CloseRequest**BEZEICHNUNG**

Window.CloseRequest – benachrichtigt, wenn das Fensterschließsymbol gedrückt wurde

BESCHREIBUNG

Wenn Sie eine Benachrichtigung für dieses Attribut einrichten, wird das Fenster nicht automatisch geschlossen, wenn der Benutzer das Fensterschließsymbol drückt. Stattdessen wird Ihre Callback-Funktion aufgerufen und Sie müssen das Fenster manuell schließen, indem Sie `Window.Open` auf `False` setzen. Oder Sie können das Fenster offen lassen, wenn es Dinge gibt, die zuerst erledigt werden müssen.

Ein typischer Anwendungsfall für dieses Attribut ist die Frage, ob der Benutzer das aktuelle Projekt speichern möchte, bevor er das Programm beendet.

Wenn es keine Benachrichtigung für dieses Attribut gibt, werden beim Drücken des Fensterschließsymbols das Fenster automatisch geschlossen.

TYP

Boolesch

ANWENDBARKEIT

N

62.8 Window.DefaultObject**BEZEICHNUNG**

Window.DefaultObject – setzt/ermittelt das Standard-Widget des Fensters

BESCHREIBUNG

Das Standard-Widget ist dasjenige, das beim Öffnen eines Fensters aktiv ist. Typischerweise kann dies ein "OK" oder eine andere Bestätigungsschaltfläche sein, so dass der Benutzer einfach RETURN drücken kann, um das Fenster zu schließen.

TYP

MOAI-Objekt

ANWENDBARKEIT

ISG

62.9 Window.DragBar

BEZEICHNUNG

Window.DragBar – konfiguriert die Zieh-Leiste des Fensters

BESCHREIBUNG

Setzen Sie dies auf `False`, wenn Sie nicht möchten, dass Ihr Fenster gezogen werden kann.

TYP

Boolesch

ANWENDBARKEIT

I

62.10 Window.Height

BEZEICHNUNG

Window.Height – stellt die Fensterhöhe ein oder ermittelt sie

BESCHREIBUNG

Hiermit können Sie die Fensterhöhe in geräteunabhängige Pixel ermitteln oder einstellen. Dies kann entweder ein absoluter Pixelwert oder einer der folgenden Sonderwerte sein:

`Default` Berechnet die Höhe aus den Standardgrößen aller Widgets.

`Screen:<1..100>`

Stellt die Höhe als Prozentsatz der Gesamthöhe des Hostbildschirms ein.

Standard für dieses Tag ist `Default`.

TYP

Zahl oder vordefinierter Sonderwert

ANWENDBARKEIT

IG

62.11 Window.HideFromTaskbar

BEZEICHNUNG

Window.HideFromTaskbar – konfiguriert die Sichtbarkeit der Taskleiste

PLATTFORMEN

Windows, Linux

BESCHREIBUNG

Setzen Sie dies auf `True`, wenn Ihr Fenster nicht in der Taskleiste erscheinen soll.

TYP

Boolesch

ANWENDBARKEIT

I

62.12 Window.Left

BEZEICHNUNG

`Window.Left` – setzt/ermittelt den linken Fensterrand

BESCHREIBUNG

Ermitteln oder stellen Sie die horizontale Position des Fensters ein. Sie können hier einen absoluten Wert in geräteunabhängige Pixel übergeben oder eines der folgenden Sonderwerte verwenden:

`Centered` Fenster im sichtbaren Bereich des Bildschirms zentrieren.

`Moused` Fenster unter dem Mauszeiger öffnen.

Standard für dieses Tag ist `Centered`.

TYP

Zahl oder Sonderwert

ANWENDBARKEIT

IG

62.13 Window.Margin

BEZEICHNUNG

`Window.Margin` – setzt den Fensterrand

BESCHREIBUNG

Legt den Fensterrand in geräteunabhängige Pixel fest. Der Fensterrand ist als der Abstand zwischen dem Rahmen des Fensters und der Wurzelgruppe definiert.

TYP

Zahl

ANWENDBARKEIT

I

62.14 Window.MaximizeGadget

BEZEICHNUNG

Window.MaximizeGadget – konfiguriert das Fenstermaximiersymbol

PLATTFORMEN

Windows, Linux, macOS

BESCHREIBUNG

Setzen Sie dies auf `False`, wenn Sie kein Maximiersymbol für Ihr Fenster haben möchten. Standardmäßig `False`, wenn `Window.Toolwindow` gesetzt ist, ansonsten `True`.

