Natural

Dialog Component Reference

Version 8.3.8

January 2017

ADABAS & NATURAL

This document applies to Natural Version 8.3.8.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 1992-2017 Software AG, Darmstadt, Germany and/or Software AG USA, Inc., Reston, VA, USA, and/or its subsidiaries and/or its affiliates and/or their licensors.

The name Software AG and all Software AG product names are either trademarks or registered trademarks of Software AG and/or Software AG USA, Inc. and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

Detailed information on trademarks and patents owned by Software AG and/or its subsidiaries is located at http://softwareag.com/licenses.

Use of this software is subject to adherence to Software AG's licensing conditions and terms. These terms are part of the product documentation, located at http://softwareag.com/licenses/ and/or in the root installation directory of the licensed product(s).

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third-Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software AG Products / Copyright and Trademark Notices of Software AG Products". These documents are part of the product documentation, located at http://softwareag.com/licenses and/or in the root installation directory of the licensed product(s).

Use, reproduction, transfer, publication or disclosure is prohibited except as specifically provided for in your License Agreement with Software AG.

Document ID: NATWIN-DCM-838-20170102

Table of Contents

Dialog Component Reference xxv
I Dialogs and Dialog Elements 1
1 General Information 3
2 ActiveX Controls 5
Description 6
Natural Attributes for ActiveX Controls 6
Events 7
3 Bitmap Control 9
Description 10
Attributes for Bitmap Control 10
Events 11
4 Canvas Control 13
Description 14
Attributes for Canvas Control 14
Events 15
5 Column Specification Control 17
Description 18
Attributes for Column Specification Control 18
COLUMN-TYPEs and their Attributes 19
Events 19
6 Control Box Control 21
Description 22
Attributes for Control Box Control 22
Events 23
7 Context Menu 25
Description 26
Attributes for Context Menu 26
Events 27
8 Date and Time Picker (DTP) Control 29
Description 30
Attributes for Date and Time Picker Control 30
Events 31
9 Dialog Bar Control 33
Description 34
Attributes for Dialog Bar Control 34
Events 35
10 Edit Area Control 37
Description 38
Attributes for Edit Area Control 38
Events 39
11 Font Control 41
Description 42
Attributes for Font Control 42

Events 43
12 Graphic Text Control 45
Description 46
Attributes for Graphic Text Control 46
Events 47
13 Group Frame Control 49
Description 50
Attributes for Group Frame Control 50
Events 51
14 GUI Control 53
Description 54
Attributes for GUI Control 54
15 Image Control 61
Description 62
Attributes for Image Control 62
Events 62
16 Image List Control 63
Description 64
Attributes for Image List Control 64
Events 64
17 Input Field Control 65
Description 66
Attributes for Input Field Control 66
Events 68
18 List Box Control 69
Description 70
Attributes for List Box Control 71
Events 72
19 List Box Item 73
Description 74
Attributes for List Box Item 74
Events 75
20 List View Column 77
Description 78
Attributes for List View Column 78
Events 79
21 List View Control 81
Description 82
..... 82
Events 84
22 List View Item 85
Description 86
Attributes for List View Item 86
Events 87
23 Line Control 89

Description 90
Attributes for Line Control 90
Events 90
24 MDI Child Window 91
Description 92
Attributes for MDI Child Window 92
Events 95
25 MDI Frame Window 97
Description 98
Attributes for MDI Frame Window 98
Events 100
26 MDI Plug-in Window 101
Description 102
Events 104
27 Menu Bar 105
Description 106
Attributes for Menu Bar 106
Events 107
28 Menu Item 109
Description 110
Attributes for Menu Item 110
Events 111
29 OLE Container Control 113
Description 114
Attributes for OLE Container Control 114
Events 115
30 Progress Bar Control 117
Description 118
Attributes for Progress Bar Control 118
Events 119
31 Push Button Control 121
Description 122
Attributes for Push Button Control 122
Events 123
32 Radio Button Control 125
Description 126
Attributes for Radio Button Control 126
Events 127
33 Rectangle Control 129
Description 130
Attributes for Rectangle Control 130
Events 131
34 Scrollbar Control 133
Description 134
Attributes for Scrollbar Control 134

Events 135
35 Selection Box Control 137
Description 138
Attributes for Selection Box Control 139
Events 140
36 Selection Box Item 141
Description 142
Attributes for Selection Box Item 142
Events 143
37 Signal 145
Description 146
Attributes for Signal 146
Events 147
38 Slider Control 149
Description 150
Attributes for Slider Control 150
Events 151
39 Spin Control 153
Description 154
..... 154
Events 155
40 Status Bar Control 157
Description 158
Attributes for Status Bar Control 158
Events 159
41 Status Bar Pane 161
Description 162
Attributes for Status Bar Pane 162
Events 163
42 Standard Window 165
Description 166
Attributes for Standard Window 166
Events 169
43 Submenu Control 171
Description 172
Attributes for Submenu Control 172
Events 173
44 Tab Control 175
Description 176
Attributes for Tab Control 176
Events 177
45 Tab Control Tab 179
Description 180
Attributes for Tab Control Tab 180
Events 180

46 Table Control 181
Description 182
Attributes for Table Control 182
Attributes for Cells in a Table Control 184
COLUMN-TYPEs in Cells and their Attributes 184
Events 185
47 Text Constant Control 187
Description 188
Attributes for Text Constant Control 188
Events 189
48 Timer Control 191
Description 192
Attributes for Timer 192
Events 193
49 Toggle Button Control 195
Description 196
Attributes for Toggle Button Control 196
Events 197
50 Tool Bar 199
Description 200
Attributes for Tool Bar 200
Events 201
51 Tool Bar Control 203
Description 204
Attributes for Tool Bar Control 204
Events 205
52 Tool Bar Item 207
Description 208
Attributes for Tool Bar Item 208
Events 209
53 Tree View Control 211
Description 212
Attributes for Tree View Control 212
Events 214
54 Tree View Item 215
Description 216
Attributes for Tree View Item 216
Events 217
55 Wallpaper Control 219
Description 220
Attributes for Wallpaper Control 220
Events 221
II Attributes 223
56 ACCELERATOR 229
57 ACTIVE-CHILD 231

58 AUTO-ADJUST 233
59 AUTOSELECT 235
60 BACKGROUND-COLOUR-NAME 237
61 BACKGROUND-COLOUR-VALUE 239
62 BAR-ID 241
63 BITMAP-FILE-NAME 243
64 BLEND 245
65 BUDDY 247
66 CELL-ATTRIBUTES 249
67 CHECKED 251
68 CHECKED-SUCCESSOR 253
69 CLIENT-DATA 255
70 CLIENT-HANDLE 257
71 CLIENT-KEY 259
72 CLIENT-VALUE 261
73 COLUMN 263
74 COLUMN-COUNT 265
75 COLUMN-TYPE 267
76 COMPATIBILITY 269
77 CONTEXT MENU 271
78 CONTROL 273
79 DEFAULT-BUTTON 275
80 DESCENDING 277
81 DIL-TEXT 279
82 DOCKING 281
83 DPI 283
84 DRAG-MODE 285
85 DRAGGABLE 287
86 DROP-MODE 289
87 EDIT-MASK 291
88 EMBEDDED-OBJECT 293
89 ENABLED 295
90 EVENT-QUEUEING 297
91 EXPANDED 299
92 FIRST-CHILD 301
93 FIRST-COLUMN-WIDTH 303
94 FIRST-VISIBLE-COLUMN 305
95 FIRST-VISIBLE-ITEM 307
96 FIRST-VISIBLE-ROW 309
97 FOLLOWS 311
98 FONT-HANDLE 313
99 FONT-STRING 315
100 FOREGROUND-COLOUR-NAME 317
101 FOREGROUND-COLOUR-VALUE 319
102 FORMAT 321

103 FROZEN-COLUMNS 323
104 GROUP-ID 325
105 HANDLE-VARIABLE 327
106 HAS-DIL 329
107 HAS-FIRST-COLUMN 331
108 HAS-HELP-BUTTON 333
109 HAS-MENU-BAR 335
110 HAS-STATUS-BAR 337
111 HAS-SYSTEM-BUTTON 339
112 HAS-TOOLBAR 341
113 HAS-TOOLTIP 343
114 HEADER-FONT-HANDLE 345
115 HEADER-HEIGHT 347
116 HELP-FILENAME 349
117 HELP-ID 351
118 HORIZ-SCROLLABLE 353
119 ICONIZED 355
120 IMAGE 357
121 IMAGE-INDEX 359
122 IMAGE-LIST 361
123 INPLACE-ACTIVE 363
124 ITEM 365
125 ITEM-H 367
126 ITEM-W 369
127 LAST-CHILD 371
128 LENGTH 373
129 LINE 375
130 LINKED 377
131 LOCATION 379
132 MARGIN-X 381
133 MARGIN-Y 383
134 MAX 385
135 MAXIMIZABLE 387
136 MAXIMIZED 389
137 MENU-HANDLE 391
138 MENU-ITEM-OLE 393
139 MENU-ITEM-TYPE 395
140 MIN 397
141 MINIMIZABLE 399
142 MINIMIZED 401
143 MODIFIED-SUCCESSOR 403
144 MODIFIABLE 405
145 MODIFIED 407
146 MULTI-SELECTION 409
147 NAME 411

148 OBJECT-SIZE 413
149 OFFSET-X 415
150 OFFSET-Y 417
151 OVERLAY 419
152 OVERLAY-INDEX 421
153 OWNER 423
154 P1-X 425
155 P1-Y 427
156 P2-X 429
157 P2-Y 431
158 PAGE 433
159 PARENT 435
160 POSITION 437
161 POPUP-HELP 439
162 PREDECESSOR 441
163 RECTANGLE-H 443
164 RECTANGLE-W 445
165 RECTANGLE-X 447
166 RECTANGLE-Y 449
167 ROW 451
168 ROW-COUNT 453
169 ROW-HEIGHT 455
170 RTL 457
171 SAME-AS 459
172 SCROLLRANGE-X 461
173 SCROLLRANGE-Y 463
174 SELECTED-SUCCESSOR 465
175 SELECTED 467
176 SHARED 469
177 SIZE-MODIFIABLE 471
178 SLIDER 473
179 SORTED 475
180 SERVER-OBJECT 477
181 SERVER-PROGID 479
182 SPACING 481
183 SPACING-X 483
184 SPACING-Y 485
185 STATUS-HANDLE 487
186 STATUS-TEXT 489
187 STRING 491
188 STYLE 493
189 SUCCESSOR 507
190 SUPPRESS-AFTER-EDIT-EVENT 509
191 SUPPRESS-BEFORE-EDIT-EVENT 511
192 SUPPRESS-BEFORE-OPEN-EVENT 513

193 SUPPRESS-BEGIN-DRAG-EVENT 515
194 SUPPRESS-CLIENT-SIZE-EVENT 517
195 SUPPRESS-DBL-CLICK-EVENT 519
196 SUPPRESS-DELETE-ROW-EVENT 521
197 SUPPRESS-INSERT-ROW-EVENT 523
198 SUPPRESS-TOP-EVENT 525
199 SUPPRESS-ACTIVATE-EVENT 527
200 SUPPRESS-CHANGE-EVENT 529
201 SUPPRESS-CHECK-EVENT 531
202 SUPPRESS-CLIPBOARD-STATUS-EVENT 533
203 SUPPRESS-CLICK-EVENT 535
204 SUPPRESS-CLOSE-EVENT 537
205 SUPPRESS-COLLAPSE-EVENT 539
206 SUPPRESS-COMMAND-STATUS-EVENT 541
207 SUPPRESS-CONTEXT-MENU-EVENT 543
208 SUPPRESS-COPY-EVENT 545
209 SUPPRESS-CUT-EVENT 547
210 SUPPRESS-DELETE-EVENT 549
211 SUPPRESS-DRAG-DROP-EVENT 551
212 SUPPRESS-DRAG-ENTER-EVENT 553
213 SUPPRESS-DRAG-LEAVE-EVENT 555
214 SUPPRESS-DRAG-OVER-EVENT 557
215 SUPPRESS-END-DRAG-EVENT 559
216 SUPPRESS-ENTER-CELL-EVENT 561
217 SUPPRESS-EXPAND-EVENT 563
218 SUPPRESS-ENTER-EVENT 565
219 SUPPRESS-FILL-EVENT 567
220 SUPPRESS-IDLE-EVENT 569
221 SUPPRESS-LEAVE-CELL-EVENT 571
222 SUPPRESS-LEAVE-EVENT 573
223 SUPPRESS-PASTE-EVENT 575
224 SUPPRESS-SIZE-EVENT 577
225 SUPPRESS-UNDO-EVENT 579
226 TIME 581
227 TIMER-INTERVAL 583
228 TOOLBAR-HANDLE 585
229 TOOLBAR-POS 587
230 TOOLTIP 589
231 TYPE 591
232 VARIABLE 593
233 VERSION 595
234 VERT-SCROLLABLE 597
235 VIEW-MODE 599
236 VISIBLE 601
237 WALLPAPER 603

238 ZOOM-FACTOR 605
III Events 607
239 Activate Event 609
Applies To 610
Description 610
240 After-Any Event 611
Applies To 612
Description 612
241 After-Edit Event 613
Applies To 614
Description 614
242 After-Open Event 615
Applies To 616
Description 616
243 Before-Any Event 617
Applies To 618
Description 618
244 Before-Edit Event 619
Applies To 620
Description 620
245 Before-Open Event 621
Applies To 622
Description 622
246 Begin-Drag Event 623
Applies To 624
Description 624
247 Change Event 625
Applies To 626
Description 626
248 Check Event 627
Applies To 628
Description 628
249 Click Event 629
Applies To 630
Description 630
250 Client-Size Event 631
Applies To 632
Description 632
251 Clipboard-Status Event 633
Applies To 634
Description 634
252 Close Event 635
Applies To 636
Description 636
253 Collapse Event 637

Applies To 638
Description 638
254 Command-Status Event 639
Applies To 640
Description 640
255 Context-Menu Event 643
Applies To 644
Description 644
256 Copy Event 645
Applies To 646
Description 646
257 Cut Event 647
Applies To 648
Description 648
258 DDE-Client Event 649
Applies To 650
Description 650
259 DDE-Server Event 651
Applies To 652
Description 652
260 Default Event 653
Applies To 654
Description 654
261 Delete Event 655
Applies To 656
Description 656
262 Delete-Row Event 657
Applies To 658
Description 658
263 Double-Click Event 659
Applies To 660
Description 660
264 Drag-Drop Event 661
Applies To 662
Description 662
265 Drag-Enter Event 663
Applies To 664
Description 664
266 Drag-Leave Event 665
Applies To 666
Description 666
267 Drag-Over Event 667
Applies To 668
Description 668
268 End-Drag Event 669

Applies To 670
Description 670
269 Enter-Cell Event 671
Applies To 672
Description 672
270 Enter Event 673
Applies To 674
Description 674
271 Error Event 675
Applies To 676
Description 676
272 Expand Event 677
Applies To 678
Description 678
273 Fill Event 679
Applies To 680
Description 680
274 Idle Event 681
Applies To 682
Description 682
275 Insert-Row Event 683
Applies To 684
Description 684
276 Leave-Cell Event 685
Applies To 686
Description 686
277 Leave Event 687
Applies To 688
Description 688
278 Paste Event 689
Applies To 690
Description 690
279 Size Event 691
Applies To 692
Description 692
280 Top Event 693
Applies To 694
Description 694
281 Undo Event 695
Applies To 696
Description 696
282 User-Defined Events 697
Apply To 698
Description 698
IV PROCESS GUI Statement Actions 699

283 General Information 703
284 ADD Action 705
Description 706
Parameters for the ADD WITH option 706
285 ADD-ITEMS Action 709
Description 710
Parameters 710
286 ADD-ITEMS-EX Action 711
Description 712
Parameters 712
287 ARRANGE Action 715
Description 716
Parameters 716
288 BEEP Action 717
Description 718
Parameters 718
289 CALL-DIALOG Action 719
Description 720
Parameters 720
290 CLEAR Action 721
Description 722
Parameters 722
291 CLEAR-TICKS Action 723
Description 724
Parameters 724
292 CLOSE-CLIPBOARD Action 725
Description 726
Parameters 726
293 DELETE-CHILDREN Action 727
Description 728
Parameters 728
294 DELETE-WINDOW Action 729
Description 730
Parameters 730
295 DELETE Action 731
Description 732
Parameters 732
296 DELETE-SUBITEM-DATA Action 733
Description 734
Parameters 734
297 EDIT-GET-LINE-NUMBER Action 735
Description 736
Parameters 736
298 EDIT-LABEL Action 737
Description 738

Parameters 738
299 EDIT-LINE-DELETE Action 739
Description 740
Parameters 740
300 EDIT-LINE-GET-SELECTION Action 741
Description 742
Parameters 742
301 EDIT-LINE-GET-TEXT Action 743
Description 744
Parameters 744
302 EDIT-LINE-INSERT Action 745
Description 746
Parameters 746
303 EDIT-LINE-SET-SELECTION Action 747
Description 748
Parameters 748
304 EDIT-LINE-SET-TEXT Action 749
Description 750
Parameters 750
305 ENUM-CHILDREN Action 753
Description 754
Parameters 754
306 ENUM-CLIENT-KEYS Action 755
Description 756
Parameters 756
307 ENSURE-VISIBLE Action 759
Description 760
Parameters 760
308 GET-CLIENT-VALUE Action 761
Description 762
Parameters 762
309 GET-CLIPBOARD-DATA Action 765
Description 766
Parameters 766
310 GET-FIRST-WINDOW Action 769
Description 770
Parameters 770
311 GET-FOCUS Action 771
Description 772
Parameters 772
312 GET-MESSAGE-TEXT Action 773
Description 774
Parameters 774
313 GET-NEXT-WINDOW Action 775
Description 776

Parameters 776
314 GET-TEXT Action 777
Description 778
Parameters 778
315 GET-SUBITEM-DATA Action 779
Description 780
Parameters 780
316 HELP Action 783
Description 784
Parameters 784
317 HOURGLASS-REMOVE Action 785
Description 786
Parameters 786
318 HOURGLASS-STACK Action 787
Description 788
Parameters 788
319 HOURGLASS-UNSTACK Action 789
Description 790
Parameters 790
320 INPUT-COPY-SELECTION Action 791
Description 792
Parameters 792
321 INPUT-CUT-SELECTION Action 793
Description 794
Parameters 794
322 INPUT-DELETE-SELECTION Action 795
Description 796
Parameters 796
323 INPUT-GET-LINE-LENGTH Action 797
Description 798
Parameters 798
324 INPUT-GET-SELECTION Action 799
Description 800
Parameters 800
325 INPUT-GET-TEXT Action 801
Description 802
Parameters 802
326 INPUT-PASTE Action 803
Description 804
Parameters 804
327 INPUT-SET-SELECTION Action 805
Description 806
Parameters 806
328 INPUT-SET-TEXT Action 807
Description 808

Parameters 808
329 INPUT-UNDO Action 809
Description 810
Parameters 810
330 INQ-CLICKPOSITION Action 811
Description 812
Parameters 812
331 INQ-DRAG-DROP Action 813
Description 814
Parameters 814
332 INQ-FORMAT-AVAILABLE Action 817
Description 818
Parameters 818
333 INQ-INNER-RECT Action 821
Description 822
Parameters 822
334 INQ-ITEM-BY-POSITION Action 825
Description 826
Parameters 826
335 INQ-NON-CLIENT-METRICS Action 827
Description 828
Parameters 828
336 LOAD-LAYOUT Action 831
Description 832
Parameters 832
337 MOVE-NAVIGATION-ITEMS Action 833
Description 834
Parameters 834
338 MESSAGE-BOX Action 835
Description 836
Parameters 836
339 OLE-ACTIVATE 839
Description 840
Parameters 840
340 OLE-DEACTIVATE 841
Description 842
Parameters 842
341 OLE-GET-DATA 843
Description 844
Parameters 844
342 OLE-INSERT-OBJECT 845
Description 846
Parameters 846
343 OLE-READ-FROM-FILE 847
Description 848

Parameters 848
344 OLE-SAVE-TO-FILE 849
Description 850
Parameters 850
345 OLE-SET-DATA 851
Description 852
Parameters 852
346 OPEN-CLIPBOARD Action 853
Description 854
Parameters 854
347 PERFORM-DRAG-DROP Action 857
Description 858
Parameters 858
348 PICK-FILENAME Action 861
Description 862
Parameters 862
349 PLAY-SOUND Action 865
Description 866
Parameters 866
350 PROCESS-EVENTS Action 867
Description 868
Parameters 868
351 RECALC-LAYOUT Action 869
Description 870
Parameters 870
352 REFRESH-LINKS Action 871
Description 872
Parameters 872
353 RESET-ATTRIBUTES Action 873
Description 874
Parameters 874
354 SAVE-LAYOUT Action 875
Description 876
Parameters 876
355 SET-ACCELERATION Action 877
Description 878
Parameters 878
356 SET-AUX-COLOR Action 879
Description 880
Parameters 880
357 SET-AUX-FONT Action 883
Description 884
Parameters 884
358 SET-CLIENT-VALUE Action 885
Description 886