TYP

Boolesch

ANWENDBARKEIT

I

62.15 Window.Menubar

BEZEICHNUNG

Window.Menubar – stellt die Menüleiste des Fensters ein

BESCHREIBUNG

Stellen Sie die Menüleiste für dieses Fenster ein. Siehe [Abschnitt 31.1 \[Menubar-Klasse\]](#), [Seite 185](#), für Details.

TYP

MOAI-Objekt

ANWENDBARKEIT

I

62.16 Window.MinimizeGadget

BEZEICHNUNG

Window.MinimizeGadget – konfiguriert das Fensterminimiersymbol

PLATTFORMEN

Windows, Linux, macOS

BESCHREIBUNG

Setzen Sie dies auf `False`, wenn Sie kein Minimiersymbol für Ihr Fenster haben wollen. Standardmäßig `False`, wenn `Window.Toolwindow` gesetzt ist, ansonsten `True`.

TYP

Boolesch

ANWENDBARKEIT

I

62.17 Window.NoCyclerMenu

BEZEICHNUNG

Window.NoCyclerMenu – fügt kein Fenster-Auswahl-Listefeld-Menü hinzu (V2.0)

PLATTFORMEN

Android

BESCHREIBUNG

Standardmäßig können Sie mit RapaGUI auf Android zwischen einzelnen Fenstern über das Menü wechseln. Wenn Sie dies nicht möchten, setzen Sie dieses Attribut auf `True`. In diesem Fall fügt RapaGUI dem Menü der Anwendung kein Auswahl-Listefeld-Menü (Cycler-Menü) hinzu. Beachten Sie, dass das Auswahl-Listefeld-Menü natürlich nur hinzugefügt wird, wenn mehrere Fenster geöffnet sind.

TYP

Boolesch

ANWENDBARKEIT

I

62.18 Window.Open

BEZEICHNUNG

Window.Open – öffnet/schließt ein Fenster

BESCHREIBUNG

Setzen Sie dieses Attribut, um Ihre Fenster zu öffnen oder zu schließen.

Beachten Sie, dass Fenster, die im XML-Code definiert oder von `moai.CreateApp()` erstellt wurden, werden automatisch geöffnet, es sei denn, Sie setzen explizit `Window.Open` auf `False`, damit sie geschlossen bleiben. Fenster, die mit `moai.CreateObject()` erstellt wurden, werden dagegen nicht automatisch geöffnet. Sie müssen sie zunächst der Fensterliste des Programms hinzufügen, indem Sie die Methode `Application.AddWindow` aufrufen und dann `Window.Open` auf `True` setzen.

TYP

Boolesch

ANWENDBARKEIT

ISG

62.19 Window.Orientation

BEZEICHNUNG

Window.Orientation – setzt/ermittelt die Fensterausrichtung (V2.0)

PLATTFORMEN

Android

BESCHREIBUNG

Setzt oder ermittelt die aktuelle Fensterausrichtung. Dies kann einer der folgenden speziellen Werte sein:

Device Verwendet die aktuelle Geräteausrichtung. Dies ist die Voreinstellung.

Landscape
Benutzt das Querformat.

Portrait Verwendet das Hochformat.

Sie können auch eine Benachrichtigung für dieses Attribut einrichten, um zu erfahren, wenn der Benutzer das Gerät dreht.

TYP

Zeichenkette (siehe oben für mögliche Werte)

ANWENDBARKEIT

ISGN

62.20 Window.Parent**BEZEICHNUNG**

Window.Parent – definiert die übergeordneten Objekte des Fensters

PLATTFORMEN

Windows, Linux, macOS

BESCHREIBUNG

Setzen Sie dies auf den Identifikator eines MOAI-Objekts, das das übergeordnete Element dieses Fensters werden soll. Wenn ein Fenster ein übergeordnetes Objektteil hat, wird das Fenster automatisch auch ausgeblendet, wenn der übergeordnete Objektteil ausgeblendet ist, und es wird auch über dem übergeordneten Objektteil zentriert sein, falls keine explizite Position angegeben wurde.

Beachten Sie, dass Sie die Dialog-Klasse verwenden sollten, wenn Sie möchten, dass das übergeordnete Fenster blockiert wird, während das Element geöffnet ist. Siehe [Abschnitt 16.1 \[Dialog-Klasse\]](#), [Seite 99](#), für Details.

TYP

MOAI-Objekt

ANWENDBARKEIT

I

62.21 Window.PubScreen**BEZEICHNUNG**

Window.PubScreen – setzt/ermittelt/definiert den Bildschirm des Fensters

PLATTFORMEN

Nur AmigaOS und kompatible Betriebssysteme

BESCHREIBUNG

Mit diesem Attribut können Sie den Namen eines öffentlichen Bildschirms angeben, auf dem das Fenster geöffnet werden soll. Bitte verwenden Sie dieses Attribut nur, wenn es wirklich notwendig ist, da normalerweise der Benutzer Ihrer Anwendung derjenige sein sollte, der entscheidet, auf welchem Bildschirm er Ihre Anwendung mit den MUI-Einstellungen ausführen möchte.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

62.22 Window.Remember

BEZEICHNUNG

Window.Remember – merkt sich die Fensterposition und -größe (V2.0)

BESCHREIBUNG

Setzen Sie dieses Attribut auf **True**, wenn sich das Fenster seine Position und Größe merken soll, wenn das Programm neu gestartet wird (oder wenn Dialoge geschlossen und neu geöffnet werden).

Beachten Sie, dass dies nur für Fenster/Dialoge möglich ist, die eine ID haben. Die ID wird verwendet, um die Position und Größe des Fensters zu speichern, um es später wiederherzustellen. Beachten Sie also, dass wenn Sie die ID des Fensters ändern, seine Positions- und Größeneinstellungen verloren gehen (oder sie werden auf ein anderes Fenster übertragen, falls Sie eine zuvor für ein bestimmtes Fenster verwendete ID für ein neues Fenster wiederverwenden).

Beachten Sie auch, dass dieses Attribut auf Amiga-Systemen standardmäßig auf **True** gesetzt ist, da das Standard-MUI-Verhalten darin besteht, sich immer Positionen zu merken. Auf allen anderen Systemen ist es standardmäßig **False** gesetzt.

TYP

Boolesch

ANWENDBARKEIT

I

62.23 Window.ScreenTitle

BEZEICHNUNG

Window.ScreenTitle – setzt/ermittelt den Bildschirmtitel des Fensters

PLATTFORMEN

Nur AmigaOS und kompatible Betriebssysteme

BESCHREIBUNG

Legen Sie den Text fest, der in der Titelleiste des Bildschirms angezeigt wird, wenn das Fenster aktiv ist.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

62.24 Window.SingleMenu

BEZEICHNUNG

Window.SingleMenu – verwendet ein einziges Menü (V2.0)

PLATTFORMEN

Android

BESCHREIBUNG

Wenn dieses Attribut auf `True` gesetzt ist, wird nur das erste Menü der Menüleiste verwendet und seine Elemente werden direkt angezeigt, wenn Sie die Radio-Schaltfläche in der Aktionsleiste drücken. Alle anderen Menüs, die sich möglicherweise in der Menüleiste befinden, werden ignoriert.

TYP

Boolesch

ANWENDBARKEIT

I

62.25 Window.SizeChange

BEZEICHNUNG

Window.SizeChange – benachrichtigt über eine Größenänderung (V2.0)

BESCHREIBUNG

Dieses Attribut löst eine Benachrichtigung aus, wenn der Benutzer die Fenstergröße ändert. Ihr Ereignis-Callback erhält zwei zusätzliche Parameter namens `Width` und `Height`, die die neuen Abmessungen des Fensters enthalten.

TYP

Boolesch

ANWENDBARKEIT

N

62.26 Window.SizeGadget

BEZEICHNUNG

Window.SizeGadget – definiert das Fenstergrößensymbol

BESCHREIBUNG

Setzen Sie dies auf `False`, wenn Sie kein Größensymbol für Ihr Fenster haben möchten.

TYP

Boolesch

ANWENDBARKEIT

I

62.27 Window.StayOnTop

BEZEICHNUNG

Window.StayOnTop – öffnet ein Fenster, das zu vorderst bleibt

PLATTFORMEN

Windows, Linux, macOS

BESCHREIBUNG

Setzen Sie dies auf `True`, damit Ihr Fenster oben bei der Z-Reihenfolge der Fenster bleibt.