Parameters 886
359 SET-CLIPBOARD-DATA Action 889
Description 890
Parameters 891
360 SET-FOCUS Action 893
Description 894
Parameters 894
361 SET-SUBITEM-DATA Action 895
Description 896
Parameters 896
362 SET-TABS Action 897
Description 898
Parameters 898
363 SET-TEXT Action 899
Description 900
Parameters 900
364 SET-TICKS Action 901
Description 902
Parameters 902
365 SET-TIME-RANGE Action 905
Description 906
Parameters 906
366 SHOW-CONTEXT-MENU Action 907
Description 908
Parameters 908
367 SORT-ITEMS Action 911
Description 912
Parameters 912
368 SYSTEM-GET-NATIVE-HANDLE Action 915
Description 916
Parameters 916
369 SYSTEM-PRINTER-SETUP Action 917
Description 918
Parameters 918
370 TABLE-DELETE-ROW Action 919
Description 920
Parameters 920
371 TABLE-FIND-FIELD Action 921
Description 922
Parameters 922
372 TABLE-GET-SELECTION Action 923
Description 924
Parameters 924
373 TABLE-INQUIRE-CELL Action 925
Description 926

Parameters ... 926
374 TABLE-INQUIRE-ROW Action ... 927
Description ... 928
Parameters ... 928
375 TABLE-INSERT-ROW Action ... 929
Description ... 930
Parameters ... 930
376 TABLE-REFRESH Action ... 931
Description ... 932
Parameters ... 932
377 TABLE-SET-SELECTION Action ... 933
Description ... 934
Parameters ... 934
378 TEXT-GET-EXTENT Action ... 935
Description ... 936
Parameters ... 936
379 UPDATE-COMMAND-STATUS Action ... 937
Description ... 938
Parameters ... 938
380 VALIDATE Action ... 939
Description ... 940
Parameters ... 940
V NGU Subprograms and Dialogs ... 941
381 General Information ... 943
382 NGU-CLIENT-ADVISE-HOT Subprogram ... 945
Natural Object Name ... 946
Parameters ... 946
383 NGU-CLIENT-ADVISE-TERM Subprogram ... 947
Natural Object Name ... 948
Description ... 948
Parameters ... 948
384 NGU-CLIENT-ADVISE-WARM Subprogram ... 949
Natural Object Name ... 950
Description ... 950
Parameters ... 950
385 NGU-CLIENT-CONNECT Subprogram ... 951
Natural Object Name ... 952
Description ... 952
Parameters ... 952
386 NGU-CLIENT-DISCONNECT Subprogram ... 953
Natural Object Name ... 954
Description ... 954
Parameters ... 954
387 NGU-CLIENT-EXECUTE Subprogram ... 955
Natural Object Name ... 956

Description 956
Parameters 956
388 NGU-CLIENT-GET-DATA Subprogram 957
Natural Object Name 958
Description 958
Parameters 958
389 NGU-CLIENT-POKE Subprogram 961
Natural Object Name 962
Description 962
Parameters 962
390 NGU-CLIENT-REQUEST Subprogram 963
Natural Object Name 964
Description 964
Parameters 964
391 NGU-CLIENT-STOP Subprogram 965
Natural Object Name 966
Description 966
Parameters 966
392 NGU-COLOUR-SELECT Dialog 967
Natural Object Name 968
Description 968
OPEN DIALOG Parameters 968
393 NGU-DIALOG-CLOSE-ALL Subprogram and Subroutine 969
Natural Object Names 970
Description 970
Parameters 970
394 NGU-FONT-SELECT Dialog 971
Natural Object Name 972
Description 972
OPEN DIALOG Parameters 972
395 NGU-MESSAGEBOX Dialog 973
Natural Object Name 974
Description 974
OPEN DIALOG Parameters 974
Separator Keyword 975
396 NGU-SERVER-DATA Subprogram 977
Natural Object Name 978
Description 978
397 NGU-SERVER-GET-DATA Subprogram 979
Natural Object Name 980
Parameters 981
398 NGU-SERVER-REGISTER Subprogram 983
Natural Object Name 984
Parameters 984
399 NGU-SERVER-STOP Subprogram 985

Natural Object Name 986
Description 986
Parameters 986
400 NGU-SERVER-UNREGISTER Subprogram 987
Natural Object Name 988
Description 988
Parameters 988
401 NGU-SERVER-WAIT Subprogram 989
Natural Object Name 990
Description 990
Parameters 991
402 NGULKEY1 Reserved Symbols 993
Color Symbols 994
Dialog Element Types 994
Event-Suppressing Symbols 994
Menu Item Style Symbols 994
Menu Item Symbols 994
Separator Symbols 995
Tool Bar Symbols 995

Dialog Component Reference

This documentation provides a reference of the components that can be used when developing an event-driven application with the dialog editor. These components include:

Dialogs and Dialog Elements	Dialogs and dialog elements represent the GUI object types available to Natural programs.
Attributes	List of attributes that be can be set in dialogs and controls.
Events	List of events that can be created by dialog elements.
PROCESS GUI Statement Actions	The PROCESS GUI statement actions execute procedures from within the PROCESS GUI statement.
NGU Subprograms and Dialogs	The NGU-prefixed subprograms and dialogs in library SYSTEM provide you with frequently needed functionality.

See also:

- *Dialog Editor* in the *Editors* documentation.
- *Introduction to Event-Driven Programming* and *Event-Driven Programming Techniques* in the *Programming Guide*.

When working with the dialog editor, you should have a working knowledge of Microsoft Windows and its terminology. If not, consult the Windows documentation for a description of basic Windows elements, usage and terminology.

I Dialogs and Dialog Elements

This part covers the following topics:

General Information
ActiveX Controls
Bitmap Control
Canvas Control
Column Specification Control
Control Box Control
Context Menu
Date and Time Picker (DTP) Control
Dialog Bar Control
Edit Area Control
Font Control
Graphic Text Control
Group Frame Control
GUI Control
Image Control
Image List Control
Input Field Control
List Box Control
List Box Item
List View Column
List View Control
List View Item
Line Control
MDI Child Window
MDI Frame Window

MDI Plug-in Window
Menu Bar
Menu Item
OLE Container Control
Progress Bar Control
Push Button Control
Radio Button Control
Rectangle Control
Scrollbar Control
Selection Box Control
Selection Box Item
Signal
Slider Control
Spin Control
Status Bar Control
Status Bar Pane
Standard Window
Submenu Control
Tab Control
Tab Control Tab
Table Control
Text Constant Control
Timer Control
Toggle Button Control
Tool Bar
Tool Bar Control
Tool Bar Item
Tree View Control
Tree View Item
Wallpaper Control

1 General Information

Dialogs and dialog elements represent the GUI object types available to Natural programs, whereby it is important to distinguish between the two uses of the word "dialog" to represent the different concepts of dialog object and dialog window (as implied here). The dialog object is a Natural object like a map or program, created by the dialog editor. When this object is invoked via the OPEN DIALOG statement, the dialog window is created, which is the physical representation of the dialog on the screen. A dialog may be created as one of the following window types: **Standard Window**, **MDI Frame Window**, **MDI Child Window** and **MDI Plug-in Window**.

Dialog Elements represent the controls used within the dialogs. These can include standard controls that have a user interface, such as push buttons and radio buttons, as well as controls such as timers and signals that do not.

Dialogs and dialog elements are created by PROCESS GUI Action **ADD** statements, either generated by the dialog editor, or explicitly coded by the Natural programmer. The value of the **TYPE** attribute determines the type of dialog or dialog element created. Each dialog or dialog element type listed above is also represented by a handle variable, which is defined as HANDLE OF GUI, where there can be any of the concrete types available for the TYPE attribute. Alternatively, the generic handle declaration HANDLE OF GUI can be used in situations where the type of object whose handle the variable should hold is not known in advance.

The handle variable allows access to the attributes of the dialog or dialog element after it has been created. Since the type of attribute access depends on the type of the dialog or dialog element, each topic listed above includes an attribute access table, where columns have the following meanings:

Column	Comment
Attribute Name	If an attribute is not listed in this column, no access to the attribute is allowed for this handle type.
Query	An "X" in this column means that the attribute may be queried for this handle type.
Set/Modify	An "X/-" in this column means that the attribute may be specified for this handle type when the dialog or dialog element is created (i.e., in the ADD action). A "-/X" in this column means that the attribute may be modified for this handle type after the dialog or dialog element has been created. An "X/X" in this column means that the attribute is both specifiable on creation and modifiable thereafter for dialogs or dialog elements of this handle type. A "-/-" in this column means that the attribute is read-only for this handle type.
In Attr. Window	An "X" in this column means that the attribute may be specified in the Attributes dialog for dialogs or dialog elements of this handle type in the Dialog Editor.

2 ActiveX Controls

- Description 6
- Natural Attributes for ActiveX Controls 6
- Events 7

Description

ActiveX controls are third-party custom controls that you can integrate into a Natural dialog. In addition to the Natural attributes listed below, the properties, methods and events of the respective ActiveX control are available. The handling of these properties, methods and events is described in Event driven programming techniques - Working with ActiveX controls.

Natural Attributes for ActiveX Controls

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-DATA	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
DRAG-MODE	X	X/X	X
DROP-MODE	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
OWNER	X	X/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STYLE	X	X/X	X
SUCCESSOR	X	-/-	
SUPPRESS-BEGIN-DRAG-EVENT	X	X/X	
SUPPRESS-CLIPBOARD-STATUS-EVENT	X	X/X	
SUPPRESS-COPY-EVENT	X	X/X	
SUPPRESS-CUT-EVENT	X	X/X	

Attribute Name	Query	Set/Modify	In Attr. Window
SUPPRESS-DELETE-EVENT	X	X/X	
SUPPRESS-DRAG-DROP-EVENT	X	X/X	
SUPPRESS-DRAG-ENTER-EVENT	X	X/X	
SUPPRESS-DRAG-LEAVE-EVENT	X	X/X	
SUPPRESS-DRAG-OVER-EVENT	X	X/X	
SUPPRESS-END-DRAG-EVENT	X	X/X	
SUPPRESS-ENTER-EVENT	X	X/X	
SUPPRESS-LEAVE-EVENT	X	X/X	
SUPPRESS-PASTE-EVENT	X	X/X	
SUPPRESS-UNDO-EVENT	X	X/X	
TOOLTIP	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

Begin-Drag Event, **Clipboard-Status Event**, **Copy Event**, **Cut Event**, **Delete Event**, **Drag-Drop Event**, **Drag-Enter Event**, **Drag-Leave Event**, **Drag-Over Event**, **End-Drag Event**, **Paste Event**, **Undo Event** (all events may be suppressed).

In addition to these Natural-specific events, this dialog element also generates Natural events from the events available with the ActiveX control.

3 Bitmap Control

- Description 10
- Attributes for Bitmap Control 10
- Events 11

Description

A bitmap control is a picture to be displayed anywhere within a dialog window. It helps explain the purpose of dialog elements in a graphical way. The end user may choose upon a bitmap control and drag it onto another bitmap control in the same dialog. You can allow this by using the attributes **ENABLED** and **DRAGGABLE** and the PROCESS GUI statement actions **INQ-CLICKPOSITION** and **INQ-DRAG-DROP**.

Attributes for Bitmap Control

Attribute Name	Query	Set/Modify	In Attr. Window
ACCELERATOR	X	X/X	X
BITMAP-FILE-NAME	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
DRAG-MODE	X	X/X	X
DROP-MODE	X	X/X	X
DRAGGABLE	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
OWNER	X	X/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STYLE	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
SUCCESSOR	X	-/-	
SUPPRESS-BEGIN-DRAG-EVENT	X	X/X	
SUPPRESS-CLICK-EVENT	X	X/X	
SUPPRESS-DBL-CLICK-EVENT	X	X/X	
SUPPRESS-DRAG-DROP-EVENT	X	X/X	
SUPPRESS-DRAG-ENTER-EVENT	X	X/X	
SUPPRESS-DRAG-LEAVE-EVENT	X	X/X	
SUPPRESS-DRAG-OVER-EVENT	X	X/X	
SUPPRESS-END-DRAG-EVENT	X	X/X	
TOOLTIP	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

Begin-Drag Event, **Click Event**, **Double-Click Event**, **Drag-Drop Event**, **Drag-Enter Event**, **Drag-Leave Event**, **Drag-Over Event**, **End-Drag Event** (all events may be suppressed).

4 Canvas Control

- Description 14
- Attributes for Canvas Control 14
- Events 15

Description

A canvas control provides a necessary background for the rectangle, line and graphic text controls. Once you have created a canvas control in the dialog, you can go on to create the rectangle, line and graphic text controls in it. The rectangle, line and graphic text controls are then displayed inside the borders of the canvas control; if they go beyond the canvas borders, they are clipped.

Attributes for Canvas Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
OFFSET-X	X	X/X	
OFFSET-Y	X	X/X	
OWNER	X	X/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STYLE	X	X/-	X

Attribute Name	Query	Set/Modify	In Attr. Window
SUCCESSOR	X	-/-	
SUPPRESS-CHANGE-EVENT	X	-/-	
SUPPRESS-CLICK-EVENT	X	X/X	
SUPPRESS-DBL-CLICK-EVENT	X	X/X	
TOOLTIP	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

Click Event, **Double-Click Event** (all events may be suppressed).

5 Column Specification Control

- Description 18
- Attributes for Column Specification Control 18
- COLUMN-TYPEs and their Attributes 19
- Events 19

Description

A column specification control is a dialog element that defines the columns in a table control. (A table control is a dialog element that represents a spreadsheet.) Once the table control defines the spreadsheet as such, its columns are defined by adding column-specification controls.

Attributes for Column Specification Control

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
COLUMN-TYPE	X	X/-	X
DIL-TEXT	X	X/X	X
FIRST-CHILD	X	-/-	
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
LENGTH	X	X/-	X
MODIFIABLE (input-field)	X	X/X	X
MODIFIABLE (selection-box)	X	X/-	X
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-W	X	X/X	X
STRING	X	X/X	X
STYLE	X	X/-	
SUCCESSOR	X	X/-	
SUPPRESS-CHANGE-EVENT	X	-/-	
SUPPRESS-CLICK-EVENT	X	X/X	
SUPPRESS-DBL-CLICK-EVENT	X	X/X	
SUPPRESS-ENTER-EVENT	X	X/X	
SUPPRESS-ENTER-CELL-EVENT	X	X/X	
SUPPRESS-FILL-EVENT	X	X/X	
SUPPRESS-LEAVE-EVENT	X	X/X	
SUPPRESS-LEAVE-CELL-EVENT	X	X/X	

Attribute Name	Query	Set/Modify	In Attr. Window
TYPE	X	X/-	
VISIBLE	X	-/-	

COLUMN-TYPEs and their Attributes

The attributes that are valid for column specification controls in general are not always available depending on the value of the attribute COLUMN-TYPE. Such a value might be, for example, “toggle button control”. The column then consists of cells that can be used like toggle button controls. For these toggle-button cells, only a subset of the attributes is available.

The following table specifies which attributes are *not* applicable to a COLUMN-TYPE.

COLUMN-TYPE	Attributes NOT Available
Input field control	(All available.)
Selection box control	(All available.)
Selection box item	DIL-TEXT, HELP-ID, LENGTH, MODIFIABLE, STYLE, SUPPRESS-CHANGE-EVENT, SUPPRESS-CLICK-EVENT, SUPPRESS-DBL-CLICK-EVENT, SUPPRESS-ENTER-EVENT, SUPPRESS-ENTER-CELL-EVENT, SUPPRESS-LEAVE-EVENT, SUPPRESS-LEAVE-CELL-EVENT, VISIBLE.
Toggle button control	LENGTH, STYLE, MODIFIABLE.

Events

This dialog element does not create events.

6 Control Box Control

- Description 22
- Attributes for Control Box Control 22
- Events 23

Description

The control box control is a general-purpose container control. Any dialog elements placed within the control box become child controls of the control box, thus allowing related dialog elements to be grouped together for programming convenience or ease of user manipulation in the dialog editor. For example, when a control box is made invisible, all the dialog elements it contains become invisible, and when it is moved, these dialog elements are moved with it.

Special attention has been paid to making it possible to create multiple “pages” of dialog elements at the same position in a dialog using control boxes marked with the 'Exclusive' style. Only one such page can then be visible at any one time, both at edit-time and at run-time, and the Dialog Editor automatically changes the active page according to the current selection. This feature can be used to support wizard dialogs and ActiveX tab controls. Please refer to the article Working with Control Boxes for more information.

Attributes for Control Box Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
DROP-MODE	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	

Attribute Name	Query	Set/Modify	In Attr. Window
SUPPRESS-DRAG-DROP-EVENT	X	X/X	
SUPPRESS-DRAG-ENTER-EVENT	X	X/X	
SUPPRESS-DRAG-LEAVE-EVENT	X	X/X	
SUPPRESS-DRAG-OVER-EVENT	X	X/X	
SUPPRESS-SIZE-EVENT	X	X/X	
TOOLTIP	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

Drag-Drop Event, **Drag-Enter Event**, **Drag-Leave Event**, **Drag-Over Event**, **Size Event** (all events may be suppressed).

7 Context Menu

- Description 26
- Attributes for Context Menu 26
- Events 27

Description

The context menu allows you to define menus which appear when the user clicks a dialog or dialog element with the right mouse button. You can retrieve the position of the click via the **INQ-CLICKPOSITION** action, should this be necessary.

Context menus are defined separately from the dialog elements and associated with the relevant dialog element(s) via the **CONTEXT-MENU** attribute. The context menu can be modified before it is displayed via the **Before-Open Event**.

The context menu is constructed and handled almost identically to the submenu control. Like the submenu control, a context menu can contain menu items which open up submenus when selected. Each submenu in such a multi-level context menu receives its own **Before-Open Event** each time the submenu is displayed.

Attributes for Context Menu

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTROL	X	-/-	
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
HELP-ID	X	X/X	
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
SUPPRESS-BEFORE-OPEN-EVENT	X	X/X	
TYPE	X	X/-	

Events

Before-Open Event (may be suppressed).

8 Date and Time Picker (DTP) Control

- Description 30
- Attributes for Date and Time Picker Control 30
- Events 31

Description

A date and time picker (DTP) control is a control that simplifies the user's task of entering date and time information.

For more information on DTP controls, please refer to the article Working with Date and Time Picker (DTP) Controls.

Attributes for Date and Time Picker Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CHECKED	X	X/X	
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
EDIT-MASK	X	X/X	X
DRAG-MODE	X	X/X	X
DROP-MODE	X	X/X	X
DRAGGABLE	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
MODIFIED	X	X/X	
MODIFIED-SUCCESSOR	X	-/-	
OWNER	X	X/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STYLE	X	X/X	X
SUCCESSOR	X	X/-	
SUPPRESS-CHANGE-EVENT	X	X/X	
SUPPRESS-CONTEXT-MENU-EVENT	X	X/X	
SUPPRESS-ENTER-EVENT	X	X/X	
SUPPRESS-LEAVE-EVENT	X	X/X	
TIME	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

Change Event, **Context-Menu Event**, **Enter Event**, **Leave Event** (all events may be suppressed).

9 Dialog Bar Control

- Description 34
- Attributes for Dialog Bar Control 34
- Events 35

Description

A dialog bar control is a general-purpose container control like the one used for the Library Workspace in Natural Studio. It can contain most other types of dialog elements as child controls, such as push buttons, list boxes, table controls and so on. A dialog bar is always "docked" to one of the sides of the frame window which contains it.

A dialog bar can optionally be (re-)dockable. A dockable dialog bar may be dragged to a new position at the one of the edges of the frame window, or floated in a separate window. Dockable dialog bars can optionally be sizeable, meaning that they can be stretched (via a splitter bar) parallel to the side of the window on which they are docked (if non-floating), or arbitrarily re-sized in the case of floating dialog bars. The application becomes notified of a change in the size of a dialog bar via a **SIZE** event, upon receipt of which it can (if desired) adapt the sizes of the child controls accordingly.

Dialog bars can optionally contain a gripper bar, by means of which the control may be dragged. Unless the control is marked with the "UI transparent" style, dragging is also possible by clicking anywhere on the control's background. The dialog bar may optionally also possess a close button and/or zoom button. The close button hides the control. The zoom button is only enabled and applicable if two or more dialog bars are placed alongside each other, and toggles the dialog bar between its maximized and "restored" state. If a dialog bar is maximized, the other dialog bars on the same row are minimized, such that only their gripper bar and close/hide buttons are visible. Clicking on the zoom button again restores the original sizes of all dialog bars on the row.

Attributes for Dialog Bar Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
BAR-ID	X	X/-	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	X
CONTEXT-MENU	X	X/X	X
DOCKING	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	

Attribute Name	Query	Set/Modify	In Attr. Window
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
LOCATION	X	X/X	X
MARGIN-X	X	X/X	X
MARGIN-Y	X	X/X	X
MAXIMIZABLE	X	-/-	
MAXIMIZED	X	X/X	X
MINIMIZED	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STRING	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	X/-	
SUPPRESS-CLOSE-EVENT	X	X/-	
SUPPRESS-SIZE-EVENT	X	X/X	
TOOLTIP	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X
WALLPAPER	X	X/X	X

Events

Close Event (may be suppressed), **Size Event**, (may be suppressed).

10 Edit Area Control

- Description 38
- Attributes for Edit Area Control 38
- Events 39

Description

In an edit area control, the end user may type in free-form text. It may contain any number of lines, and scroll bars may be set: if there are more lines in the edit area control than can be displayed, the end user may scroll to the desired line.

To use the free form text elsewhere in your application, you will have to set the text of the edit area control into alphanumeric Natural fields line by line. You can then query these lines of text one after the other.

You can manipulate the text in the edit area control by using PROCESS GUI statement actions. This helps you insert a new line, query a certain line, set a selection, query the selected text, and so on. Note that when transferring to and from an edit area control, the **STRING** attribute will only be able to hold 253 characters. If you want to move text longer than 253 characters, you can use the PROCESS GUI statement actions named EDIT-*. Alternatively, to get and set large quantities of text in a single operation, the **GET-TEXT** and **SET-TEXT** actions can be used, respectively.

Attributes for Edit Area Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
DROP-MODE	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
HELP-ID	X	X/X	X
HORIZ-SCROLLABLE	X	X/-	X

Attribute Name	Query	Set/Modify	In Attr. Window
LAST-CHILD	X	-/-	
LENGTH	X	X/X	
MODIFIABLE	X	X/X	X
MODIFIED	X	-/X	
MODIFIED-SUCCESSOR	X	-/-	
OWNER	X	X/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
RTL	X	-/-	X
STRING	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
SUPPRESS-CHANGE-EVENT	X	-/-	
SUPPRESS-DRAG-DROP-EVENT	X	X/X	
SUPPRESS-DRAG-ENTER-EVENT	X	X/X	
SUPPRESS-DRAG-LEAVE-EVENT	X	X/X	
SUPPRESS-DRAG-OVER-EVENT	X	X/X	
SUPPRESS-ENTER-EVENT	X	X/X	
SUPPRESS-LEAVE-EVENT	X	X/X	
TOOLTIP	X	X/X	
TYPE	X	X/-	X
VERT-SCROLLABLE	X	X/-	X
VISIBLE	X	X/X	X

Events

Change Event, **Drag-Drop Event**, **Drag-Enter Event**, **Drag-Leave Event**, **Drag-Over Event**, **Enter Event**, **Leave Event** (all events may be suppressed).

11 Font Control

- Description 42
- Attributes for Font Control 42
- Events 43

Description

A font control is used to display the **STRING** attribute value of a dialog element in a certain font face, size and style. It is generated automatically if you select a font in the attributes window of a dialog element. You should, however, not rely on the automatic generation of specific font control names. You can also create a font control dynamically by using the **NGU-FONT-SELECT** dialog from library SYSTEM. When you assign the handle value of this font control to the **FONT-HANDLE** attribute of another dialog element, the STRING of the other dialog element is displayed accordingly.

Example:

```
#TC-1.FONT-HANDLE := #FNT-1
/* The STRING of the text constant control /* #TC-1 will be displayed in the face,
/* style and size of #FNT-1.
```

Attributes for Font Control

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
FIRST-CHILD	X	-/-	
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
STRING	X	X/-	X
SUCCESSOR	X	-/-	
TYPE	X	X/-	

Events

This dialog element does not create events.

12 Graphic Text Control

- Description 46
- Attributes for Graphic Text Control 46
- Events 47

Description

A graphic text control represents a one-line piece of text to be created on top of a canvas control. If it goes beyond the area of the canvas control, it is clipped.

Attributes for Graphic Text Control

Attribute Name	Query	Set/Modify
CLIENT-DATA	X	X/X
CLIENT-HANDLE	X	X/X
CLIENT-KEY	X	X/X
CLIENT-VALUE	X	X/X
FIRST-CHILD	X	-/-
FONT-HANDLE	X	X/X
FOREGROUND-COLOUR-NAME	X	X/X
FOREGROUND-COLOUR-VALUE	X	X/X
LAST-CHILD	X	-/-
PARENT	X	X/-
PREDECESSOR	X	-/-
RECTANGLE-H	X	X/X
RECTANGLE-W	X	X/X
RECTANGLE-X	X	X/X
RECTANGLE-Y	X	X/X
STRING	X	X/X
SUCCESSOR	X	X/-
STYLE	X	-/-
TYPE	X	X/-
VISIBLE	X	X/X

Events

This dialog element does not create events.

13 Group Frame Control

- Description 50
- Attributes for Group Frame Control 50
- Events 51

Description

A group frame control is used to optically group related dialog elements within a dialog. It can, for example, be used to frame a group of radio button controls that have a common **GROUP-ID**, but where it is not optically evident to the end user that these radio button controls are related.

You may use the group frame control without the text in the upper left corner. Then the group frame control acts as a simple frame. If you use it with text, this text may contain a mnemonic key (&). If the mnemonic key is pressed, the following dialog element in the navigation sequence gets the focus.

Attributes for Group Frame Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
LAST-CHILD	X	-/-	
OWNER	X	X/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
RTL	X	-/-	X
STRING	X	X/X	X
SUCCESSOR	X	-/-	

Attribute Name	Query	Set/Modify	In Attr. Window
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

This dialog element does not create events.

14 GUI Control

- Description 54
- Attributes for GUI Control 54

Description

In event handler code, you can use the HANDLE OF GUI variable to refer to the handle of any type of dialog element. This can be useful, for example, if you are querying an attribute value in all dialog elements on one level: you go through the dialog elements one after the other; in the course of this query, it is not clear which type of dialog element is going to be queried next. Then a GUI handle allows you to query the next dialog element regardless of its type. This saves a lot of coding, because otherwise, you would have to query each dialog element's attribute value separately.

The syntax checker will accept all existing attributes for a HANDLE OF GUI variable because all existing dialog elements must be covered. Nevertheless, if you query or modify an attribute which is not valid for the dialog element to which the HANDLE OF GUI variable refers, a runtime error will occur. It is therefore only advisable to use a HANDLE OF GUI variable if the attribute you are querying or modifying applies to most, if not all of the dialog elements.

Example:

```
... 1 #CONTROL
HANDLE OF GUI ... #CONTROL := #DLG$WINDOW.FIRST-CHILD REPEAT UNTIL #CONTROL =
NULL-HANDLE ... #CONTROL := #CONTROL.SUCCESSOR END-REPEAT
```

Attributes for GUI Control

Attribute Name	Query	Set/Modify
ACCELERATOR	X	X/X
ACTIVE-CHILD	X	-/X
AUTOSELECT	X	X/X
BACKGROUND-COLOUR-NAME	X	X/X
BACKGROUND-COLOUR-VALUE	X	X/X
BAR-ID	X	X/-
BITMAP-FILE-NAME	X	X/X
BLEND	X	X/-
BUDDY	X	X/X
CELL-ATTRIBUTES	X	X/X
CHECKED	X	X/X
CHECKED-SUCCESSOR	X	X/X
CLIENT-DATA	X	X/X

Attribute Name	Query	Set/Modify
CLIENT-HANDLE	X	X/X
CLIENT-KEY	X	X/X
CLIENT-VALUE	X	X/X
COLUMN	X	X/X
COLUMN-COUNT	X	-/-
COLUMN-TYPE	X	X/-
COMPATIBILITY	X	X/-
CONTEXT-MENU	X	X/X
CONTROL	X	-/-
DEFAULT-BUTTON	X	X/X
DESCENDING	X	X/X
DIL-TEXT	X	X/X
DOCKING	X	X/X
DPI	X	X/-
DRAGGABLE	X	X/X
DRAG-MODE	X	X/X
DROP-MODE	X	X/X
EDIT-MASK	X	X/X
EMBEDDED-OBJECT	X	X/X
ENABLED	X	X/X
EVENT-QUEUEING	X	X/X
EXPANDED	X	X/X
FIRST-CHILD	X	-/-
FIRST-COLUMN-WIDTH	X	X/X
FIRST-VISIBLE-COLUMN	X	X/X
FIRST-VISIBLE-ITEM	X	X/X
FIRST-VISIBLE-ROW	X	X/X
FOLLOWS	X	X/X
FONT-HANDLE	X	X/X
FONT-STRING	X	X/-
FOREGROUND-COLOUR-NAME	X	X/X
FOREGROUND-COLOUR-VALUE	X	X/X
FORMAT	X	X/-
FROZEN-COLUMNS	X	X/X
GROUP-ID	X	X/X
HAS-DIL	X	X/X

Attribute Name	Query	Set/Modify
HAS-FIRST-COLUMN	X	X/X
HAS-HELP-BUTTON	X	X/-
HAS-MENU-BAR	X	X/-
HAS-STATUS-BAR	X	X/X
HAS-SYSTEM-BUTTON	X	X/-
HAS-TOOLBAR	X	X/X
HAS-TOOLTIP	X	X/X
HEADER-FONT-HANDLE	X	X/X
HEADER-HEIGHT	X	X/X
HELP-FILENAME	X	X/X
HELP-ID	X	X/X
HORIZ-SCROLLABLE	X	X/-
ICONIZED	X	X/X
IMAGE	X	X/X
IMAGE-INDEX	X	X/X
IMAGE-LIST	X	X/X
INPLACE-ACTIVE	X	-/-
ITEM	X	-/-
ITEM-H	X	X/X
ITEM-W	X	X/X
LAST-CHILD	X	-/-
LENGTH	X	X/X
LINE	X	X/X
LINKED	X	X/-
LOCATION	X	X/X
MARGIN-X	X	X/X
MARGIN-Y	X	X/X
MAX	X	X/X
MAXIMIZABLE	X	X/-
MAXIMIZED	X	X/X
MENU-HANDLE	X	X/X
MENU-ITEM-OLE	X	X/X
MENU-ITEM-TYPE	X	X/-
MIN	X	X/X
MINIMIZABLE	X	X/-
MINIMIZED	X	X/X

Attribute Name	Query	Set/Modify
MODIFIABLE	X	X/-
MODIFIED	X	-/X
MODIFIED-SUCCESSOR	X	-/-
MULTI-SELECTION	X	X/-
NAME	X	-/-
OBJECT-SIZE	X	X/X
OFFSET-X	X	X/X
OFFSET-Y	X	X/X
OVERLAY	X	X/X
OVERLAY-INDEX	X	X/X
OWNER	X	X/-
P1-X	X	X/X
P1-Y	X	X/X
P2-X	X	X/X
P2-Y	X	X/X
PAGE	X	X/X
PARENT	X	X/-
POPUP-HELP	X	X/X
POSITION	X	X/X
PREDECESSOR	X	-/-
RECTANGLE-H	X	X/X
RECTANGLE-W	X	X/X
RECTANGLE-X	X	X/X
RECTANGLE-Y	X	X/X
ROW	X	X/X
ROW-COUNT	X	X/-
ROW-HEIGHT	X	X/X
SAME-AS	X	X/-
SCROLLRANGE-X	X	X/X
SCROLLRANGE-Y	X	X/X
SELECTED	X	X/X
SELECTED-SUCCESSOR	X	-/-
SERVER-OBJECT	X	X/X
SERVER-PROGID	X	X/X
SHARED	X	X/X
SIZE-MODIFIABLE	X	X/-

Attribute Name	Query	Set/Modify
SLIDER	X	X/X
SORTED	X	X/-
SPACING	X	X/X
SPACING-X	X	X/X
SPACING-Y	X	X/X
STATUS-HANDLE	X	X/X
STATUS-TEXT	-	-/X
STRING	X	X/X
STYLE	X	X/-
SUCCESSOR	X	X/-
SUPPRESS-ACTIVATE-EVENT	X	X/X
SUPPRESS-AFTER-EDIT-EVENT	X	X/X
SUPPRESS-BEFORE-EDIT-EVENT	X	X/X
SUPPRESS-BEGIN-DRAG-EVENT	X	X/X
SUPPRESS-BEFORE-OPEN-EVENT	X	X/X
SUPPRESS-CHANGE-EVENT	X	X/X
SUPPRESS-CHECK-EVENT	X	X/X
SUPPRESS-CLICK-EVENT	X	X/X
SUPPRESS-CLIENT-SIZE-EVENT	X	X/X
SUPPRESS-CLIPBOARD-STATUS-EVENT	X	X/X
SUPPRESS-CLOSE-EVENT	X	X/X
SUPPRESS-COLLAPSE-EVENT	X	X/X
SUPPRESS-COMMAND-STATUS-EVENT	X	X/X
SUPPRESS-CONTEXT-MENU-EVENT	X	X/X
SUPPRESS-COPY-EVENT	X	X/X
SUPPRESS-CUT-EVENT	X	X/X
SUPPRESS-DBL-CLICK-EVENT	X	X/X
SUPPRESS-DELETE-EVENT	X	X/X
SUPPRESS-DELETE-ROW-EVENT	X	X/X
SUPPRESS-DRAG-DROP-EVENT	X	X/X
SUPPRESS-DRAG-ENTER-EVENT	X	X/X
SUPPRESS-DRAG-LEAVE-EVENT	X	X/X
SUPPRESS-DRAG-OVER-EVENT	X	X/X
SUPPRESS-END-DRAG-EVENT	X	X/X
SUPPRESS-ENTER-EVENT	X	X/X
SUPPRESS-ENTER-CELL-EVENT	X	X/X

Attribute Name	Query	Set/Modify
SUPPRESS-EXPAND-EVENT	X	X/X
SUPPRESS-FILL-EVENT	X	X/X
SUPPRESS-IDLE-EVENT	X	X/X
SUPPRESS-INSERT-ROW-EVENT	X	X/X
SUPPRESS-LEAVE-EVENT	X	X/X
SUPPRESS-LEAVE-CELL-EVENT	X	X/X
SUPPRESS-PASTE-EVENT	X	X/X
SUPPRESS-SIZE-EVENT	X	X/X
SUPPRESS-TOP-EVENT	X	X/X
SUPPRESS-UNDO-EVENT	X	X/X
TIME	X	X/X
TIMER-INTERVAL	X	X/X
TOOLBAR-HANDLE	X	X/X
TOOLBAR-POS	X	X/-
TOOLTIP	X	X/X
TYPE	X	X/-
VARIABLE	-	X/X
VERSION	X	-/-
VERT-SCROLLABLE	X	X/-
VIEW-MODE	X	X/X
VISIBLE	X	X/X
WALLPAPER	X	X/X
ZOOM-FACTOR	X	X/X

15 Image Control

- Description 62
- Attributes for Image Control 62
- Events 62

Description

An image control represents an item in an **image list control**.

Image controls may contain more than one image, which can be either base images or overlay images, as determined by the image control's **STYLE** attribute value. Overlay images (if specified) are superimposed on the base image displayed for an item.

For more information on image controls, please refer to the article Working with Image List Controls.

Attributes for Image Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
BITMAP-FILE-NAME	X	X/X	X
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
FIRST-CHILD	X	-/-	
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
STYLE	X	X/X	X
SUCCESSOR	X	X/-	
TYPE	X	X/-	

Events

This dialog element does not create events.

16 Image List Control

- Description 64
- Attributes for Image List Control 64
- Events 64

Description

An image list control is used for storing multiple images of the same size and color representation, allowing the images to be efficiently shared by multiple dialog elements (e.g., by multiple items in a list view or tree view control).

For more information on image list controls, please refer to the article Working with Image List Controls.

Attributes for Image List Control

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
FIRST-CHILD	X	-/-	
ITEM-H	X	X/-	X
ITEM-W	X	X/-	X
FOLLOWS	X	X/X	
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
STYLE	X	X/X	X
SUCCESSOR	X	X/-	
TYPE	X	X/-	

Events

This dialog element does not create events.

17 Input Field Control

- Description 66
- Attributes for Input Field Control 66
- Events 68

Description

An input field control is used to enter data in a single line. It corresponds to a field in a Natural program's INPUT statement. You can map database fields or other program variables to an input field control by means of a linked variable. The input is automatically copied to a linked variable if you select the “Linked Variable” option in the attribute window's “Source” dialog box and you enter the name of the linked variable, for example, the database field. When the end user has finished entering data and the input field control loses the focus, the data entered are validated.

You can also validate the input data by assigning an **EDIT-MASK** attribute to a (linked variable) input field control: the check is performed when the input field control loses the focus.

When the linked variables have been modified by code and you want to display the new values, you use the PROCESS GUI statement action **REFRESH-LINKS**.

Whenever input is rejected, for example, because the linked variable has another Natural data type, or because the **EDIT-MASK** was not matched, a message box is displayed that prompts the end user to “Retry” or “Cancel”. The end user must press “Retry” to keep the current content of the input field control and to modify it. The end user must press “Cancel” to reset the input field control to the last valid content.

Note: Notes: When you create an input field control, and you assign it a STYLE value of “Center” ('c') or “Right” ('r'), the input field control's height must be greater than the height of the system font. Otherwise, the STRING will not be displayed. Input in an input field control is limited to 253 characters.

Attributes for Input Field Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
EDIT-MASK	X	X/X	X
ENABLED	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
LENGTH	X	X/X	X
LINKED	X	X/-	X
MODIFIABLE	X	X/X	X
MODIFIED	X	-/X	
MODIFIED-SUCCESSOR	X	-/-	
OWNER	X	X/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
RTL	X	-/-	X
STRING	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
SUPPRESS-CHANGE-EVENT	X	-/-	
SUPPRESS-ENTER-EVENT	X	X/X	
SUPPRESS-LEAVE-EVENT	X	X/X	
TOOLTIP	X	X/X	
TYPE	X	X/-	
VARIABLE	-	X/X	X
VISIBLE	X	X/X	X

Events

Change Event, **Enter Event**, **Leave Event** (all events may be suppressed).

Note: The change event occurs for an input field control if either the end user or the code changes the content. It does not occur on the initial setting of the input field control. It is not recommended to manipulate the system focus from within the change event handler. Furthermore, the content of a linked variable is not updated in a change event.

18 List Box Control

- Description 70
- Attributes for List Box Control 71
- Events 72

Description

A list box control contains a number of list box items from which the end user can select one or more. You can insert items into a list box control by:

- defining them in the dialog editor at design time, and by
- creating them dynamically at runtime using the PROCESS GUI statement actions **ADD**, **ADD-ITEMS** and **ADD-ITEMS-EX**.

The **Fill Event** enables you to implement dynamically growing list box controls.

The **MULTI-SELECTION** attribute specifies for a list box control whether the end user may select one or several items at a time.

Note: A list box control must have MULTI-SELECTION = TRUE to be able to use **SELECTED-SUCCESSOR** with the children list box items.

To sort list box item STRINGs alphabetically, you use the **SORTED** attribute for the list box control. When the STRING of a list box item is changed, the items are automatically sorted again. You can then go through the list of items with the **SUCCESSOR** attribute and the new sort sequence is reflected.

When a **Click Event** or a **Double-Click Event** occurs for a list box item, the list box control receives the event and the list- box control's event handler can determine the clicked item by querying the attribute **SELECTED-SUCCESSOR** for the list box control. When several list box items are selected at the same time, the second selected item can be determined by querying the attribute **SELECTED-SUCCESSOR** of the first item, and so on.

If you specify event handlers for the click and the double-click events, the click-event handler will be executed prior to the double-click-event handler.

This may imply that the click-event handler suppresses the execution of the double-click-event handler (**SUPPRESS-DBL-CLICK-EVENT** attribute).

Attributes for List Box Control

Attribute Name	Query	Set/Modify	In Attr. Window
AUTOSELECT	X	X/X	
ACCELERATOR	X	X/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
DRAG-MODE	X	X/X	X
DROP-MODE	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FIRST-VISIBLE-ITEM	X	X/X	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
MODIFIED	X	-/X	
MODIFIED-SUCCESSOR	X	-/-	
MULTI-SELECTION	X	X/-	X
OWNER	X	X/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
ROW-COUNT	X	-/-	
RTL	X	-/-	X

Attribute Name	Query	Set/Modify	In Attr. Window
SELECTED-SUCCESSOR	X	-/-	
SORTED	X	X/-	X
STYLE	X	-/-	X
SUCCESSOR	X	-/-	
SUPPRESS-BEGIN-DRAG-EVENT	X	X/X	
SUPPRESS-CLICK-EVENT	X	X/X	
SUPPRESS-CLIPBOARD-STATUS-EVENT	X	X/X	
SUPPRESS-COPY-EVENT	X	X/X	
SUPPRESS-CUT-EVENT	X	X/X	
SUPPRESS-DBL-CLICK-EVENT	X	X/X	
SUPPRESS-DELETE-EVENT	X	X/X	
SUPPRESS-DRAG-DROP-EVENT	X	X/X	
SUPPRESS-DRAG-ENTER-EVENT	X	X/X	
SUPPRESS-DRAG-LEAVE-EVENT	X	X/X	
SUPPRESS-DRAG-OVER-EVENT	X	X/X	
SUPPRESS-END-DRAG-EVENT	X	X/X	
SUPPRESS-FILL-EVENT	X	X/X	
SUPPRESS-PASTE-EVENT	X	X/X	
SUPPRESS-UNDO-EVENT	X	X/X	
TOOLTIP	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

Begin-Drag Event, **Click Event**, **Clipboard-Status Event**, **Copy Event**, **Cut Event**, **Delete Event**, **Double-Click Event**, **Drag-Drop Event**, **Drag-Enter Event**, **Drag-Leave Event**, **Drag-Over Event**, **End-Drag Event**, **Fill Event**, **Paste Event**, **Undo Event** (all events may be suppressed).