TYP

Boolesch

ANWENDBARKEIT

I

62.28 Window.Subtitle

BEZEICHNUNG

Window.Subtitle – setzt/ermittelt den Untertitel des Fensters (V2.0)

PLATTFORMEN

Android

BESCHREIBUNG

Legt den Text fest, der als Untertitel der Aktionsleiste angezeigt wird.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

62.29 Window.Title

BEZEICHNUNG

Window.Title – setzt/ermittelt den Titel des Fensters

BESCHREIBUNG

Setzt oder ermittelt den Titel des Fensters.

Wenn Sie dieses Attribut nicht festlegen, verwendet RapaGUI den Titel, der in Hollywoods Präprozessor-Anweisung `@APPTITLE` angegeben wurde. Wenn `@APPTITLE` ebenfalls nicht angegeben wurde, wird der Standardtitel "RapaGUI" verwendet.

TYP

Zeichenkette

ANWENDBARKEIT

ISG

62.30 Window.Toolwindow**BEZEICHNUNG**

Window.Toolwindow – kennzeichnet das Fenster als Werkzeugfenster

PLATTFORMEN

Windows, Linux, macOS

BESCHREIBUNG

Setzen Sie dies auf `True`, um Ihr Fenster im Werkzeugfensterdesign zu öffnen. Werkzeugfenster haben in der Regel einen kleineren Rahmen als normale Top-Level-Fenster.

TYP

Boolesch

ANWENDBARKEIT

I

62.31 Window.Top**BEZEICHNUNG**

Window.Top – setzt/ermittelt die Fensteroberkante

BESCHREIBUNG

Stellen Sie die vertikale Position des Fensters ein. Sie können hier einen absoluten Wert in geräteunabhängige Pixel übergeben oder eines der folgenden Sonderwerte verwenden:

Centered Fenster im sichtbaren Bereich des Bildschirms zentrieren.

Moused Fenster unter dem Mauszeiger öffnen.

Delta:<p>

Öffnen Sie das Fenster <p> geräteunabhängige Pixel unterhalb der Titelleiste des Bildschirms.

Der Standardwert für dieses Tag ist **Centered**.

TYP

Zahl oder vordefinierte Sonderwerte

ANWENDBARKEIT

IG

62.32 Window.UseBottomBorderScroller

BEZEICHNUNG

Window.UseBottomBorderScroller – aktiviert die Bildlaufleiste am unteren Rand

PLATTFORMEN

Nur AmigaOS und kompatible Betriebssysteme

BESCHREIBUNG

Setzen Sie dies auf `True`, um RapaGUI mitzuteilen, dass sich in Ihrem Fensterlayout ein Element befindet, das eine Bildlaufleiste im unteren Fensterrand platziert, z.B. mit `Scrollbar.UseWinBorder`.

TYP

Boolesch

ANWENDBARKEIT

I

62.33 Window.UseLeftBorderScroller

BEZEICHNUNG

Window.UseLeftBorderScroller – aktiviert die Bildlaufleiste am linken Rand

PLATTFORMEN

Nur AmigaOS und kompatible Betriebssysteme

BESCHREIBUNG

Setzen Sie dies auf `True`, um RapaGUI mitzuteilen, dass sich in Ihrem Fensterlayout ein Element befindet, das eine Bildlaufleiste im linken Fensterrand platziert, z.B. mit `Scrollbar.UseWinBorder`.

TYP

Boolesch

ANWENDBARKEIT

I

62.34 Window.UseRightBorderScroller

BEZEICHNUNG

Window.UseRightBorderScroller – aktiviert die Bildlaufleiste am rechten Rand

PLATTFORMEN

Nur AmigaOS und kompatible Betriebssysteme

BESCHREIBUNG

Setzen Sie dies auf `True`, um RapaGUI mitzuteilen, dass es in Ihrem Fensterlayout ein Element gibt, das eine Bildlaufleiste in den rechten Fensterrand setzt, z.B. mit `Scrollbar.UseWinBorder`.