19 List Box Item

- Description 74
- Attributes for List Box Item 74
- Events 75

Description

A list box item is an item inside a list box control.

The list box items are added to the list box control either by specifying them in the dialog editor's List Box Attributes window, or by using the PROCESS GUI statement actions **ADD-ITEMS**, **ADD-ITEMS-EX** or **ADD**.

Items created using the dialog editor are represented in a Natural variable. The variable name is generated as: *list-box-handle-name* -ITEMS (1: *number-of-list-box-items)*. Example: #LB-1-ITEMS (1:5). The fifth item would then be called: #LB-1-ITEMS (5).

Items created with the PROCESS GUI statement can be given any variable name in the HANDLE definitions. Example: #MYITEM-1 HANDLE OF LISTBOXITEM.

To sort list box item STRINGs alphabetically, you use the **SORTED** attribute for the list box control. When the STRING of a list box item is changed, the items are automatically sorted again. You can then go through the list of items with the **SUCCESSOR** attribute and the new sort sequence is reflected. For more information on how to go through a list of items, see *Working with List Box Controls and Selection Box Controls* in the section *Event-driven Programming Techniques*.

A list box item does not receive any events. Instead, the list box control receives them, triggering event handlers there.

Note: To select or deselect individual list box items at runtime, hold down CTRL while clicking with the left mouse button.

Attributes for List Box Item

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
FIRST-CHILD	X	-/-	
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
SELECTED	X	X/X	
SELECTED-SUCCESSOR	X	-/-	

Attribute Name	Query	Set/Modify	In Attr. Window
STRING	X	X/X	X
SUCCESSOR	X	X/-	
TYPE	X	X/-	

Note: To be able to use **SELECTED-SUCCESSOR**, the parent list box control must have **MULTI-SELECTION** = TRUE.

Events

This dialog element does not create events.

20 List View Column

- Description 78
- Attributes for List View Column 78
- Events 79

Description

A list view column represents a column in a list view control, as displayed when the list view control is in report view mode.

For more information on list view columns, please refer to the article Working with List View Controls.

Attributes for List View Column

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DESCENDING	X	-/-	
EDIT-MASK	X	X/-	X
FIRST-CHILD	X	-/-	
FORMAT	X	X/-	X
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
SORTED	X	-/-	
STRING	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	X/-	
SUPPRESS-CLICK-EVENT	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

Click Event (may be suppressed).

21 List View Control

- Description 82
- 82
- Events 84

Description

A list view control optionally provides an icon and/or column-based item list, as used for displaying library contents within MDI document windows in Natural Studio, for example.

For more information on list view controls, please refer to the article Working with List View Controls.

Attribute Name	Query	Set/Modify	In Attr. Window
ACCELERATOR	X	X/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CHECKED-SUCCESSOR	X	-/-	
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
COLUMN-COUNT	X	-/-	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
DRAG-MODE	X	X/X	X
DROP-MODE	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FIRST-VISIBLE-ITEM	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
HAS-TOOLTIP	X	X/X	X
HELP-ID	X	X/X	X
IMAGE-LIST	X	X/X	X
ITEM	X	-/-	
LAST-CHILD	X	-/-	
MODIFIABLE	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
MULTI-SELECTION	X	X/-	X
OFFSET-X	X	-/-	
OFFSET-Y	X	-/-	
OWNER	X	X/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
ROW-COUNT	X	-/-	
SELECTED-SUCCESSOR	X	-/-	
SORTED	X	X/-	X
SPACING	X	X/X	X
SPACING-X	X	X/X	X
SPACING-Y	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	X/-	
SUPPRESS-ACTIVATE-EVENT	X	X/X	
SUPPRESS-AFTER-EDIT-EVENT	X	X/X	
SUPPRESS-BEFORE-EDIT-EVENT	X	X/X	
SUPPRESS-BEGIN-DRAG-EVENT	X	X/X	
SUPPRESS-CHECK-EVENT	X	X/X	
SUPPRESS-CLICK-EVENT	X	X/X	
SUPPRESS-CLIPBOARD-STATUS-EVENT	X	X/X	
SUPPRESS-CONTEXT-MENU-EVENT	X	X/X	
SUPPRESS-COPY-EVENT	X	X/X	
SUPPRESS-CUT-EVENT	X	X/X	
SUPPRESS-DELETE-EVENT	X	X/X	
SUPPRESS-DRAG-DROP-EVENT	X	X/X	
SUPPRESS-DRAG-ENTER-EVENT	X	X/X	
SUPPRESS-DRAG-LEAVE-EVENT	X	X/X	
SUPPRESS-DRAG-OVER-EVENT	X	X/X	
SUPPRESS-END-DRAG-EVENT	X	X/X	
SUPPRESS-PASTE-EVENT	X	X/X	
SUPPRESS-UNDO-EVENT	X	X/X	

Attribute Name	Query	Set/Modify	In Attr. Window
TYPE	X	X/-	
VIEW-MODE	X	X/X	X
VISIBLE	X	X/X	X
WALLPAPER	X	X/X	X

Events

Activate Event, After-Edit Event, Before-Edit Event, Begin-Drag Event, Check Event, Click Event, Clipboard-Status Event, Context-Menu Event, Copy Event, Cut Event, Delete Event, Drag-Drop Event, Drag-Enter Event, Drag-Leave Event, Drag-Over Event, End-Drag Event, Paste Event, Undo Event (all events may be suppressed).

22 List View Item

- Description 86
- Attributes for List View Item 86
- Events 87

Description

A list view item represents a data item in a list view control, and is displayed by a caption and (optionally) an image from the parent control's image list (if any).

For more information on list view items, please refer to the article Working with List View Controls.

Attributes for List View Item

Attribute Name	Query	Set/Modify	In Attr. Window
CHECKED	X	X/X	X
CHECKED-SUCCESSOR	X	-/-	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
FIRST-CHILD	X	-/-	
IMAGE	X	X/X	X
IMAGE-INDEX	X	X/X	X
LAST-CHILD	X	-/-	
LENGTH	X	X/X	X
MODIFIABLE	X	X/X	X
OVERLAY	X	X/X	X
OVERLAY-INDEX	X	X/X	X
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
SELECTED	X	X/X	
SELECTED-SUCCESSOR	X	-/-	
STRING	X	X/X	X
STYLE	X	X/X	X
SUCCESSOR	X	X/-	
TOOLTIP	X	X/X	X
TYPE	X	X/-	

Events

This dialog element does not create events.

23 Line Control

- Description 90
- Attributes for Line Control 90
- Events 90

Description

A line control represents a line to be created in a canvas control. If it goes beyond the area of the canvas control, it is clipped.

Attributes for Line Control

Attribute Name	Query	Set/Modify
CLIENT-DATA	X	X/X
CLIENT-HANDLE	X	X/X
CLIENT-KEY	X	X/X
CLIENT-VALUE	X	X/X
FIRST-CHILD	X	-/-
FOREGROUND-COLOUR-NAME	X	X/X
FOREGROUND-COLOUR-VALUE	X	X/X
LAST-CHILD	X	-/-
PARENT	X	X/-
PREDECESSOR	X	-/-
P1-X	X	X/X
P1-Y	X	X/X
P2-X	X	X/X
P2-Y	X	X/X
STYLE	X	-/-
SUCCESSOR	X	X/-
TYPE	X	X/-
VISIBLE	X	X/X

Events

This dialog element does not create events.

24 MDI Child Window

- Description 92
- Attributes for MDI Child Window 92
- Events 95

Description

An MDI (Multiple Document Interface) child window is a child dialog of an **MDI Frame Window**. When an MDI child window is maximized, it expands to fill the interior (MDI client window) of the MDI frame. When minimized, it appears as an icon within the MDI client window, and not as an icon in the Windows Task Bar.

At any one time, only one MDI child window can be active. This window appears in front of the other child windows (if any) and is displayed with an emphasized title bar color. Note that any frame components (e.g., **Menu Bar**, **Tool Bar** or **Status Bar**) defined for an active MDI child window are not displayed within the child window itself, but instead replace the corresponding components (if any) of the **MDI Frame Window**. The frame components (if any) of inactive MDI child windows are not displayed at all. The active MDI child window may be queried or set via the **ACTIVE-CHILD** attribute of the **MDI Frame Window**. An MDI child window also receives an **ENTER** event when it is activated, and a **LEAVE** event when it is deactivated.

Note that (in general) multiple instances of an MDI child dialog may be open at any time, each existing as a separate MDI child window. For example, several instances of an "Employee Details" MDI child dialog may be simultaneously open in a personnel-tracking application, displaying information for different employees in a company or department. For this reason, docked windows (such as **Tool Bar Controls**, **Status Bar Controls** and **Dialog Bar Controls**) "belonging" to an MDI child dialog must instead be defined for the MDI frame dialog and explicitly switched by the Natural application when a window belonging to a different MDI child dialog is activated, or when the last MDI child window is deactivated. This also has the advantage of minimizing the performance and resource overheads when opening new instances of an MDI child dialog, since these controls are created just once, regardless of the number of instances opened, and are shared by each instance. Only the state of the window's contents (e.g., the **ENABLED** or **CHECKED** states of the Tool Bar Items) is updated in this case, in order to reflect the current selection (if any) in the currently active instance. This is typically done in the **COMMAND-STATUS** event.

Attributes for MDI Child Window

Attribute Name	Query	Set/Modify	In Attr. Window
AUTO-ADJUST	X	-/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
BITMAP-FILE-NAME	X	X/X	X
CLIENT-DATA	X	X/-	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-VALUE	X	X/X	
COMPATIBILITY	X	X/-	X
CONTEXT-MENU	X	X/X	X
DEFAULT-BUTTON	X	X/X	X
DOCKING	X	X/X	X
DPI	X	X/-	
DROP-MODE	X	X/X	X
ENABLED	X	X/X	X
EVENT-QUEUEING	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	-/-	
FONT-HANDLE	X	-/-	
FONT-STRING	X	X/-	X
HAS-DIL	X	X/X	X
HAS-HELP-BUTTON	X	X/-	X
HAS-MENU-BAR	X	X/-	
HAS-STATUS-BAR	X	X/X	X
HAS-SYSTEM-BUTTON	X	-/-	
HAS-TOOLBAR	X	X/X	X
HAS-TOOLTIP	X	X/X	
HELP-FILENAME	X	X/X	X
HELP-ID	X	X/X	X
HORIZ-SCROLLABLE	X	X/X	X
LAST-CHILD	X	-/-	
MAXIMIZABLE	X	-/-	
MAXIMIZED	X	X/X	X
MENU-HANDLE	X	X/X	
MINIMIZABLE	X	-/-	
MINIMIZED	X	X/X	X
MODIFIED	X	X/X	
MODIFIED-SUCCESSOR	X	-/-	
NAME	X	-/-	
PARENT	X	X/-	
POPUP-HELP	X	X/X	X
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
RTL	X	-/-	X
SCROLLRANGE-X	X	X/X	
SCROLLRANGE-Y	X	X/X	
SIZE-MODIFIABLE	X	-/-	
STATUS-HANDLE	X	X/X	
STATUS-TEXT		-/X	
STRING	X	X/X	X
STYLE	X	-/X	X
SUCCESSOR	X	-/-	
SUPPRESS-CLIENT-SIZE-EVENT	X	X/X	
SUPPRESS-COMMAND-STATUS-EVENT	X	X/X	
SUPPRESS-DRAG-DROP-EVENT	X	X/X	
SUPPRESS-DRAG-ENTER-EVENT	X	X/X	
SUPPRESS-DRAG-LEAVE-EVENT	X	X/X	
SUPPRESS-DRAG-OVER-EVENT	X	X/X	
SUPPRESS-ENTER-EVENT	X	X/X	
SUPPRESS-IDLE-EVENT	X	X/X	
SUPPRESS-LEAVE-EVENT	X	X/X	
SUPPRESS-SIZE-EVENT	X	X/X	
TOOLBAR-HANDLE	X	X/-	
TOOLTIP	X	X/X	
TYPE	X	X/-	X
VERSION	X	-/-	
VERT-SCROLLABLE	X	X/X	X
VISIBLE	X	X/X	X
WALLPAPER	X	X/X	X

Events

After-Any Event, **After-Open Event**, **Before-Any Event**, **Before Open Event**, **Before-Open Event**, **Client-Size Event** (may be suppressed), **Close Event**, **Command-Status Event** (may be suppressed), **Drag-Drop Event** (may be suppressed), **Drag-Enter Event** (may be suppressed), **Drag-Leave Event** (may be suppressed), **Drag-Over Event** (may be suppressed), **Enter Event** (may be suppressed), **Error Event**, **Idle Event** (may be suppressed), **Leave Event** (may be suppressed), **Size Event** (may be suppressed).

25 MDI Frame Window

- Description 98
- Attributes for MDI Frame Window 98
- Events 100

Description

An MDI (Multiple Document Interface) frame window is a type of dialog that hosts a workspace window known as the "MDI client window", which can contain multiple **MDI Child Windows**. These child windows are also sometimes referred to as "document" windows, since they are often used in Windows applications to display the contents of different documents - hence the term "Multiple Document Interface").

An application that makes use of the MDI concept is known as an MDI application. A good example of such an application is Natural Studio itself, the main window being the MDI frame window and the editor windows and list views being the MDI child windows.

Note that the MDI client window is implicitly created by Natural when the MDI frame widow is created, and is not a separate GUI object. Thus, attributes relating to the MDI client window (such as **BACKGROUND-COLOUR-NAME**) are applied to the MDI frame window instead. The MDI client window automatically extends to fill the complete interior area of the MDI frame window not occupied by any status bars, tool bars or dialog bars. Its current size can be retrieved by applying the **INQ-INNER-RECT** action to the MDI frame window. Note that the MDI client window may not contain any dialog elements.

Attributes for MDI Frame Window

Attribute Name	Query	Set/Modify	In Attr. Window
ACTIVE-CHILD	X	-/X	
AUTO-ADJUST	X	-/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
BITMAP-FILE-NAME	X	X/X	X
CLIENT-DATA	X	X/-	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
COMPATIBILITY	X	X/-	X
CONTEXT-MENU	X	X/X	X
DEFAULT-BUTTON	X	X/X	X
DOCKING	X	X/X	X
DPI	X	X/-	
DROP-MODE	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
ENABLED	X	X/X	X
EVENT-QUEUEING	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	-/-	
FONT-HANDLE	X	-/-	
FONT-STRING	X	X/-	X
HAS-DIL	X	X/X	X
HAS-HELP-BUTTON	X	X/X	X
HAS-MENU-BAR	X	X/-	
HAS-STATUS-BAR	X	X/X	X
HAS-SYSTEM-BUTTON	X	X/-	X
HAS-TOOLBAR	X	X/X	X
HAS-TOOLTIP	X	X/X	
HELP-FILENAME	X	X/X	X
HELP-ID	X	X/X	X
HORIZ-SCROLLABLE	X	X/X	X
LAST-CHILD	X	-/-	
MAXIMIZABLE	X	-/-	X
MAXIMIZED	X	X/X	X
MENU-HANDLE	X	X/X	
MINIMIZABLE	X	X/-	X
MINIMIZED	X	X/X	X
MODIFIED	X	X/X	
MODIFIED-SUCCESSOR	X	-/-	
NAME	X	-/-	
PARENT	X	X/-	
POPUP-HELP	X	X/X	X
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
RTL	X	-/-	X
SCROLLRANGE-X	X	X/X	
SCROLLRANGE-Y	X	X/X	
SIZE-MODIFIABLE	X	-/-	X

Attribute Name	Query	Set/Modify	In Attr. Window
STATUS-HANDLE	X	X/X	
STATUS-TEXT		-/X	
STRING	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
SUPPRESS-CLIENT-SIZE-EVENT	X	X/X	
SUPPRESS-COMMAND-STATUS-EVENT	X	X/X	
SUPPRESS-DRAG-DROP-EVENT	X	X/X	
SUPPRESS-DRAG-ENTER-EVENT	X	X/X	
SUPPRESS-DRAG-LEAVE-EVENT	X	X/X	
SUPPRESS-DRAG-OVER-EVENT	X	X/X	
SUPPRESS-ENTER-EVENT	X	X/X	
SUPPRESS-IDLE-EVENT	X	X/X	
SUPPRESS-LEAVE-EVENT	X	X/X	
SUPPRESS-SIZE-EVENT	X	X/X	
TOOLBAR-HANDLE	X	X/-	
TOOLBAR-POS	X	X/-	X
TOOLTIP	X	X/X	
TYPE	X	X/-	X
VERSION	X	-/-	
VERT-SCROLLABLE	X	X/X	X
VISIBLE	X	X/X	X
WALLPAPER	X	X/X	X

Events

After-Any Event, **After-Open Event**, **Before-Any Event**, **Before Open Event**, **Before-Open Event**, **Client-Size Event** (may be suppressed), **Close Event**, **Command-Status Event** (may be suppressed), **Drag-Drop Event** (may be suppressed), **Drag-Enter Event** (may be suppressed), **Drag-Leave Event** (may be suppressed), **Drag-Over Event** (may be suppressed), **Enter Event** (may be suppressed), **Error Event**, **Idle Event** (may be suppressed), **Leave Event** (may be suppressed), **Size Event** (may be suppressed).

26 MDI Plug-in Window

- Description 102
- Events 104

Description

An MDI (Multiple Document Interface) plug-in window is very similar to an **MDI Child Window**. However, instead of being opened as a child of an MDI frame window created in Natural, it is used to create a dialog to appear as an MDI child window within Natural Studio itself. Note, however, that the creation of such a plug-in window is a two-step process. Firstly, the dialog is created as usual, with the **PARENT** attribute set to NULL-HANDLE. At this point in time, the dialog is always invisible. Secondly, the plug-in code creates an instance of the generic document wrapper class, specifying the dialog's ID as parameter, causing a new MDI child window to be opened containing the specified dialog.

Note: Because plug-ins should harmonize with the existing Natural Studio environment, accelerator key combinations used by Natural Studio (e.g., CTRL+N to open a new program editor window) are processed by Natural Studio and are not forwarded to the plug-in. Therefore, you cannot use the **ACCELERATOR** attribute of any dialog elements created in the plug-in to redefine the behavior of these reserved key combinations.

Attribute Name	Query	Set/Modify	In Attr. Window
AUTO-ADJUST	X	-/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
BITMAP-FILE-NAME	X	X/-	X
CLIENT-DATA	X	X/-	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
COMPATIBILITY	X	X/-	X
CONTEXT-MENU	X	X/X	X
DEFAULT-BUTTON	X	X/X	X
DPI	X	X/-	
DROP-MODE	X	X/X	X
ENABLED	X	X/X	X
EVENT-QUEUEING	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	-/-	
FONT-HANDLE	X	-/-	
FONT-STRING	X	X/-	X
HAS-TOOLTIP	X	X/X	

Attribute Name	Query	Set/Modify	In Attr. Window
HELP-FILENAME	X	X/X	X
HELP-ID	X	X/X	X
HORIZ-SCROLLABLE	X	X/X	X
LAST-CHILD	X	-/-	
MAXIMIZED	X	-/-	
MINIMIZED	X	-/-	
MODIFIED	X	X/X	
MODIFIED-SUCCESSOR	X	-/-	
NAME	X	-/-	
PARENT	X	X/-	
POPUP-HELP	X	X/X	X
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/-	X
RECTANGLE-W	X	X/-	X
RECTANGLE-X	X	X/-	X
RECTANGLE-Y	X	X/-	X
SCROLLRANGE-X	X	X/X	
SCROLLRANGE-Y	X	X/X	
STRING	X	X/-	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
SUPPRESS-CLIENT-SIZE-EVENT	X	X/X	
SUPPRESS-DRAG-DROP-EVENT	X	X/X	
SUPPRESS-DRAG-ENTER-EVENT	X	X/X	
SUPPRESS-DRAG-LEAVE-EVENT	X	X/X	
SUPPRESS-DRAG-OVER-EVENT	X	X/X	
SUPPRESS-ENTER-EVENT	X	X/X	
SUPPRESS-LEAVE-EVENT	X	X/X	
TYPE	X	X/-	X
VERSION	X	-/-	
VERT-SCROLLABLE	X	X/X	X
WALLPAPER	X	X/X	X

Events

After-Any Event, **After Open Event**, **Before-Any Event**, **Before Open Event**, **Client-Size Event** (may be suppressed), **Close Event**, **Drag-Drop Event** (may be suppressed), **Drag-Enter Event** (may be suppressed), **Drag-Leave Event** (may be suppressed), **Drag-Over Event** (may be suppressed), **Enter Event** (may be suppressed), **Error Event**, **Leave Event** (may be suppressed).

27 Menu Bar

- Description 106
- Attributes for Menu Bar 106
- Events 107

Description

A menu bar is displayed at the top of the dialog window. It is the top level of a menu structure and contains menu items (second level). A menu item may be of type submenu. Then it is a submenu control which may be pulled down; it contains menu items (third level). The number of levels is unlimited.

A menu bar only becomes visible in a dialog when the attribute **MENU-HANDLE** is set to a value for the dialog. The PARENT of a menu bar may be a dialog or a NULL-HANDLE. If a dialog is the PARENT, the menu structure is specific to the dialog and is deleted when the dialog is deleted. If a NULL-HANDLE is the PARENT, the menu structure is free and will be closed when the application is closed. A free menu structure can be shared among several dialogs when the handles are defined in a global data area rather than in a local data area.

For an MDI child window, the menu bar is displayed only at the top of the MDI frame window, not in the MDI child window. The MDI children all share one menu bar, which is displayed in the MDI frame window. Every time another MDI child window is activated, the menu bar changes to reflect the menu bar defined for the particular MDI child window.

Note: You may create only one menu bar per dialog. By default, the dialog editor generates a menu bar named "#DLG-MENU-BAR", the parent being the dialog window.

Attributes for Menu Bar

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
FIRST-CHILD	X	-/-	
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
SUCCESSOR	X	-/-	
TYPE	X	X/-	
VISIBLE	X	X/X	

Events

This dialog element does not create events.

28 Menu Item

- Description 110
- Attributes for Menu Item 110
- Events 111

Description

A menu item is an item inside a menu bar or a submenu control. It is a child of a menu bar or a submenu control.

There are several types of menu items (values of the attribute **MENU-ITEM-TYPE**). Menu items of type MT-NORMAL trigger the click-event handler when an end user clicks on them. The MT-SUBMENU type of menu item is associated with a submenu control, which is pulled down when an end user clicks on the menu item.

It is not recommended to define menu items of type MT-NORMAL on the first level of a menu bar. Instead, you should define menu items of type MT-SUBMENU. When you give a name to a menu item by setting the **STRING** attribute to a name used by the windowing system, such as "Cascade", you must not modify this STRING value dynamically.

Attributes for Menu Item

Attribute Name	Query	Set/Modify	In Attr. Window
ACCELERATOR	X	X/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
BITMAP-FILE-NAME	X	X/X	X
CHECKED [1)]	X	X/X	X
CLIENT-DATA [2)]	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
DIL-TEXT [2)]	X	X/X	X
ENABLED [3)]	X	X/X	X
FIRST-CHILD	X	-/-	
LAST-CHILD	X	-/-	
MENU-HANDLE [4)]	X	X/-	
MENU-ITEM-OLE	X	X/X	
MENU-ITEM-TYPE [5)]	X	X/-	X
PARENT	X	X/-	
PREDECESSOR	X	-/-	
SAME-AS	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
SHARED	X	X/X	X
STRING [3)]	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
TOOLTIP	X	-/-	
TYPE	X	X/-	

1) For MENU-ITEM-TYPE = MT-NORMAL only. Not for menu items inside the menu bar.

2) For MENU-ITEM-TYPEs MT-NORMAL and MT-MDI only.

3) Not for MENU-ITEM-TYPE = MT-SEPARATOR.

4) For MENU-ITEM-TYPEs MT-SUBMENU and MT-WINDOWMENU only.

5) If MENU-ITEM-TYPE = MT-WINDOWMENU, the menu item must be inside the menu bar (at the top level).

Events

Click Event.

29 OLE Container Control

- Description 114
- Attributes for OLE Container Control 114
- Events 115

Description

An OLE container control enables you to integrate OLE objects in a Natural dialog.

Attributes for OLE Container Control

Attribute Name	Query	Set/Modify	In Attr. Window
ACCELERATOR	X	X/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
DIL-TEXT	X	X/X	X
EMBEDDED-OBJECT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
HELP-ID	X	X/X	X
ICONIZED	X	X/-	
INPLACE-ACTIVE	X	-/-	
LAST-CHILD	X	-/-	
MODIFIABLE	X	X/X	X
OBJECT-SIZE	X	-/-	
OWNER	X	X/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
SERVER-OBJECT	X	X/X	X
SERVER-PROGID	X	X/X	X
STYLE	X	X/-	X

Attribute Name	Query	Set/Modify	In Attr. Window
SUCCESSOR	X	-/-	
SUPPRESS-ACTIVATE-EVENT	X	X/X	
SUPPRESS-CHANGE-EVENT	X	-/-	
SUPPRESS-CLICK-EVENT	X	X/X	
SUPPRESS-CLOSE-EVENT	X	X/X	
SUPPRESS-DBL-CLICK-EVENT	X	X/X	
TOOLTIP	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X
ZOOM-FACTOR	X	X/X	X

Events

Activate Event, **Change Event**, **Click Event**, **Close Event**, **Double-Click Event** (all events may be suppressed).

30 Progress Bar Control

- Description 118
- Attributes for Progress Bar Control 118
- Events 119

Description

A progress bar control is used to indicate the progress made during a time-consuming task.

The progress bar's range is defined by the control's **MIN** (no bar visible) and **MAX** (bar covers full extent of control) attribute values. The current extent of the progress bar is determined via the **POSITION** attribute, which must be an integer within the defined range. The smaller the range, the higher the granularity and the less accurately the progress may be specified. In order to effectively use the control, the range must be non-zero.

Attributes for Progress Bar Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
LAST-CHILD	X	-/-	
MAX	X	X/X	X
MIN	X	X/X	X
OWNER	X	X/-	
PARENT	X	X/-	
POSITION	X	X/X	X
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
STYLE	X	X/-	X
SUCCESSOR	X	X/-	
SUPPRESS-CONTEXT-MENU-EVENT	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

Context-Menu Event (may be suppressed).

31 Push Button Control

- Description 122
- Attributes for Push Button Control 122
- Events 123

Description

A push button control is a representation of a button. The end user can click on it to trigger a certain action (specified in the click-event handler code). You can use a push button control's handle value in the **DEFAULT-BUTTON** attribute of a dialog; your push button control then serves as the dialog's default button.

Note: Under Windows, a push button control's background and foreground colors will be displayed as the system default at runtime, regardless of the specified value.

Attributes for Push Button Control

Attribute Name	Query	Set/Modify	In Attr. Window
ACCELERATOR	X	X/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
OWNER	X	X/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
RECTANGLE-Y	X	X/X	X
STRING	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
TOOLTIP	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

Click Event.

32 Radio Button Control

- Description 126
- Attributes for Radio Button Control 126
- Events 127

Description

A radio button control is a selection item. You can logically group several radio button controls by assigning them a common **GROUP-ID** attribute. Out of this group, none or one may be selected by default. To pre-select a radio button control, you assign it a **CHECKED** attribute with a value of checked. Please note that only the radio button control checked last will remain checked; the one checked before will be unchecked automatically.

You use it, for example, to let the end user switch an option on or off, affecting the status of other radio button controls of the same group. A change in the state of a toggle button control, by contrast, will not affect the state of other toggle button controls in the same dialog.

Attributes for Radio Button Control

Attribute Name	Query	Set/Modify	In Attr. Window
ACCELERATOR	X	X/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CHECKED	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
GROUP-ID	X	X/-	X
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
MODIFIED	X	-/X	
MODIFIED-SUCCESSOR	X	-/-	

Attribute Name	Query	Set/Modify	In Attr. Window
OWNER	X	X/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
RTL	X	-/-	X
STRING	X	X/X	X
SUCCESSOR	X	-/-	
SUPPRESS-CLICK-EVENT	X	X/X	
TOOLTIP	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

Click Event (may be suppressed).

33 Rectangle Control

- Description 130
- Attributes for Rectangle Control 130
- Events 131

Description

A rectangle control represents a filled rectangle to be created in of a **Canvas Control**. If it goes beyond the area of the canvas control, it is clipped.

Attributes for Rectangle Control

Attribute Name	Query	Set/Modify
BACKGROUND-COLOUR-NAME	X	X/X
BACKGROUND-COLOUR-VALUE	X	X/X
CLIENT-DATA	X	X/X
CLIENT-HANDLE	X	X/X
CLIENT-KEY	X	X/X
CLIENT-VALUE	X	X/X
FIRST-CHILD	X	-/-
FOREGROUND-COLOUR-NAME	X	X/X
FOREGROUND-COLOUR-VALUE	X	X/X
LAST-CHILD	X	-/-
PREDECESSOR	X	-/-
PARENT	X	X/-
RECTANGLE-H	X	X/X
RECTANGLE-W	X	X/X
RECTANGLE-X	X	X/X
RECTANGLE-Y	X	X/X
STYLE	X	X/-
SUCCESSOR	X	X/-
TYPE	X	X/-
VISIBLE	X	X/X

Events

This dialog element does not create events.

34 Scrollbar Control

- Description 134
- Attributes for Scrollbar Control 134
- Events 135

Description

A scroll bar control enables the end user to select a position on a scale. If, for example, you want the end user to select a percentage value on a scale from 1 to 100, you can use a scroll bar control and tell the end user in the **STRING** attribute of a text constant control which percentage value (integer) was selected. To limit the scale, you must always determine values for the **MIN** and **MAX** attributes of your scroll bar control. You can then use the **LINE** and **PAGE** attributes to determine step sizes for scrolling with the arrow buttons (LINE) or the scroll bar shaft (PAGE). You can also set and query the position of the slider with the **SLIDER** attribute.

Note: The range of the dialog scroll bar control attributes **MIN**, **MAX**, **PAGE**, **LINE** and **SLIDER** has been increased. Now for all of these attributes a range of -1073741823 to 1073741823 is permitted.

Attributes for Scrollbar Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
LINE	X	X/X	X
MAX	X	X/X	X
MIN	X	X/X	X
MODIFIED	X	-/X	
MODIFIED-SUCCESSOR	X	-/-	

Attribute Name	Query	Set/Modify	In Attr. Window
OWNER	X	X/-	
PAGE	X	X/X	X
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
RTL	X	-/-	X
SLIDER	X	X/X	X
STYLE	-	X/X	X
SUCCESSOR	X	-/-	
SUPPRESS-CHANGE-EVENT	X	-/-	
TOOLTIP	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

Change Event (may be suppressed).

35 Selection Box Control

- Description 138
- Attributes for Selection Box Control 139
- Events 140

Description

A selection box control is a combination of an input field, a list box, and a push button. When no selection has been made, it consists of an input-field with a push button next to it. With the push button, the end user may open a list box from the input field. This list box contains any number of items from which the end user can select one. The selected item is then copied into the input-field.

It is possible to get a selection box control without a push button where the list box is always dropped down: the **STYLE** attribute value must have the value "X".

You can map database fields or other program variables to the input section of a selection box control by means of a linked variable. The input is automatically copied to a linked variable if you select the “Linked Variable” option in the attribute window's “Source” dialog box and you enter the name of the linked variable, for example the database field. When the end user has finished entering data and the selection box control loses the focus, the entered data are validated.

You can also validate the input data by assigning an **EDIT-MASK** attribute to the selection box control: the check is performed when the input-field in the selection box control loses the focus.

When the linked variables have been modified by code and you want to display the new values, you use the PROCESS GUI statement action **REFRESH-LINKS**.

Whenever input is rejected, for example, because the linked variable has another Natural data type, or because the **EDIT-MASK** was not matched, a message box is displayed that prompts the end user to “Retry” or “Cancel”. The end user must press “Retry” to keep the current content of the input field control and to modify it. The end user must press “Cancel” to reset the input field control to the last valid content.

Note: Input in a selection box control's input-field is limited to 253 characters. If you set the **STYLE** attribute to the value "X" (dropped down), the **MODIFIABLE** attribute value is automatically set to TRUE. If you set the **MODIFIABLE** attribute to FALSE, the **STYLE** attribute automatically no longer has the value “X”.

Attributes for Selection Box Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
EDIT-MASK	X	X/-	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
LENGTH	X	X/X	X
LINKED	X	X/-	X
MODIFIABLE	X	X/-	X
MODIFIED	X	-/X	
MODIFIED-SUCCESSOR	X	-/-	
OWNER	X	X/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
ROW-COUNT	X	-/-	
RTL	X	-/-	X
SELECTED-SUCCESSOR	X	-/-	
SORTED	X	X/-	

Attribute Name	Query	Set/Modify	In Attr. Window
STRING	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
SUPPRESS-BEFORE-OPEN-EVENT	X	X/X	
SUPPRESS-CHANGE-EVENT	X	-/-	
SUPPRESS-ENTER-EVENT	X	X/X	
SUPPRESS-LEAVE-EVENT	X	X/X	
TOOLTIP	X	X/X	
TYPE	X	X/-	
VARIABLE		X/X	X
VISIBLE	X	X/X	X

Events

Before Open Event, **Change Event**, **Enter Event**, **Leave Event** (all events may be suppressed).

Note: The change event occurs for a selection box control if the end user changes the content, if the code changes the content or if an item is selected and transferred to the input-field area. It does not occur on the initial setting of the selection box control. It is not recommended to manipulate the system focus from within the change event handler.

36 Selection Box Item

- Description 142
- Attributes for Selection Box Item 142
- Events 143

Description

A selection box item is an item inside a selection box control.

The selection box items are added to the list box control either by specifying them in the dialog editor's selection box attributes window, or by using the PROCESS GUI statement actions **ADD-ITEMS**, **ADD-ITEMS-EX** or **ADD**. You can find out the selected item by querying the **STRING** attribute of the selection box control.

Items created using the dialog editor are represented in a Natural variable. The variable name is generated as: *selection-box-handle-name* -ITEMS (1: *number-of-selection-box-items*). Example: #SB-1-ITEMS (1:5). The fifth item would then be called: #SB-1-ITEMS (5).

Items created with the PROCESS GUI statement actions can be given any variable name in the HANDLE definitions.

Example: #MYITEM-1 HANDLE OF SELECTIONBOXITEM.

A selection box item does not receive any events. Instead, the selection box control receives them, triggering event handlers there.

Attributes for Selection Box Item

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
DIL-TEXT	X	X/X	X
FIRST-CHILD	X	-/-	
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
SELECTED	X	X/X	
STRING	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
SUCCESSOR	X	X/-	
TYPE	X	X/-	

Events

This dialog element does not create events.

37 Signal

- Description 146
- Attributes for Signal 146
- Events 147

Description

A signal is an abstract dialog element, which does not have a user interface of its own. It is used to represent an application action which can be triggered via any number of menu items and/or tool bar items, by linking these items to the signal via their **SAME-AS** attribute.

By representing each program action by a signal, and linking to the signal from each menu or tool bar item which triggers this action, the relevant attribute values only need to be specified once for the signal itself. The signal's attributes are automatically inherited by all items which are linked to it. This inheritence mechanism also applies to any modifications made to the signal after it has been created. For example, if the **ENABLED** attribute of the signal is set to FALSE, all menu items and tool bar items linked to it will be automatically disabled

If a menu or tool bar item that is linked to a signal is clicked, a **Click Event** is raised for the signal itself, rather than for the menu or tool bar item, ensuring that the same code is invoked for all items which are linked to the same signal.

Attributes for Signal

Attribute Name	Query	Set/Modify	In Attr. Window
ACCELERATOR	X	X/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
BITMAP-FILE-NAME	X	X/X	X
CHECKED	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	X
DIL-TEXT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
LAST-CHILD	X	-/-	
MENU-ITEM-TYPE	X	X/X	X
PARENT	X	X/-	
PREDECESSOR	X	-/-	
SHARED	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
SUCCESSOR	X	-/-	
SUPPRESS-CLICK-EVENT	X	X/X	
TOOLTIP	X	X/X	X
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

Click Event (may be suppressed).

38 Slider Control

- Description 150
- Attributes for Slider Control 150
- Events 151

Description

A slider control is akin to the sliders often found on sound mixers or graphic equalizers. It consists of a slider, often referred to as the thumb, that can be moved along a linear track.

The **SLIDER** attribute can be used to set or query the thumb position, which represents the control's current value, and can be any integer value within the slider's range, as defined by the control's **MIN** and **MAX** attributes. Hence the slider range also determines the number of possible slider positions. In order to be able to move the thumb, a non-zero slider range must be defined.

The thumb's track may be oriented vertically if the "Vertical (v)" **STYLE** is set, otherwise it is oriented horizontally. The thumb can be moved by either dragging the thumb, clicking on the slider track on either side of the thumb, or using the cursor or Page Up or Page Down keys. If the thumb is dragged, the "Position tip (p)" **STYLE** determines whether or not a tooltip window should appear in order to display the current position. Otherwise, the **LINE** and **PAGE** attributes can be used to specify the number of positions the thumb should be moved in the respective cases.

Depending on the setting of the control's "Side 1 ticks (1)" and "Side 2 ticks (2)" **STYLE** flags, tick marks are either not displayed at all, or are displayed on either or both sides of the skider track. Note that the tick marks at the each end of the track are always present. If the "Auto ticks (a)" **STYLE** is set, tick marks (if shown) are automatically displayed at regular intervals, as determined by the **SPACING** attribute (defaults to 1, implying a tick mark at every position). Without this style, the tick marks (apart from the implicit tick marks at either end of the range), if desired, must be set manually via the **SET-TICKS** action. These ticks marks can be removed again via the **CLEAR-TICKS** action.

marks can be removed again via the **CLEAR-TICKS** action. When the slider is moved, a **Change Event** is raised (if not suppressed) for the control, and the control's **MODIFIED** attribute is implicitly set.

Attributes for Slider Control

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
ENABLED	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
HAS-TOOLTIP	X	X/-	X
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
LINE	X	X/X	X
MAX	X	X/X	X
MIN	X	X/X	X
MODIFIED	X	X/X	
MODIFIED-SUCCESSOR	X	-/-	
OWNER	X	X/-	
PAGE	X	X/X	X
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
SLIDER	X	X/X	X
SPACING	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	X/-	
SUPPRESS-CHANGE-EVENT	X	X/X	
SUPPRESS-CONTEXT-MENU-EVENT	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

Change Event, **Context-Menu Event** (all events may be suppressed).

39 Spin Control

- Description 154
- 154
- Events 155

Description

A spin control allows the user to select a value from a range of possible values by “spinning” (i.e., repeatedly incrementing or decrementing) through them, similar to the process of setting the time on a digital clock.

For more information on spin controls, please refer to the article Working with Spin Controls.

Attribute Name	Query	Set/Modify	In Attr. Window
BUDDY	X	X/X	
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
MAX	X	X/X	X
MIN	X	X/X	X
MODIFIED	X	X/X	
MODIFIED-SUCCESSOR	X	-/-	
OWNER	X	X/-	
PARENT	X	X/-	
POSITION	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	X/-	
SUPPRESS-CHANGE-EVENT	X	X/X	

Attribute Name	Query	Set/Modify	In Attr. Window
SUPPRESS-CONTEXT-MENU-EVENT	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

Change Event, Context-Menu Event (all events may be suppressed).

40 Status Bar Control

- Description 158
- Attributes for Status Bar Control 158
- Events 159

Description

A status bar control is an alternative to the traditional status bar that is created by setting the dialog's **HAS-STATUS-BAR** attribute. However, the status bar control offers a range of advanced features which the tradional Natural status bars do not support. These new features include a Windows-like appearance with (optionally) recessed or raised sections - referred to as “panes” - and (optionally) a sizing grip (see illustration above). The status bar panes are themselves dialog elements, and are therefore documented under their own section below. Note that, although it is possible to define a status bar control with no panes, such a status bar control can only display a single text string. In order to use any of the advanced features, it is necessary to define one or more panes.

A status bar control can be displayed either at the top or bottom of a dialog. Furthermore, a dialog may have more than one status bar control. Status information set via the dialog's **STATUS-TEXT** attribute is automatically redirected to the status bar control (if any) specified by the dialog's **STATUS-HANDLE** attribute.

You can set a minimum pane height for the status bar control using the **ITEM-H** attribute. The distance between the items and the status bar control's border is determined with the **MARGIN-X** and **MARGIN-Y** attributes.

Attributes for Status Bar Control

Attribute Name	Query	Set/Modify	In Attr. Window
BAR-ID	X	X/-	
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	X
CONTEXT-MENU	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
HAS-TOOLTIP	X	X/X	X
ITEM-H	X	X/X	X
LAST-CHILD	X	-/-	
LOCATION	X	X/X	X
MARGIN-X	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
MARGIN-Y	X	X/X	X
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	-/-	
RECTANGLE-W	X	-/-	
RECTANGLE-X	X	-/-	
RECTANGLE-Y	X	-/-	
STRING	X	-/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
TOOLTIP	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

This dialog element does not create events.