TYP

Boolesch

ANWENDBARKEIT

I

62.35 Window.Width

BEZEICHNUNG

Window.Width – setzt/ermittelt die Fensterbreite

BESCHREIBUNG

Legen Sie die Fensterbreite in geräteunabhängige Pixel fest. Dies kann entweder ein absoluter Pixelwert oder einer der folgenden Sonderwerte sein:

Default Berechnet die Breite aus den Standardgrößen aller Widgets.

Screen:<1..100>

Legt die Breite als Prozentsatz der Gesamtbreite des Hostbildschirms fest.

Standard für dieses Tag ist **Default**.

TYP

Zahl oder vordefinierte Sonderwerte

ANWENDBARKEIT

IG

Anhang A Lizenzen

A.1 wxWidgets license

wxWindows Library Licence, Version 3.1

Copyright (c) 1998-2005 Julian Smart, Robert Roebing et al

Everyone is permitted to copy and distribute verbatim copies of this licence document, but changing it is not allowed.

WXWINDOWS LIBRARY LICENCE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public Licence as published by the Free Software Foundation; either version 2 of the Licence, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but **WITHOUT ANY WARRANTY**; without even the implied warranty of **MERCHANTABILITY** or **FITNESS FOR A PARTICULAR PURPOSE**. See the GNU Library General Public Licence for more details.

You should have received a copy of the GNU Library General Public Licence along with this software, usually in a file named **COPYING.LIB**. If not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

EXCEPTION NOTICE

1. As a special exception, the copyright holders of this library give permission for additional uses of the text contained in this release of the library as licenced under the wxWindows Library Licence, applying either version 3.1 of the Licence, or (at your option) any later version of the Licence as published by the copyright holders of version 3.1 of the Licence document.
2. The exception is that you may use, copy, link, modify and distribute under your own terms, binary object code versions of works based on the Library.
3. If you copy code from files distributed under the terms of the GNU General Public Licence or the GNU Library General Public Licence into a copy of this library, as this licence permits, the exception does not apply to the code that you add in this way. To avoid misleading anyone as to the status of such modified files, you must delete this exception notice from such code and/or adjust the licensing conditions notice accordingly.
4. If you write modifications of your own for this library, it is your choice whether to permit this exception to apply to your modifications. If you do not wish that, you must delete the exception notice from such code and/or adjust the licensing conditions notice accordingly.

A.2 MUI license

This application uses MUI - MagicUserInterface (c) Copyright 1992-97 by Stefan Stuntz. MUI is a system to generate and maintain graphical user interfaces. With the aid of a preferences program, the user of an application has the ability to customize the outfit according to his personal taste.

MUI is distributed as shareware. To obtain a complete package containing lots of examples and more information about registration please look for a file called "muiXXusr.lha" (XX means the latest version number) on your local bulletin boards or on public domain disks.

If you want to pageview directly, feel free to send DM 30.- or US\$ 20.- to

Stefan Stuntz
Eduard-Spranger-Straße 7
80935 München
GERMANY

Support and online registration is available at <http://www.sasg.com/>

A.3 Expat license

Copyright (c) 1998, 1999, 2000 Thai Open Source Software Center Ltd and Clark Cooper
Copyright (c) 2001, 2002, 2003, 2004, 2005, 2006 Expat maintainers.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

A.4 LGPL license

GNU Lesser General Public License Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc. 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages—typically libraries—of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library

does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- a) The modified work must itself be a software library.
- b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.
- c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.
- d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which

must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

- b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.
- c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
- d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
- e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

- a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
- b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Index

A

Acceleratoritem.Mod	63
Acceleratoritem.Pressed	64
Application.AboutMUI	65
Application.AboutRapaGUI	65
Application.AddWindow	65
Application.ContextMenu	66
Application.HelpFile	66
Application.Icon	67
Application.OpenConfigWindow	67
Application.RemoveWindow	68
Application.ShowHelp	68
Application.ShowHelpNode	69
Application.Sleep	69
Application.UseIcons	69
Application.WindowMenu	70
Area.ContextMenu	71
Area.Disabled	72
Area.FixHeight	72
Area.FixWidth	73
Area.FontName	73
Area.FontSize	73
Area.FontStyle	74
Area.Height	75
Area.Hide	75
Area.Left	75
Area.NoAutoKey	76
Area.Redraw	76
Area.Tooltip	76
Area.Top	77
Area.Weight	77
Area.Width	77