41 Status Bar Pane

- Description 162
- Attributes for Status Bar Pane 162
- Events 163

Description

A status bar pane is a logical section of a status bar control. You can define more panes than are physically visible in the control by selectively setting the **VISIBLE** attribute of those logical panes which should not appear, according to the application context (for example, depending on which MDI child dialog is active).

A pane can be stretchy or non-stretchy. A stretchy pane is indicated by setting the pane's **RECTANGLE-W** attribute to zero. The width of a stretchy pane changes when the width of the parent dialog is changed, and is determined by the following formula:

width = (available width - total width of all non-stretchy panes) / no. of stretchy panes

The first (visible) stretchy pane is referred to as the message pane, and is used as the default pane in cases where no pane is explicitly specified. This is the case, for example, when text is output by setting the status bar control's **STRING** attribute or the dialog's **STATUS-TEXT** attribute. The message pane is also the one that is used for displaying a dialog element's dynamic information line (DIL) text, defined via its **DIL-TEXT** attribute.

Both an icon and a string can be specified for a pane by setting the pane's **BITMAP-FILE-NAME** and **STRING** attributes, respectively. If the **ENABLED** attribute is set to FALSE, both the icon and string are either grayed out or completely suppressed (depending on a pane-style flag setting). Furthermore, the status bar pane supports the **TOOLTIP** attributes, allowing pane-specific tool tips to be specified when the mouse pointer hovers over the pane.

Attributes for Status Bar Pane

Attribute Name	Query	Set/Modify	In Attr. Window
BITMAP-FILE-NAME	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-W	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
SHARED	X	X/X	X
STRING	X	-/X	X
STYLE	X	X/-	X
SUCCESSOR	X	X/-	
SUPPRESS-CLICK-EVENT	X	X/X	
SUPPRESS-DBL-CLICK-EVENT	X	X/X	
TOOLTIP	X	X/X	X
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

Click Event, **Double-Click Event** (all events may be suppressed).

42 Standard Window

- Description 166
- Attributes for Standard Window 166
- Events 169

Description

A standard window is the most common dialog type. It is a top-level dialog that is not capable of containing any child dialogs. For this reason, it is sometimes known as an SDI (Single Document Interface) window, to distinguish it from MDI (Multiple Document Interface) frame windows, where multiple MDI child windows can be simultaneously open, each capable of displaying a separate "document".

Note: The term "document" is used to refer to any set of related data, and does not necessarily correspond to a separate file in the file system.

Standard windows can exist in one of three forms: Modeless, Modal or Dialog Box. These are not separate types, but are instead implemented as values of the **STYLE** attribute. A Dialog Box differs from the other two forms in that control only returns from the OPEN DIALOG statement after the dialog box has been processed and closed. Thus, the execution of a dialog box is said to be "synchronous". In contrast, Modeless and Modal dialogs return control to the caller immediately after the OPEN and **AFTER-OPEN** events have been processed, although the dialog remains open and capable of receiving events until it is closed. The execution in this case is said to be "asynchronous", because the code following the OPEN DIALOG statement is executed before the dialog is closed. The difference between the Modeless and Modal dialogs is that, when a Modal dialog is activated, all other dialogs are implicitly disabled and are thus temporarily inaccessible to the user, whereas Modeless dialogs allow the user to switch to, and work with, another dialog. Note that Dialog Box dialogs are also implicitly modal. Of the three forms of dialog, Modal dialogs are the most infrequently encountered. The vast majority of dialogs fall into the "modeless and asynchronous" (i.e., Modeless) or "modal and synchronous" (i.e., Dialog Box) categories.

Note that the **PARENT** (if any) of a Standard Window refers to the "owner" window, and does not imply any special containment. This informs Windows that the two dialogs "belong" together. For example, Windows ensures that owned windows are kept together with (and in front of) their owner in the window depth ("Z-order") sequence, and that owned windows are automatically hidden when their owner window is minimized.

Attributes for Standard Window

Attribute Name	Query	Set/Modify	In Attr. Window
AUTO-ADJUST	X	-/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
BITMAP-FILE-NAME	X	X/X	X
CLIENT-DATA	X	X/-	

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
COMPATIBILITY	X	X/-	X
CONTEXT-MENU	X	X/X	X
CLIENT-VALUE	X	X/X	
DEFAULT-BUTTON	X	X/X	X
DOCKING	X	X/X	X
DPI	X	X/-	
DROP-MODE	X	X/X	X
ENABLED	X	X/X	X
EVENT-QUEUEING	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	-/-	
FONT-HANDLE	X	-/-	
FONT-STRING	X	X/-	X
HAS-DIL	X	X/X	X
HAS-HELP-BUTTON	X	X/-	X
HAS-MENU-BAR	X	X/-	
HAS-STATUS-BAR	X	X/X	X
HAS-SYSTEM-BUTTON	X	X/-	X
HAS-TOOLBAR	X	X/X	X
HAS-TOOLTIP	X	X/X	
HELP-FILENAME	X	X/X	X
HELP-ID	X	X/X	X
HORIZ-SCROLLABLE	X	X/X	X
LAST-CHILD	X	-/-	
MAXIMIZABLE	X	X/-	X
MAXIMIZED	X	X/X	X
MENU-HANDLE	X	X/X	
MINIMIZABLE	X	X/-	X
MINIMIZED	X	X/X	X
MODIFIED	X	X/X	
MODIFIED-SUCCESSOR	X	-/-	
NAME	X	-/-	
PARENT	X	X/-	
POPUP-HELP	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
RTL	X	-/-	X
SCROLLRANGE-X	X	X/X	
SCROLLRANGE-Y	X	X/X	
SIZE-MODIFIABLE	X	-/-	X
STATUS-HANDLE	X	X/X	
STATUS-TEXT		-/X	
STRING	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
SUPPRESS-CLIENT-SIZE-EVENT	X	X/X	
SUPPRESS-COMMAND-STATUS-EVENT	X	X/X	
SUPPRESS-DRAG-DROP-EVENT	X	X/X	
SUPPRESS-DRAG-ENTER-EVENT	X	X/X	
SUPPRESS-DRAG-LEAVE-EVENT	X	X/X	
SUPPRESS-DRAG-OVER-EVENT	X	X/X	
SUPPRESS-ENTER-EVENT	X	X/X	
SUPPRESS-IDLE-EVENT	X	X/X	
SUPPRESS-LEAVE-EVENT	X	X/X	
SUPPRESS-SIZE-EVENT	X	X/X	
TOOLBAR-HANDLE	X	X/-	
TOOLBAR-POS	X	X/-	X
TOOLTIP	X	X/X	
TYPE	X	X/-	X
VERSION	X	-/-	
VERT-SCROLLABLE	X	X/X	X
VISIBLE	X	X/X	X
WALLPAPER	X	X/X	X

Events

After-Any Event, **After-Open Event**, **Before-Any Event**, **Before Open Event**, **Before-Open Event**, **Client-Size Event** (may be suppressed), **Close Event**, **Command-Status Event** (may be suppressed), **Drag-Drop Event** (may be suppressed), **Drag-Enter Event** (may be suppressed), **Drag-Leave Event** (may be suppressed), **Drag-Over Event** (may be suppressed), **Enter Event** (may be suppressed), **Error Event**, **Idle Event** (may be suppressed), **Leave Event** (may be suppressed), **Size Event** (may be suppressed).

43 Submenu Control

- Description 172
- Attributes for Submenu Control 172
- Events 173

Description

A submenu control is a part of a complex menu structure. When the end user chooses a menu item of type submenu, the submenu control is pulled down, containing a vertical list of menu items.

You create a submenu control in the dialog editor by creating a menu item with the **MENU-ITEM-TYPE** of MT-SUBMENU or MT-WINDOWMENU. In Natural code, a submenu control is embedded into a menu structure by assigning its HANDLE to the **MENU-HANDLE** attribute of a menu item.

An MT-WINDOWMENU submenu control is used for MDI frame and MDI child windows. The windowing system adds the currently existing MDI child windows dynamically to such a submenu control.

The PARENT of a submenu control may be a dialog or NULL-HANDLE. If a dialog is the PARENT, the submenu control is private to the dialog and is deleted when the dialog is closed. If a NULL-HANDLE is the PARENT, the submenu control is free and will be deleted when the application is closed. A free submenu control can be shared among several menu structures in several dialogs if the handles are defined in a global data area rather than in a local data area.

Attributes for Submenu Control

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
FIRST-CHILD	X	-/-	
HELP-ID	X	X/X	
LAST-CHILD	X	-/-	
MENU-ITEM-OLE	X	X/X	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
STYLE	X	X/-	
SUCCESSOR	X	-/-	
SUPPRESS-BEFORE-OPEN-EVENT	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	

Events

Before Open Event (may be suppressed).

44 Tab Control

- Description 176
- Attributes for Tab Control 176
- Events 177

Description

A tab control is a container control consisting of a display area and (optionally) a set of tabs that are typically used to switch between groups of controls within the display area. Tab controls are often used to categorize object properties or user options.

Attributes for Tab Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	X
CONTEXT-MENU	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
HAS-TOOLTIP	X	X/X	X
HELP-ID	X	X/X	X
ITEM-H	X	X/-	X
ITEM-W	X	X/-	X
LAST-CHILD	X	-/-	
MARGIN-X	X	X/-	X
MARGIN-Y	X	X/-	X
MODIFIABLE	X	X/X	X
OWNER	X	X/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
ROW-COUNT	X	-/-	
SELECTED-SUCCESSOR	X	-/-	
STYLE	X	X/-	X
SUCCESSOR	X	X/-	
SUPPRESS-CHANGE-EVENT	X	X/X	
TOOLTIP	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X
WALLPAPER	X	X/X	X

Events

Change Event (may be suppressed).

45 Tab Control Tab

- Description 180
- Attributes for Tab Control Tab 180
- Events 180

Description

A tab control tab represents an individual tab within a tab control. A tab control tab can consist of a caption and/or icon, and can optionally display tool tip text.

Attributes for Tab Control Tab

Attribute Name	Query	Set/Modify	In Attr. Window
BITMAP-FILE-NAME	X	X/-	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
FIRST-CHILD	X	-/-	
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
SELECTED	X	X/X	X
STRING	X	X/X	X
SUCCESSOR	X	X/-	
SUPPRESS-ENTER-EVENT	X	X/X	
SUPPRESS-LEAVE-EVENT	X	X/X	
TOOLTIP	X	X/X	X
TYPE	X	X/-	X
VISIBLE	X	X/X	X

Events

Enter Event (may be suppressed), **Leave Event** (may be suppressed).

46 Table Control

- Description 182
- Attributes for Table Control 182
- Attributes for Cells in a Table Control 184
- COLUMN-TYPEs in Cells and their Attributes 184
- Events 185

Description

A table control is a dialog element representing a spreadsheet. It provides an optional header row for naming the columns and an optional left column for naming the rows.

Once the table control has defined the spreadsheet as such, the columns are defined by adding column-specification controls. The cells of the table control are not defined as dialog elements but are identified by the values of the table control's **COLUMN** and **ROW** attributes. The first cell is identified by ROW =1 and COLUMN =1. The cells of each column may be used differently depending on the value of the column specification control's **COLUMN-TYPE** attribute: they may be used as input fields, selection boxes with items or toggle buttons.

You manipulate table controls with a set of PROCESS GUI statement actions.

Attributes for Table Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CELL-ATTRIBUTES	X	X/X	
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
COLUMN	X	-/X	
COLUMN-COUNT	X	X/-	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FIRST-COLUMN-WIDTH	X	X/X	X
FIRST-VISIBLE-COLUMN	X	X/X	X
FIRST-VISIBLE-ROW	X	X/X	X
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
FROZEN-COLUMNS	X	X/X	X
HAS-FIRST-COLUMN	X	X/-	X
HEADER-FONT-HANDLE	X	X/X	X
HEADER-HEIGHT	X	X/X	X
HELP-ID	X	X/X	X
HORIZ-SCROLLABLE	X	X/-	X
LAST-CHILD	X	-/-	
MODIFIABLE	X	X/-	X
MODIFIED	X	-/X	
MODIFIED-SUCCESSOR	X	-/-	
OWNER	X	X/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
ROW	X	-/X	
ROW-COUNT	X	X/-	X
ROW-HEIGHT	X	X/X	X
RTL	X	-/-	X
STYLE	X	X/-	X
SUCCESSOR	X	X/-	
SUPPRESS-CHANGE-EVENT	X	-/-	
SUPPRESS-CLICK-EVENT	X	X/X	
SUPPRESS-DBL-CLICK-EVENT	X	X/X	
SUPPRESS-DELETE-ROW-EVENT	X	X/X	
SUPPRESS-ENTER-EVENT	X	X/X	
SUPPRESS-ENTER-CELL-EVENT	X	X/X	
SUPPRESS-FILL-EVENT	X	X/X	
SUPPRESS-INSERT-ROW-EVENT	X	X/X	
SUPPRESS-LEAVE-EVENT	X	X/X	
SUPPRESS-LEAVE-CELL-EVENT	X	X/X	
SUPPRESS-TOP-EVENT	X	X/X	
TOOLTIP	X	X/X	
TYPE	X	X/-	

Attribute Name	Query	Set/Modify	In Attr. Window
VERT-SCROLLABLE	X	X/-	X
VISIBLE	X	X/X	X

Attributes for Cells in a Table Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	
BACKGROUND-COLOUR-VALUE	X	X/X	
CHECKED	X	X/X	
COLUMN	X	X/X	
DIL-TEXT	X	X/X	
FOREGROUND-COLOUR-NAME	X	X/X	
FOREGROUND-COLOUR-VALUE	X	X/X	
ROW	X	X/X	
STRING	X	X/X	

COLUMN-TYPEs in Cells and their Attributes

The attributes that are valid for cells in table controls in general are not always available depending on the value of the column specification control's attribute COLUMN-TYPE. Such a value might be, for example, “input field control”. The cell can then be used like an input field control. For this input-field cell, only a subset of the attributes is available.

The following table specifies which attributes are *not* applicable to specific COLUMN-TYPE.

COLUMN-TYPE	Attributes NOT Available
Input field control	CHECKED
Selection box control	CHECKED
Toggle button control	STRING

Events

Change Event, **Click Event**, **Delete Row Event**, **Double Click Event**, **Enter Event**, **Enter-Cell Event**, **Fill Event**, **Insert-Row Event**, **Leave Event**, **Leave-Cell Event**, **Top Event** (all events may be suppressed).

The following events are not always triggered:

- click event - only for toggle-button fields,
- dbl-click event - for input-fields and toggle-buttons,
- change event - only for input-fields and for selection-boxes.

47 Text Constant Control

- Description 188
- Attributes for Text Constant Control 188
- Events 189

Description

A text constant control displays non-modifiable text within a dialog. You use it, for example, to describe which data can be entered in an adjacent input field control. The text STRING in text constant controls may also contain a mnemonic key (&). If the end user presses the mnemonic key, the next input dialog element in the navigation sequence gets the focus.

Attributes for Text Constant Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
LAST-CHILD	X	-/-	
OWNER	X	X/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STRING	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

This dialog element does not create events.

48 Timer Control

- Description 192
- Attributes for Timer 192
- Events 193

Description

A timer is a dialog element which is invisible to the end user but which allows events to be triggered periodically in the dialog. You can, for example, update the **STRING** attribute of a text constant control with the **Click Event** of a timer.

Example:

```
#TC-1.STRING:= *TIMX
/* Display the system time in the text-constant
/* control #TC-1 and  update the time (= timer /* control click event) every 1000 ↩
milliseconds
/* (TIMER-INTERVAL attribute).
```

Note: It is not recommended to write code that counts clicks in the timer click-event handler, because the click events that occur while the system is busy may get lost.

Attributes for Timer

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
FIRST-CHILD	X	-/-	
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
SUCCESSOR	X	-/-	
SUPPRESS-CLICK-EVENT	X	X/X	
TIMER-INTERVAL	X	X/X	
TYPE	X	X/-	

Events

Click Event (may be suppressed).

49 Toggle Button Control

- Description 196
- Attributes for Toggle Button Control 196
- Events 197

Description

A toggle button control offers the end user a checked/not checked (on/off) alternative. You use it, for example, to let the end user switch an option on or off without affecting the status of other adjacent toggle button controls. A change in the state of a **Radio Button Control**, by contrast, will affect the state of other radio button controls that belong to the same group.

Attributes for Toggle Button Control

Attribute Name	Query	Set/Modify	In Attr. Window
ACCELERATOR	X	X/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CHECKED	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
MODIFIED	X	-/X	
MODIFIED-SUCCESSOR	X	-/-	
OWNER	X	X/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
RTL	X	-/-	X
STRING	X	X/X	X
SUCCESSOR	X	-/-	
SUPPRESS-CLICK-EVENT	X	X/X	
TOOLTIP	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

Click Event (may be suppressed).

50 Tool Bar

- Description 200
- Attributes for Tool Bar 200
- Events 201

Description

A tool bar is similar to a menu bar. It consists of horizontally or vertically grouped tool bar items with a bitmap on each of them, providing quick access to the most frequently needed program functions. By default, it is located at the top of the dialog, directly below the menu bar. It can also be located at the bottom, the left or the right of the dialog window.

A tool bar only becomes visible within a dialog when its handle value is assigned to the dialog's **TOOLBAR-HANDLE** attribute and **HAS-TOOLBAR** is TRUE for the dialog. When you create the tool bar with the dialog editor, the dialog editor does this automatically. For MDI child windows, the tool bar is displayed at the top of the MDI frame window rather than in the child window itself. Any time another MDI child window is activated, the tool bar changes to reflect the tool bar defined for the particular MDI child window.

You can determine the size of the tool bar items using the **ITEM-W** and **ITEM-H** attributes. The distance between the items and the tool bar's border is determined with the **MARGIN-X** and **MARGIN-Y** attributes. If you set the tool bar's style to "w", the items that would exceed the tool bar's width are wrapped around.

Attributes for Tool Bar

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	X
CLIENT-VALUE	X	X/X	
FIRST-CHILD	X	-/-	
ITEM-H	X	X/X	X
ITEM-W	X	X/X	X
LAST-CHILD	X	-/-	
MARGIN-X	X	X/X	X
MARGIN-Y	X	X/X	X
PARENT	X	X/-	
PREDECESSOR	X	-/-	
STYLE	X	X/X	X
SUCCESSOR	X	-/-	
TYPE	X	X/-	

Events

This dialog element does not create events.

51 Tool Bar Control

- Description 204
- Attributes for Tool Bar Control 204
- Events 205

Description

A tool bar control is an alternative to the traditional tool bar described above. It, too, can contain one or more tool bar items used to trigger specific application functionality. However, the tool bar control offers a range of advanced features which the traditional Natural tool bars do not support. These new features include a Windows-like appearance with (optionally) flat buttons, the ability to specify that the tool bar should be "dockable" (see below), the ability to embed other dialog elements (such as selection-boxes), and the ability to define tool tips for the tool bar items.

A tool bar control is dockable if its **DRAGGABLE** attribute is set to TRUE when the control is created. Dockable tool bars have the advantage that they can be repositioned by the user by tearing off the tool bar control via clicking on the gripper (see illustration above), a separator or any part of its background area and dragging it with the primary mouse button held down. When the tool bar control is dropped, it will either be snapped into position at the side of the dialog ("docked"), or "floated" in its own window. Whether the control is floated or docked depends on where the control was dropped, whether the dialog and tool bar control both allow docking on the target side and whether the CTRL key is being held down. In the latter case, the tool bar will be floated regardless of its drop position. The border width of the drag rectangle informs the user as to whether the tool bar control will be floated (thick border) or docked (thin border). The dragging process can be aborted at any time by pressing the ESC key.

A further major benefit of using tool bar controls instead of the traditional tool bars is that multiple tool bar controls can be used within a dialog, which can be independently docked or floated as required. The layout can be preserved on a per-user basis between sessions by activating the "save layout" option in the Dialog Attributes window in the dialog editor.

You can determine the size of the tool bar items using the **ITEM-W** and **ITEM-H** attributes. The distance between the items and the tool bar control's border is determined with the **MARGIN-X** and **MARGIN-Y** attributes.

Attributes for Tool Bar Control

Attribute Name	Query	Set/Modify	In Attr. Window
BAR-ID	X	X/-	
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	X
CONTEXT-MENU	X	X/X	X
DOCKING	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
DRAGGABLE	X	X/-	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
HAS-DIL	X	X/X	X
HAS-TOOLTIP	X	X/X	X
ITEM-H	X	X/-	X
ITEM-W	X	X/-	X
LAST-CHILD	X	-/-	
LOCATION	X	X/X	X
MARGIN-X	X	X/X	X
MARGIN-Y	X	X/X	X
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/-	X
RECTANGLE-W	X	X/-	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STRING	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	X/-	
SUPPRESS-CLOSE-EVENT	X	X/X	
TOOLTIP	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

Close Event (may be suppressed).

52 Tool Bar Item

- Description 208
- Attributes for Tool Bar Item 208
- Events 209

Description

A tool bar item is an item inside a tool bar. A tool bar item is similar to a push button with a bitmap displayed on it. In most cases, a tool bar item will serve as an alias for a menu item. So, instead of programming again the same click-event handler code as for the menu item, you can use the tool bar item's **SAME-AS** attribute and assign the corresponding menu item handle value. If the tool bar item is clicked, the event handler of the corresponding menu item is triggered. If you do not use the **SAME-AS** attribute, you must specify the event handler code with the tool bar item.

You can determine the size of the tool bar items using the parent tool bar's **ITEM-W** and **ITEM-H** attributes.

Note: For performance reasons, you should not use more than 16 colors in tool bar icon bitmaps; all bitmaps should share the same palette of colors.

Attributes for Tool Bar Item

Attribute Name	Query	Set/Modify	In Attr. Window
ACCELERATOR	X	X/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
BITMAP-FILE-NAME	X	X/X	X
CHECKED	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
DIL-TEXT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
LAST-CHILD	X	-/-	
MENU-ITEM-TYPE	X	X/-	X
PARENT	X	X/-	
PREDECESSOR	X	-/-	
SAME-AS	X	X/X	X
SHARED	X	X/X	X
STYLE	X	X/-	X

Attribute Name	Query	Set/Modify	In Attr. Window
SUCCESSOR	X	X/-	
TOOLTIP	X	X/X	X
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

Click Event.

53 Tree View Control

- Description 212
- Attributes for Tree View Control 212
- Events 214

Description

A tree view control provides a hierarchical item list, as used for displaying library contents within the library workspace window in Natural Studio, for example.

For more information on tree view controls, please refer to the article Working with Tree View Controls.

Attributes for Tree View Control

Attribute Name	Query	Set/Modify	In Attr. Window
ACCELERATOR	X	X/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
DRAG-MODE	X	X/X	X
DROP-MODE	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FIRST-VISIBLE-ITEM	X	-/X	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
HAS-TOOLTIP	X	X/X	X
HELP-ID	X	X/X	X
IMAGE-LIST	X	X/X	X
ITEM	X	X/X	X
ITEM-H	X	X/X	X
LAST-CHILD	X	-/-	

Attribute Name	Query	Set/Modify	In Attr. Window
MODIFIABLE	X	X/X	X
OWNER	X	X/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
SELECTED-SUCCESSOR	X	-/-	
SORTED	X	X/-	X
SPACING	X	X/-	X
STYLE	X	X/-	X
SUCCESSOR	X	X/-	
SUPPRESS-ACTIVATE-EVENT	X	X/X	
SUPPRESS-AFTER-EDIT-EVENT	X	X/X	
SUPPRESS-BEFORE-EDIT-EVENT	X	X/X	
SUPPRESS-BEGIN-DRAG-EVENT	X	X/X	
SUPPRESS-CHECK-EVENT	X	X/X	
SUPPRESS-CLICK-EVENT	X	X/X	
SUPPRESS-CLIPBOARD-STATUS-EVENT	X	X/X	
SUPPRESS-COLLAPSE-EVENT	X	X/X	
SUPPRESS-CONTEXT-MENU-EVENT	X	X/X	
SUPPRESS-COPY-EVENT	X	X/X	
SUPPRESS-CUT-EVENT	X	X/X	
SUPPRESS-DELETE-EVENT	X	X/X	
SUPPRESS-DRAG-DROP-EVENT	X	X/X	
SUPPRESS-DRAG-ENTER-EVENT	X	X/X	
SUPPRESS-DRAG-LEAVE-EVENT	X	X/X	
SUPPRESS-DRAG-OVER-EVENT	X	X/X	
SUPPRESS-END-DRAG-EVENT	X	X/X	
SUPPRESS-EXPAND-EVENT	X	X/X	
SUPPRESS-PASTE-EVENT	X	X/X	
SUPPRESS-UNDO-EVENT	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

Events

Activate Event, **After-Edit Event**, **Before-Edit Event**, **Begin-Drag Event**, **Check Event**, **Click Event**, **Clipboard-Status Event**, **Collapse Event**, **Context-Menu Event**, **Copy Event**, **Cut Event**, **Delete Event**, **Drag-Drop Event**, **Drag-Enter Event**, **Drag-Leave Event**, **Drag-Over Event**, **End-Drag Event**, **Expand Event**, **Paste Event**, **Undo Event** (all events may be suppressed).

54 Tree View Item

- Description 216
- Attributes for Tree View Item 216
- Events 217

Description

A tree view item represents a node displayed in a **tree view control**.

For more information on list view items, please refer to the article Working with Tree View Controls.

Attributes for Tree View Item

Attribute Name	Query	Set/Modify	In Attr. Window
CHECKED	X	X/X	X
CHECKED-SUCCESSOR	X	-/-	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
EDIT-MASK	X	X/-	X
EXPANDED	X	X/-	X
FIRST-CHILD	X	-/-	
FORMAT	X	X/-	X
IMAGE	X	X/X	X
IMAGE-INDEX	X	X/X	X
LAST-CHILD	X	-/-	
LENGTH	X	X/X	X
MODIFIABLE	X	X/X	X
OVERLAY	X	X/X	X
OVERLAY-INDEX	X	X/X	X
PARENT	X	X/-	
PREDECESSOR	X	-/-	
SELECTED	X	X/X	
STRING	X	X/X	X
STYLE	X	X/X	X
SUCCESSOR	X	X/-	
TOOLTIP	X	X/X	X
TYPE	X	X/-	

Events

This dialog element does not create events.

55 Wallpaper Control

- Description 220
- Attributes for Wallpaper Control 220
- Events 221

Description

A wallpaper control is defines a background that can be displayed by dialogs or certain dialog elements (e.g. control box and dialog bar). A wallpaper control specifies an image together with associated attributes that determine how the wallpaper is to be rendered. For example, it is possible to specify that the wallpaper is to be rendered as a repeated pattern (tiling) rather than as a single image. Or, in the latter case, the horizontal and vertical justification of the image relative to the host dialog or dialog element can be set.

A wallpaper image is drawn transparently if the wallpaper control's “transparent” style is set, implying that all pixels in the wallpaper control's background color (if any) are not drawn. If no explicit background color is set, the transparent color is assumed to be the color of the top-left pixel in the image.

To use a wallpaper to paint the background of a dialog or dialog element, the **WALLPAPER** attribute of that dialog or dialog element must be set to the handle of the desired wallpaper control. Natural keeps track of the objects using a particular wallpaper control, and updates each of these objects automatically if any changes are made to the wallpaper control, or if the wallpaper control is deleted. In the latter case, the **WALLPAPER** attribute of each object is automatically reset to NULL-HANDLE.

Attributes for Wallpaper Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
BITMAP-FILE-NAME	X	X/X	X
BLEND	X	X/-	X
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
FIRST-CHILD	X	-/-	
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
STYLE	X	X/-	X
SUCCESSOR	X	X/-	
TYPE	X	X/-	

Attribute Name	Query	Set/Modify	In Attr. Window
VISIBLE	X	X/X	X

Events

This dialog element does not create events.

II Attributes

Note: The names of the alphanumeric attributes below can be suffixed with a "-U" to support characters that are not available in the active code page. See Querying and Modifying Unicode Attribute Values in the *Event-Driven Programming Techniques* documentation for more information.

This part covers the following topics:

ACCELERATOR
ACTIVE-CHILD
AUTO-ADJUST
AUTOSELECT
BACKGROUND-COLOUR-NAME
BACKGROUND-COLOUR-VALUE
BAR-ID
BITMAP-FILE-NAME
BLEND
BUDDY
CELL-ATTRIBUTES
CHECKED
CHECKED-SUCCESSOR
CLIENT-DATA
CLIENT-HANDLE
CLIENT-KEY
CLIENT-VALUE
COLUMN
COLUMN-COUNT
COLUMN-TYPE
COMPATIBILITY

CONTEXT MENU
CONTROL
DEFAULT-BUTTON
DESCENDING
DIL-TEXT
DOCKING
DPI
DRAG-MODE
DRAGGABLE
DROP-MODE
EDIT-MASK
EMBEDDED-OBJECT
ENABLED
EVENT-QUEUEING
EXPANDED
FIRST-CHILD
FIRST-COLUMN-WIDTH
FIRST-VISIBLE-COLUMN
FIRST-VISIBLE-ITEM
FIRST-VISIBLE-ROW
FOLLOWS
FONT-HANDLE
FONT-STRING
FOREGROUND-COLOUR-NAME
FOREGROUND-COLOUR-VALUE
FORMAT
FROZEN-COLUMNS
GROUP-ID
HANDLE-VARIABLE
HAS-DIL
HAS-FIRST-COLUMN
HAS-HELP-BUTTON
HAS-MENU-BAR
HAS-STATUS-BAR
HAS-SYSTEM-BUTTON
HAS-TOOLBAR
HAS-TOOLTIP
HEADER-FONT-HANDLE

HEADER-HEIGHT
HELP-FILENAME
HELP-ID
HORIZ-SCROLLABLE
ICONIZED
IMAGE
IMAGE-INDEX
IMAGE-LIST
INPLACE-ACTIVE
ITEM
ITEM-H
ITEM-W
LAST-CHILD
LENGTH
LINE
LINKED
LOCATION
MARGIN-X
MARGIN-Y
MAX
MAXIMIZABLE
MAXIMIZED
MENU-HANDLE
MENU-ITEM-OLE
MENU-ITEM-TYPE
MIN
MINIMIZABLE
MINIMIZED
MODIFIED-SUCCESSOR
MODIFIABLE
MODIFIED
MULTI-SELECTION
NAME
OBJECT-SIZE
OFFSET-X
OFFSET-Y
OVERLAY
OVERLAY-INDEX

OWNER
P1-X
P1-Y
P2-X
P2-Y
PAGE
PARENT
POSITION
POPUP-HELP
PREDECESSOR
RECTANGLE-H
RECTANGLE-W
RECTANGLE-X
RECTANGLE-Y
ROW
ROW-COUNT
ROW-HEIGHT
RTL
SAME-AS
SCROLLRANGE-X
SCROLLRANGE-Y
SELECTED-SUCCESSOR
SELECTED
SHARED
SIZE-MODIFIABLE
SLIDER
SORTED
SERVER-OBJECT
SERVER-PROGID
SPACING
SPACING-X
SPACING-Y
STATUS-HANDLE
STATUS-TEXT
STRING
STYLE
SUCCESSOR
SUPPRESS-AFTER-EDIT-EVENT

SUPPRESS-BEFORE-EDIT-EVENT
SUPPRESS-BEFORE-OPEN-EVENT
SUPPRESS-BEGIN-DRAG-EVENT
SUPPRESS-CLIENT-SIZE-EVENT
SUPPRESS-DBL-CLICK-EVENT
SUPPRESS-DELETE-ROW-EVENT
SUPPRESS-INSERT-ROW-EVENT
SUPPRESS-TOP-EVENT
SUPPRESS-ACTIVATE-EVENT
SUPPRESS-CHANGE-EVENT
SUPPRESS-CHECK-EVENT
SUPPRESS-CLIPBOARD-STATUS-EVENT
SUPPRESS-CLICK-EVENT
SUPPRESS-CLOSE-EVENT
SUPPRESS-COLLAPSE-EVENT
SUPPRESS-COMMAND-STATUS-EVENT
SUPPRESS-CONTEXT-MENU-EVENT
SUPPRESS-COPY-EVENT
SUPPRESS-CUT-EVENT
SUPPRESS-DELETE-EVENT
SUPPRESS-DRAG-DROP-EVENT
SUPPRESS-DRAG-ENTER-EVENT
SUPPRESS-DRAG-LEAVE-EVENT
SUPPRESS-DRAG-OVER-EVENT
SUPPRESS-END-DRAG-EVENT
SUPPRESS-ENTER-CELL-EVENT
SUPPRESS-EXPAND-EVENT
SUPPRESS-ENTER-EVENT
SUPPRESS-FILL-EVENT
SUPPRESS-IDLE-EVENT
SUPPRESS-LEAVE-CELL-EVENT
SUPPRESS-LEAVE-EVENT
SUPPRESS-PASTE-EVENT
SUPPRESS-SIZE-EVENT
SUPPRESS-UNDO-EVENT
TIME
TIMER-INTERVAL
TOOLBAR-HANDLE

TOOLBAR-POS
TOOLTIP
TYPE
VARIABLE
VERSION
VERT-SCROLLABLE
VIEW-MODE
VISIBLE
WALLPAPER
ZOOM-FACTOR

56 ACCELERATOR

Enables you to define accelerator keys. If the end user presses an accelerator key, the double-click event occurs for the dialog element, or if no double-click event is available, the click event occurs. The accelerator key does not work if the corresponding event is suppressed.

Note: User-defined accelerator keys overwrite identical operating system accelerator keys, with the exception of hot keys such as Alt+Tab, Ctrl+Esc, Ctrl+Alt+Del, etc.

Applies to	Bitmap control, list box control, list view control, menu item, OLE container control, push button control, radio button control, signal, tree view control, toggle button control, tool bar item.
Data Type	A253
Default Value	None
Possible Values	The standard syntax is: `[Ctrl+][Alt+][Shift+]<key>` where <key> can be any of the following values: “A” to “Z” “0” to “9” (but not in conjunction with shift key) “F1” to “F12” (but not F1 on its own) “Num 0” to “Num 9” (=numeric keypad number keys) “Num Dec” (=numeric keypad decimal point key) “Num +” (=numeric keypad add key) “Num -” (=numeric keypad subtract key) “Num *” (=numeric keypad multiply key) “Num /” (=numeric keypad divide key) “Space” “Backspace” “Tab” “Enter” (=return key)

	“Esc” (=escape key) “Ins” (=insert key) “Del” (=delete key) “Home” “End” “Page Up” or “Page Down” “Left Arrow”, “Right Arrow”, “Up Arrow” or “Down Arrow” Examples: “3”, “F7”, “Shift+Page Up”, “Alt+Enter”, “Ctrl+Alt+G”. Note that the key modifiers (“Ctrl”, “Alt” and “Shift”) can be specified in any order. Accelerators are not case-sensitive. Please keep in mind that F1 cannot be an accelerator and therefore help topics cannot be invoked this way.

57 ACTIVE-CHILD

Specifies the active MDI child dialog (if any) for the specified MDI frame dialog.

Applies to	Dialog (MDI frame).
Data Type	HANDLE
Default Value	None
Possible Values	NULL-HANDLE / MDI child dialog handle

58 AUTO-ADJUST

Indicates whether the coordinates specified on dialog element creation are to be scaled in the case of scalable dialogs (i.e., dialogs created with a non-zero **DPI** attribute). This is useful for dynamically creating controls in scalable dialogs whose size and/or position is based on an existing dialog element.

Applies to	Dialog (all types).
Data Type	BOOLEAN
Default Value	TRUE for scalable dialogs; FALSE otherwise.
Possible Values	TRUE / FALSE

59 AUTOSELECT

Specifies whether the list box item under the mouse cursor is to be selected automatically, if not already selected, before a context menu is displayed. If the list box item is not already selected, the existing selection (if any) is cleared before selecting it in the case of multi-selection list boxes. Otherwise the existing selection is left intact.

Applies to	List box control.
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	TRUE / FALSE

60 BACKGROUND-COLOUR-NAME

Provides a choice of existing background colors.

When setting background colors, you have four possibilities:

- Use the BACKGROUND-COLOUR-NAME attribute and leave the value at DEFAULT (0). The color will then be determined by your color settings in the operating system.
- Use the BACKGROUND-COLOUR-NAME attribute and choose from one of the dialog editor's predefined colors. These are represented by the values 1 to 18.
- Define your own color by using the **BACKGROUND-COLOUR-VALUE** attribute in a dialog editor attributes window. To do so, you must first set the BACKGROUND-COLOUR-NAME attribute to CUSTOM (50). A dialog box appears where you can set values for the red, green, and blue elements of your color (RGB model).
- Set the color dynamically in event handler code by assigning a value to the attribute.

Note: Due to consistency constraints imposed by Windows, the specified setting may not be effective for particular dialog element types, depending on the version of Windows in use and whether visual themes are enabled.

Applies to	Canvas control, control box control, dialog (all types), date/time picker control, dialog bar control, edit area control, group frame control, image control, input field control, list box control, list view control, OLE container control, progress bar control, push button control, radio button control, rectangle control, scroll bar control, selection box control, signal, tab control, table control, text constant control, toggle button control, tree view control, wallpaper control.
Data Type	I4
Default Value	DEFAULT (0)
Possible Values	See table below.

Possible Values

DEFAULT (0)	WHITE (1)	BLACK (2)
LTGRAY (3)	GRAY (4)	DKGRAY (5)
RED (6)	GREEN (7)	BLUE (8)
CYAN (9)	MAGENTA (10)	BROWN (11)
YELLOW (12)	LIGHTRED (13)	LIGHTGREEN (14)
LIGHTBLUE (15)	LIGHTCYAN (16)	LIGHTMAGENTA (17)
BRIGHTWHITE (18)	CUSTOM (50)	

Note: The text representation can be substituted for the integer representation because the local data area NGULKEY1 (supplied in library SYSTEM) is automatically included in the event handler code.

61 BACKGROUND-COLOUR-VALUE

Provides a facility to define customized background colors.

When setting background colors, you have four possibilities:

- Define your own color by using the BACKGROUND-COLOUR-VALUE attribute in a dialog editor Attributes window. To do so, you must first set the **BACKGROUND-COLOUR-NAME** attribute to CUSTOM (50). A dialog box appears where you can set values for the red, green, and blue elements of your color (RGB model).
- Use the **BACKGROUND-COLOUR-NAME** attribute and leave the value at DEFAULT (0). Your color will then be determined by your color settings in the operating system.
- Use the BACKGROUND-COLOUR-NAME attribute and choose from one of the dialog editor's predefined colors. These are represented by the values 1 to 18.
- Set the color dynamically in event handler code by assigning a set of three RGB values to the attribute. To do so, you must first set the **BACKGROUND-COLOUR-NAME** attribute to CUSTOM (50). You enter the three RGB values in hexadecimal form, such as "H'FF0000'.

Note: Due to consistency constraints imposed by Windows, the specified setting may not be effective for particular dialog element types, depending on the version of Windows in use and whether visual themes are enabled.

Applies to	Canvas control, control box control, dialog (all types), date/time picker control, dialog bar control, edit area control, group frame control, image control, input field control, list box control, list view control, OLE container control, progress bar control, push button control, radio button control, rectangle control, scroll bar control, selection box control, signal, tab control, table control, text constant control, toggle button control, tree view control, wallpaper control.
Data Type	B3
Default Value	None
Possible Values	Any red, green, and blue value from 1 to 253

62 BAR-ID

Specifies an identifier for a status bar or tool bar control. The value is used to identify the control between sessions when the bar layout is saved and restored via the **SAVE-LAYOUT** and **LOAD-LAYOUT** actions, and must therefore be unique within a dialog.

Applies to	Dialog bar control, status bar control, tool bar control.
Data Type	I4
Default Value	0
Possible Values	0 to 250

63 BITMAP-FILE-NAME

Assigns an image file to a dialog or dialog element. If no path information is supplied, the image file is searched for in the logon library's RES subdirectory first, then in the RES subdirectory of each steplib, then in the directory assigned to the environment variable NATGUI_BMP. If no image file with this name exists in any of the search paths, the file “default.bmp” or “default.ico” (for dialogs) is searched for, using the same search sequence. If even this search fails, a hard-coded default image stored as an executable file resource is used.

For dialogs, tab control tabs and status bar panes, the image file must be an icon file (*.ico), which is used to provide the dialog's icons.

For image controls, the image file may be either a bitmap file (*.bmp), a JPEG file (*.jpg), a GIF file (*.gif) or an icon file, which is used to provide the image control's image(s). By default, the image file is assumed to contain a single image. However, if the image file is a bitmap file and the image control's “Composite image (C)” **STYLE** is set, the bitmap is assumed to contain multiple images joined together into one big composite image, the size of the individual images being determined by the image list's **ITEM-W** and **ITEM-H** attributes. For more information, please refer to the article Working with Image List Controls.

For all other dialog elements, the image file must be a bitmap file, JPEG file or GIF file, which is assumed to consist of a single image.

Applies to	Dialogs (all types), bitmap control, image control, menu item, signal, status bar pane, tab control tab, tool bar item, wallpaper control.
Data Type	A253
Default Value	Empty string
Possible Values	Empty string, or file name of up to 253 alphanumeric characters

64 BLEND

Specifies the alpha blending factor used for image drawing.

This can be used for wallpaper controls to optionally reduce the image contrast, and hence improve the visibility of the controls placed on the wallpaper.