B

Busybar.Move	79
Busybar.Reset	79
Button.Hint	81
Button.Icon	81
Button.IconPos	82
Button.IconScale	82
Button.IconType	83
Button.Pressed	83
Button.Selected	84
Button.Text	84
Button.Toggle	84

C

Checkbox.Right	87
Checkbox.Selected	87
Choice.Active	89
Choice.Clear	89
Choice.Count	90
Choice.GetEntry	90
Choice.Insert	90
Choice.Remove	91
Choice.Rename	92
Combobox.Acknowledge	93
Combobox.Clear	93
Combobox.Close	94
Combobox.Count	94
Combobox.GetEntry	94
Combobox.Insert	95
Combobox.Open	95
Combobox.Popup	95
Combobox.Remove	96
Combobox.Rename	96
Combobox.Selected	96
Combobox.Value	97

D

Dialog.EndModal	100
Dialog.ShowModal	100

F

Finddialog.Down	103
Finddialog.Find	103
Finddialog.FindNext	103
Finddialog.FindString	104
Finddialog.MatchCase	104
Finddialog.NoMatchCase	104
Finddialog.NoUpDown	105
Finddialog.NoWholeWord	105
Finddialog.Replace	105
Finddialog.ReplaceAll	106
Finddialog.ReplaceMode	106
Finddialog.ReplaceString	106
Finddialog.WholeWord	107

G

Group.Append	110
Group.Color	110
Group.Columns	111
Group.ExitChange	111
Group.Frame	112
Group.FrameTitle	112
Group.HAlign	112
Group.Hide	113
Group.HorizSpacing	113
Group.Icon	113
Group.IconScale	114
Group.IconType	115
Group.InitChange	115
Group.Insert	116
Group.Padding	116
Group.Paint	117
Group.Prepend	118
Group.Remove	119
Group.SameSize	119
Group.Spacing	120
Group.Title	120
Group.VAlign	121
Group.VertSpacing	121
Group.Weight	121

H

Hollywood.Display	125
Hollywood.DropFile	126
Hollywood.DropTarget	126
HSpace.Width	127
HSplitter.Border	129
HSplitter.Gravity	129
HSplitter.MinPaneSize	130
HSplitter.Position	130
HSplitter.Split	130
HSplitter.Unsplit	131
HTMLview.CanGoBack	133
HTMLview.CanGoForward	133
HTMLview.ClearHistory	133
HTMLview.Contents	134
HTMLview.File	134
HTMLview.GoBack	134
HTMLview.GoForward	135
HTMLview.Reload	135
HTMLview.Search	135
HTMLview.Title	136
HTMLview.URL	136
Hyperlink.Label	139
Hyperlink.URL	139

I

Image.Brush	141
Image.BrushScale	141
Image.BrushType	142

L

Label.Align	143
Label.Text	143
Listview.AbortEditing	146
Listview.Active	146
Listview.Alternate	147
Listview.Clear	147
Listview.ClickColumn	148
Listview.Columns	148
Listview.CompareItems	148
Listview.DefClickColumn	149
Listview.DoubleClick	149
Listview.DropFile	150
Listview.DropTarget	150
Listview.Edit	150
Listview.Entries	151
Listview.Exchange	151
Listview.First	152
Listview.ForceMode	152
Listview.GetColumnID	153
Listview.GetDisabled	154
Listview.GetEntry	154
Listview.GetSelection	155
Listview.GetState	155
Listview.HRules	156
Listview.Insert	156
Listview.InsertColumn	157
Listview.ItemStyle	159
Listview.Jump	159
Listview.LongClick	160
Listview.Move	160
Listview.MultiSelect	161
Listview.Quiet	161
Listview.Remove	161
Listview.RemoveColumn	162
Listview.Rename	162
Listview.RowHeight	163
Listview.Select	164
Listview.SetDisabled	164
Listview.SetState	165
Listview.Sort	165
Listview.SortColumn	166
Listview.StartEditing	166
Listview.SystemTheme	167
Listview.TitleClick	167
Listview.ValueChange	167
Listview.Visible	168
Listview.VRules	168
Listviewcolumn.Align	171
Listviewcolumn.Checkbox	171
Listviewcolumn.Editable	172
Listviewcolumn.Hide	173
Listviewcolumn.Icon	173
Listviewcolumn.IconScale	174
Listviewcolumn.IconType	174
Listviewcolumn.Sortable	175
Listviewcolumn.SortOrder	175
Listviewcolumn.Title	176

Listviewcolumn.Width 176
 Listviewitem.Icon 177

M

Menu.Append 179
 Menu.Disabled 180
 Menu.Insert 180
 Menu.NoAutoKey 180
 Menu.Prepend 181
 Menu.Remove 181
 Menu.Title 182
 Menu.Type 182
 Menubar.Append 186
 Menubar.Insert 186
 Menubar.Prepend 187
 Menubar.Remove 187
 MenuItem.Disabled 189
 MenuItem.Help 189
 MenuItem.NoAutoKey 190
 MenuItem.Selected 190
 MenuItem.Shortcut 190
 MenuItem.Title 192
 MenuItem.Type 193
 MOAI.Class 195
 moai.CreateApp 47
 moai.CreateDialog 48
 moai.CreateObject 49
 moai.DoMethod 51
 moai.FreeApp 52
 moai.FreeDialog 52
 moai.FreeImage 53
 moai.FreeObject 54
 moai.Get 54
 moai.HaveObject 55
 MOAI.I18N 195
 MOAI.ID 195
 MOAI.NoNotify 196
 moai.Notify 55
 MOAI.Notify 196
 MOAI.NotifyData 197
 moai.Request 56
 moai.Set 57
 moai.UpdateImage 58
 MOAI.UserData 197

P

Pageview.Active 199
 Pageview.Append 200
 Pageview.GetPageID 200
 Pageview.Insert 201
 Pageview.Mode 202
 Pageview.Multiline 202
 Pageview.Pages 203
 Pageview.PlainBG 203
 Pageview.Position 203
 Pageview.Prepend 204

Pageview.Remove 204
 Popcolor.RGB 207
 Popcolor.Title 207
 Popfile.File 209
 Popfile.Pattern 209
 Popfile.SaveMode 209
 Popfile.Title 210
 Popfont.Font 211
 Popfont.MaxSize 211
 Popfont.MinSize 211
 Popfont.Title 212
 Poppath.Path 213
 Poppath.Title 213
 Progressbar.Horiz 215
 Progressbar.Level 215
 Progressbar.Max 215

R

Radio.Active 217
 Radio.Columns 217
 Radio.GetItem 218
 Radio.SetItem 218
 Radio.Title 219

S

Scrollbar.AutoScale 223
 Scrollbar.Horiz 223
 Scrollbar.Level 224
 Scrollbar.Range 224
 Scrollbar.StepSize 224
 Scrollbar.Target 225
 Scrollbar.UseWinBorder 225
 Scrollbar.Visible 226
 Scrollcanvas.AutoBars 227
 Scrollcanvas.AutoScale 228
 Scrollcanvas.Paint 228
 Scrollcanvas.Scroll 229
 Scrollcanvas.StepSize 230
 Scrollcanvas.UseLeftBorder 230
 Scrollcanvas.UseWinBorder 231
 Scrollcanvas.VirtHeight 231
 Scrollcanvas.VirtWidth 232
 Scrollgroup.AutoBars 233
 Scrollgroup.Horiz 233
 Scrollgroup.UseWinBorder 234
 Slider.Drag 235
 Slider.Horiz 235
 Slider.Level 235
 Slider.Max 236
 Slider.Min 236
 Slider.Quiet 236
 Slider.Release 237
 Slider.Reverse 237
 Slider.StepSize 237
 Statusbaritem.Text 241
 Statusbaritem.Width 241

T

Text.Align	243
Text.Frame	243
Text.Text	244
Texteditor.Align	245
Texteditor.AreaMarked	245
Texteditor.Bold	246
Texteditor.Clear	246
Texteditor.Color	246
Texteditor.Copy	247
Texteditor.CursorPos	247
Texteditor.Cut	248
Texteditor.GetLineLength	248
Texteditor.GetPosition	248
Texteditor.GetSelection	249
Texteditor.GetText	249
Texteditor.GetXY	249
Texteditor.HasChanged	250
Texteditor.Hint	250
Texteditor.Insert	251
Texteditor.Italic	251
Texteditor.Mark	251
Texteditor.MarkAll	252
Texteditor.MarkNone	252
Texteditor.NoWrap	252
Texteditor.Paste	253
Texteditor.ReadOnly	253
Texteditor.Redo	253
Texteditor.RedoAvailable	254
Texteditor.ScrollToLine	254
Texteditor.SetBold	254
Texteditor.SetColor	255
Texteditor.SetItalic	255
Texteditor.SetUnderline	256
Texteditor.Styled	256
Texteditor.Text	257
Texteditor.Underline	257
Texteditor.Undo	258
Texteditor.UndoAvailable	258
Textentry.Accept	259
Textentry.Acknowledge	259
Textentry.AdvanceOnCR	260
Textentry.Copy	260
Textentry.CursorPos	260
Textentry.Cut	261
Textentry.GetSelection	261
Textentry.Hint	261
Textentry.Insert	262
Textentry.Mark	262
Textentry.MarkAll	262
Textentry.MarkNone	263
Textentry.MaxLen	263
Textentry.Password	263
Textentry.Paste	264
Textentry.ReadOnly	264
Textentry.Redo	264
Textentry.Reject	265
Textentry.Text	265
Textentry.Undo	265
Textview.Align	267
Textview.Append	267
Textview.Styled	268
Textview.Text	268
Toolbar.Horiz	269
Toolbar.ViewMode	270
Toolbarbutton.Disabled	271
Toolbarbutton.Help	271
Toolbarbutton.Icon	271
Toolbarbutton.IconScale	272
Toolbarbutton.IconType	272
Toolbarbutton.Pressed	273
Toolbarbutton.Selected	274
Toolbarbutton.Tooltip	274
Toolbarbutton.Type	274
TreeView.AbortEditing	278
TreeView.Active	279
TreeView.Alternate	280
TreeView.Clear	280
TreeView.ClickColumn	280
TreeView.Close	280
TreeView.DoubleClick	281
TreeView.DropFile	281
TreeView.DropTarget	282
TreeView.EditableNodes	282
TreeView.ForceMode	283
TreeView.GetEntry	284
TreeView.HRules	285
TreeView.InsertLeaf	286
TreeView.InsertNode	287
TreeView.Open	288
TreeView.Remove	289
TreeView.StartEditing	289
TreeView.ValueChange	290
TreeView.VRules	290
TreeViewcolumn.Align	293
TreeViewcolumn.Checkbox	293
TreeViewcolumn.Editable	294
TreeViewcolumn.Hide	295
TreeViewcolumn.Icon	295
TreeViewcolumn.IconScale	296
TreeViewcolumn.IconType	296
TreeViewcolumn.Title	297
TreeViewcolumn.Width	297
TreeViewleaf.Edit	299
TreeViewleaf.GetDisabled	299
TreeViewleaf.GetIcon	300
TreeViewleaf.GetItem	300
TreeViewleaf.GetState	301
TreeViewleaf.SetDisabled	301
TreeViewleaf.SetIcon	301
TreeViewleaf.SetItem	302
TreeViewleaf.SetState	302
TreeViewleaf.UID	303
TreeViewleafitem.Icon	305
TreeViewnode.Edit	307
TreeViewnode.Icon	307

TreeViewnode.Name 308
TreeViewnode.UID 308

V

VSpace.Height 313
VSplitter.Border 315
VSplitter.Gravity 315
VSplitter.MinPaneSize 316
VSplitter.Position 316
VSplitter.Split 316
VSplitter.Unsplit 317

W

Window.Accelerator 319
Window.Activate 320
Window.ActiveObject 320
Window.Borderless 320
Window.CloseGadget 321
Window.CloseRequest 321
Window.DefaultObject 321
Window.DragBar 322
Window.Height 322

Window.HideFromTaskbar 322
Window.Left 323
Window.Margin 323
Window.MaximizeGadget 323
Window.Menubar 324
Window.MinimizeGadget 324
Window.NoCyclerMenu 324
Window.Open 325
Window.Orientation 325
Window.Parent 326
Window.PubScreen 326
Window.Remember 327
Window.ScreenTitle 327
Window.SingleMenu 328
Window.SizeChange 328
Window.SizeGadget 328
Window.StayOnTop 329
Window.Subtitle 329
Window.Title 329
Window.Toolwindow 330
Window.Top 330
Window.UseBottomBorderScroller 330
Window.UseLeftBorderScroller 331
Window.UseRightBorderScroller 331
Window.Width 332