The alpha blending value is specified as a percentage. Each of the red, green and blue color components of the wallpaper image is separately blended with the corresponding value of the host control's background color according to the formula:

Color(blended) = (BlendFactor * Color(host)) + ((100 - BlendFactor) * Color(wallpaper)

where *Color* is the value (0 - 255) of the color component (red, green or blue), *BlendFactor* is the value of the BLEND attribute, and the host is the dialog or dialog element within which the wallpaper is being displayed.

Note that, from the above formula, when the alpha blending factor is set to 0, the contribution to the resulting color from the host control becomes zero and the wallpaper appears opaque. Similarly, when the alpha blending factor is set to 100, the contribution to the resulting color from the wallpaper becomes zero and the wallpaper is fully transparent and cannot be seen. Intermediate values make the wallpaper appear faded and translucent.

Applies to	Wallpaper control.
Data Type	I4
Default Value	0 (opaque)
Possible Values	0 - 100

65 BUDDY

Identifies a spin control's buddy input field control. A value of NULL-HANDLE indicates that the spin control has no associated buddy.

Note that spin controls with either the “Left align (l)” or “Right alight (r)” **STYLE** have implicitly-created buddy controls which cannot be changed. Hence an update to this attribute in this case is not allowed. Otherwise an update to this attribute is allowed only if the specified buddy has the same **PARENT** as the spin control.

If the control has the “Set buddy (s)” **STYLE**, the contents of the new buddy (if any) are automatically updated to reflect the current spin control **POSITION** when this attribute is set.

Applies to	Spin control.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE / handle of sibling input field control.

66 CELL-ATTRIBUTES

Determines whether the attributes **FOREGROUND-COLOUR-NAME**, **BACKGROUND-COLOUR-NAME**, **FOREGROUND-COLOUR-VALUE**, **BACKGROUND-COLOUR-VALUE** and **DIL-TEXT** of the table control apply to the whole table (FALSE) or to a single cell of the table (TRUE). If the above attributes apply to a cell, the **ROW** and **COLUMN** attribute of the table specify which cell is changed.

Applies to	Table Control
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	TRUE / FALSE

67 CHECKED

Determines whether a dialog element is checked (selected) or not.

For list view items and tree view items, the control must have the “Check boxes (c)” **STYLE**, whereupon check boxes are displayed for each item.

For date and time picker (DTP) controls, the control must have the “Allow 'no value' (n)” **STYLE**, in which case the control has a check box, indicating whether a value has been set. The program should check for this before trying to retrive the control's **TIME** value. On update, the CHECKED attribute is set implicitly when a value is assigned.

Applies to	Date/time picker control, list view item, menu item, radio button control, signal, table control, toggle button control, tool bar item, tree view item.
Data Type	I4
Default Value	UNCHECKED
Possible Values	UNCHECKED / CHECKED

Note: You can use the symbols CHECKED and UNCHECKED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

68 CHECKED-SUCCESSOR

Returns the first or next checked item.

If applied to a list view control, the handle of the first checked list view item (if any) is returned. If applied to a list view item or tree view item, the handle of the next checked item (if any) is returned. In the latter case, it is not necessary that the item to which this attribute is applied is itself checked.

The search sequence used for finding the first or next item in the case of list views is based on an internal index. For **SORTED** controls, this is alphabetic sequence. Otherwise it is the creation sequence (earliest first). Furthermore, if the items are sorted via the **SORT-ITEMS** action, the internal indexes are updated automatically to correspond to the sorted sequence. For tree views, the search sequence is identical to the visual sequence (i.e., the order in which the tree view items are displayed), and child items (if any) are not searched.

Note that check boxes are only displayed if the list or tree view control's “Check boxes (c)” **STYLE** is set.

For more information, please refer to the articles Working with List View Controls and Working with Tree View Controls.

Applies to	List view control, list view item, tree view item.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE / handle of list view item or tree view item.

69 CLIENT-DATA

Data associated with a dialog element. Often, it is useful to store information such as a database key or an array index with a dialog element. This allows for quick retrieval of the client data. If you use this attribute with a dialog (window, MDI frame, MDI child), Natural automatically assigns the dialog ID and you may therefore only query this attribute.

Applies to	ActiveX control, bitmap control, date/time picker control, canvas control, context menu, column specification control, control box control, dialog (all types), dialog bar control, edit area control, graphic text control, input field control, line control, list box control, list box item, list view control, menu bar, menu item, OLE container control, progress bar control, push button control, radio button control, rectangle control, scroll bar control, selection box control, selection box item, signal, status bar control, status bar pane, submenu control, slider control, spin control, tab control, tab control tab, table control, timer, tool bar control, toggle button control, tool bar, tool bar item, tree view control.
Data Type	I4
Default Value	0
Possible Values	Any I4 value

70 CLIENT-HANDLE

Specifies a general-purpose attribute that can be used to store the handle of any GUI object. This attribute is not used by Natural itself.

Applies to	Dialogs (all types) and all dialog elements, date/time picker control, dialog bar control, image control, image list control, list view column, list view control, list view item, progress bar control, slider control, spin control,tab control, tab control tab, tree view control, tree view item, wallpaper control.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE / any dialog or dialog element handle

71 CLIENT-KEY

Alphanumeric data associated with a dialog or a dialog element. Often, it is useful to store a text string with a dialog element to allow for direct access of the string. The **CLIENT-DATA** attribute enables you to assign an integer value to many dialog elements; the CLIENT-KEY attribute is used in conjunction with the attribute **CLIENT-VALUE** and enables you to store alphanumeric key/string pairs.

You first assign a value to the CLIENT-KEY attribute. This determines the key under which the string is to be stored for an instance of a dialog element. You then assign an alphanumeric string to the **CLIENT-VALUE** attribute of the dialog element.

It is advisable to reuse keys that are not needed because you may use only a limited number of keys.

You can query the **CLIENT-VALUE** of a dialog element by first setting the CLIENT-KEY attribute of the dialog element to the desired value and then accessing the **CLIENT-VALUE** attribute.

Applies to	Dialogs (all types) and all dialog elements, date/time picker control, image control, image list control, list view column, list view control, list view item, progress bar control, slider control, spin control, tree view control, tree view item.
Data Type	A253
Default Value	Empty string (no key/value pair will be assigned to the dialog element)
Possible Values	Empty string or any text string

72 CLIENT-VALUE

Alphanumeric data associated with a dialog or a dialog element. For a description of the context in which this attribute is used, see the attribute **CLIENT-KEY**.

Applies to	Dialogs (all types) and all dialog elements, date/time picker control, dialog bar control, image control, image list control, list view column, list view control, list view item, progress bar control, slider control, spin control, tab control tab, tree view control, tree view itemtab control, wallpaper control.
Data Type	A253
Default Value	Empty string (no key/value pair will be assigned to the dialog element)
Possible Values	Empty string or any text string

73 COLUMN

A COLUMN attribute value corresponds to exactly one **ROW** attribute value. The combination of these two denotes a cell inside a table control.

- To denote a cell of the table: COLUMN must be >= 1 and ROW must be >=1.
- To denote the first column in the table: COLUMN must be 0 and ROW must be 1.
- To denote the header row in the table: COLUMN must be 1 and ROW must be 0.

Applies to	Table control.
Data Type	I4
Default Value	1
Possible Values	0 - 9999

74 COLUMN-COUNT

Returns the current number of columns defined for a dialog element.

Applies to	List view control, table control.
Data Type	I4
Default Value	1
Possible Values	0 - 9999

75 COLUMN-TYPE

Determines the type of column in a column specification control (which itself is part of a table control). If, for example, the value is set to INPUTFIELD, all cells in this column behave as if they were input field controls.

Applies to	Column specification control.
Data Type	I4
Default Value	INPUTFIELD
Possible Values	INPUTFIELD / SELECTIONBOX / TOGGLEBUTTON

76 COMPATIBILITY

Indicates the Natural version (if any) to which the dialog should behave compatibly.

This attribute is currently relevant for dialogs containing edit area, input field or list box controls with 3-D borders. In Natural versions up to and including Version 2.2.3, the effective sizes of such controls was larger than the defined value (1 pixel on all sides). In later Natural versions, prior to Version 6.1.1, the effective sizes of such controls was smaller than the defined value (1 pixel on all sides). As from Natural Version 6.1.1, the controls appear in exactly the defined size, at exactly the defined position.

The use of the COMPATIBILITY attribute is intended to isolate applications from the effects of these changes. If a dialog last saved with a Natural version older than Version 6.1.1 is loaded into the dialog editor in Version 6.1.1 (or higher), and the dialog contains controls affected by the above border change, the dialog editor automatically sets the dialog's COMPATIBILITY attribute to the dialog's previous **VERSION** attribute. When the dialog is run, and the COMPATIBILITY attribute is set, Natural treats the coordinates passed by the program for the above controls as logical co-ordinates. The physical coordinates with which the control is created are based on the logical co-ordinates, but with an adjustment to ensure that the control appears with the same effective size and position as in the older Natural version. Similarly, when the control's coordinates are read, the logical coordinates are returned instead of the physical coordinates.

The relationship between logical and physical coordinates for the above control types are as follows:

Compatibility	Relationship
None	Logical and physical coordinates are identical.
5.1.1	Logical x-position (**RECTANGLE-X**) = Physical x-position - 1 Logical y-position (**RECTANGLE-Y**) = Physical y-position - 1 Logical width (**RECTANGLE-W**) = Physical width + 2 Logical height (**RECTANGLE-H**) = Physical height + 2
2.2.3	Logical x-position (**RECTANGLE-X**) = Physical x-position + 1

Compatibility	Relationship
	Logical y-position (**RECTANGLE-Y**) = Physical y-position + 1 Logical width (**RECTANGLE-W**) = Physical width - 2 Logical height (**RECTANGLE-H**) = Physical height - 2

It is important to realize that the above distinction between logical and physical coordinates only applies to controls for which a border upgrade is performed (edit areas, input fields and list boxes, for which the "3-D border (3)" **STYLE** is set). In all other cases, the logical and physical coordinates are identical, regardless of the above compatibility setting.

Note that "2.2.3" in the above table implies Version 2.2.3 or earlier, and "5.1.1" implies versions greater than 2.2.3 up to 5.1.1 (inclusive). Thus, if the COMPATIBILITY attribute is set to "3.1.1", the dialog behaves compatibly with Version 5.1.1. Any patch level number in the COMPATIBILITY attribute string is ignored.

Applies to	Dialogs (all types).
Data Type	A253
Default Value	Empty string
Possible Values	Empty string / Compatible Natural version information in encoded string format MMNNSSPP, where MM = major version, NN = minor version, SS = SM level and PP = patch level.

77 CONTEXT MENU

Specifies the handle of the context menu associated with the dialog or dialog element. A value of NULL-HANDLE indicates that no context menu assigned. The context menu specified here overrides any default context menu displayed by the system. Note that, for table controls, the specified context menu is not displayed for a cell which is being edited. For MDI frames, the context menu applies to the MDI client area.

Before this attribute is evaluated by Natural, the dialog or dialog element receives a **Context-Menu Event** (if not suppressed), allowing dynamic assignment to this attribute and, hence, dynamic context menu selection.

Note: For some ActiveX controls, depending on their internal implementation, the associated context menu may not be displayed when the user attempts to invoke it. However, if the control offers a MouseDown event (or equivalent), this event may be used to open a specific context menu explicitly via the **SHOW-CONTEXT-MENU** action.

For more information, please refer to the article Defining and Using Context Menus.

Applies to	ActiveX control, bitmap control, canvas control, date/time picker control, dialog (all types), dialog bar control, edit area control, input field control, list box control, list view control, push button control, progress bar control, radio button control, scroll bar control, selection box control, slider control, spin control, status bar control, tab control, table control, toggle button control, tool bar control, tree view control.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE / handle value of context menu.

78 CONTROL

Contains the handle of the right-clicked dialog or dialog element. This is typically used in the **Before-Open Event** to distinguish controls sharing the same context menu. Note that this is not necessarily the dialog or dialog element for which the context menu is being opened, due to "inheritance" of the container's context menu. For example, if a dialog with a context menu contains a push button without a **Context Menu**, right-clicking the push button will cause the CONTROL attribute to be set to the handle of the push button, even though the context menu is opened for the dialog.

Applies to	Context menu.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	Handle of right-clicked dialog or dialog element.

79 DEFAULT-BUTTON

If you assign the handle value of a push button control to the dialog's DEFAULT-BUTTON attribute, the push button control's click event handler is triggered when the end user presses ENTER in the dialog.

This is true unless another push button control has the focus, in which case pressing of ENTER triggers the focused button's click event handler.

A push button control defined as DEFAULT-BUTTON overrides a push button control defined as OK button (STYLE = 'O').

Alternatively to a push button also an ActiveX control can be assigned to the DEFAULT_BUTTON attribute, provided it behaves like a button. These ActiveX controls are marked in the system registry with the style OLEMISC_ACTSLIKEBUTTON.

Note: Default buttons have a black margin.

Applies to	Dialog (all types).
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE or any push button control handle

80 DESCENDING

Specifies the sort direction (ascending or descending) for a dialog element. The value of this attribute is only meaningful and effective if the dialog element's **SORTED** attribute is also set to TRUE.

In the case of list view controls, this attribute determines whether new items are inserted in ascending or descending sequence based on their labels.

For list view columns, this attribute returns TRUE if the column header is currently displaying a descending sort indicator, or FALSE otherwise. In this case, it does not necessarily imply that the column's data is currently in sorted sequence, because new items may have been added, or the values of existing items may have been changed, since the sort on the column was performed.

Applies to	List view control, list view column.
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	FALSE (ascending) / TRUE (descending).

81 DIL-TEXT

Determines the DIL text for a dialog element. The dynamic information line and the status line are updated with this text when this dialog element gets the focus. However, the text will only be displayed if **HAS-DIL** is TRUE. Then the DIL-TEXT will overlap the **STATUS-TEXT** value or vice versa, depending on which was modified last.

Applies to	ActiveX control, bitmap control, canvas control, column specification control, date/time picker control, edit area control, input field control, list box control, list view control, menu item, OLE container control, push button control, radio button control, scroll bar control, selection box control, signal, slider control, table control, toggle button control, tool bar item, tree view control.
Data Type	A253
Default Value	Empty string
Possible Values	Empty string / any text

82 DOCKING

Specifies the sides of the owning dialog on which docking is allowed. In order for a dockable tool bar control to be able to dock, both the tool bar control and the owning dialog must allow docking on the specified side(s). The supported docking positions are: none (tool bar control may only be floated), top only, bottom only, left only, right only, horizontal (top or bottom), vertical (left or right) or any side.

Applies to	Dialog (window, MDI Frame), dialog bar control, tool bar control.
Data Type	I4
Default Value	DL-ANY
Possible Values	DL-NONE / DL-LEFT / DL-TOP / DL-RIGHT / DL-BOTTOM / DL-VERT / DL-HORZ / DL-ANY

83 DPI

Specifies the number of dots (pixels) per logical inch in effect at the time the dialog was last saved from within the editor. This attribute, if non-zero, also indicates that the dialog is scalable.

Note: The value of this attribute varies depending on whether small or large fonts (or a custom font size setting) are being used.

Applies to	Dialog (all types), date/time picker control, list view control, progress bar control, slider control, spin control, tree view control.
Data Type	I4
Default Value	0
Possible Values	0 - 200

84 DRAG-MODE

Indicates which drag-drop operation types (if any) are supported by the dialog element when acting as the drag source.

The drag source is the control sourcing the data if Natural initiates the drag-drop operation automatically (e.g., for list box controls), or the control whose handle is passed to the **PERFORM-DRAG-DROP** action if the drag-drop operation is initiated manually.

Note that the symbolic drag mode constants listed below are defined in the local data area NGULKEY1, which is automatically used by all dialogs created within the dialog editor.

Applies to	ActiveX control, bitmap control, list box control, list view control, tree view control.
Data Type	I4
Default Value	DM-NONE (0)
Possible Values	DM-NONE (0) = No drag operation supported DM-COPY (1) = Copy operations supported DM-MOVE (2) = Move operations supported DM-COPYMOVE (3) = Copy and Move operations supported DM-LINK (4) = Link operations supported DM-COPYLINK (5) = Copy and Link operations supported DM-MOVELINK (6) = Move and Link operations supported DM-COPYMOVELINK (7) = Copy, Move and Link operations supported

85 DRAGGABLE

Determines whether a dialog element may be dragged by the user.

For bitmap controls with **DRAG-MODE** set to DM-NONE, this attribute determines whether the end user can use the mouse to drag a bitmap control onto another bitmap control in the same dialog. When dropping the bitmap control by releasing the mouse button, a drag and drop event will occur for the target bitmap control. Whenever a bitmap control is DRAGGABLE, you can also use the **INQ-DRAG-DROP** action provided with the PROCESS GUI statement to find out where the bitmap control was dropped.

For bitmap controls with **DRAG-MODE** set to a value other than DM-NONE, this attribute determines whether a drag and drop operation with the bitmap control acting as the drag source is allowed. For more information, please refer to the article Using the Clipboard and Drag and Drop .

For tool bar controls, this attribute determines whether the control is dockable and/or floatable in its own window. The sides of the parent dialog (if any) on which docking is allowed is determined by the **DOCKING** attribute values for both the tool bar control and the dialog.

Applies to	Bitmap control, tool bar control.
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	TRUE / FALSE

86 DROP-MODE

Indicates whether the dialog or dialog element can act as a drop target and, if so, which drag-drop operation types it supports.

The active drop target at any time during a drag-drop operation is the dialog or dialog element immediately under the drag cursor, if this is a drop target. Otherwise, it is the first ancestor control found (if any) when traversing up through the PARENT hierarchy that is a drop target (i.e., has a DROP-MODE attribute not equal to DM-NONE). The active drop target (if any) is the dialog or dialog element that receives the **DRAG-ENTER**, **DRAG-OVER** and **DRAG-DROP** events (if not suppressed).

The drag-drop operations available to the user at any time are determined by both the drag source's DRAG-MODE and the active drop target's DROP-MODE. Which of these available operations (if any) is used when a drop occurs and for determining which drag cursor is displayed to the user is determined by the augmentation keys held down whilst dragging. For example, the <ctrl> key forces a Copy operation (if available), whilst the <shift> and <ctrl> keys together force a Link operation (if available).

A Link operation implies that the source data is not moved or copied to the target. Instead, a reference to the source data is inserted at the target location. Thus, if the source data is changed, the link automatically refers to the changed data.

Note: The meaning of the word “reference” here is, in general, application-defined. However, in the case of the source data being a file or folder, it is conventionally a desktop shortcut.

Note that the symbolic drag mode constants listed below are defined in the local data area NGULKEY1, which is automatically used by all dialogs created within the dialog editor.

Applies to	ActiveX control, bitmap control, control box control, dialog (all types), edit area control, list box control, list view control, tree view control.
Data Type	I4
Default Value	DM-NONE (0)
Possible Values	DM-NONE (0) = Dialog or dialog element is not a drop target DM-COPY (1) = Copy operations supported DM-MOVE (2) = Move operations supported DM-COPYMOVE (3) = Copy and Move operations supported DM-LINK (4) = Link operations supported DM-COPYLINK (5) = Copy and Link operations supported DM-MOVELINK (6) = Move and Link operations supported DM-COPYMOVELINK (7) = Copy, Move and Link operations supported

87 EDIT-MASK

For input field controls and selection box controls, this attribute specifies the Natural **edit mask** that is used to validate end user input. You can, however, only use an edit mask if you also use the **Linked Variable** option in the Attribute Source dialog box, which can be accessed by clicking on the STRING's **Source** button in the dialog element's Attributes window. When the dialog element loses the focus (the end user leaves the input-field), the content of the field is checked against the edit mask. If the input is not compatible with the edit mask, Natural displays a message box asking the end user to **Retry** or **Cancel**. For more information, please refer to the article Validating Input in a Dialog Element.

For list view columns, this attribute specifies the Natural edit mask used to format the values for the items within the column for display. Note that these values are stored internally in the format determined by the column's **FORMAT** attribute. Note that the primary column in a list view displays the item labels, so an edit mask applied to this column also determines the display format for the labels (even if the list view control is not in report view mode). For more information, please refer to the article Working with List View Controls.

For tree view items, this attribute specifies the Natural edit mask used to format the item's label for display. Note that the values is stored internally in the format determined by the item's **FORMAT** attribute. For more information, please refer to the article Working with Tree View Controls.

For date and time picker controls, this attribute specifies a custom format string used to format the control's contents. Note, however, that the format string does not conform to the Natural edit mask syntax. Instead, the control uses its own format specifiers, as documented in the article Working with Date and Time Picker (DTP) Controls.

Applies to	Date/time picker control, input field control, list view column, selection box control, tree view item.
Data Type	A253
Default Value	Empty string
Possible Values	Empty string / any Natural edit mask

88 EMBEDDED-OBJECT

The value of this attribute is the name of a Natural embedded object, which has the default file extension ".neo". When a value is assigned to an OLE container control's EMBEDDED-OBJECT attribute, Natural loads the embedded object into the container, discarding any previously loaded object. If the values of the **SERVER-OBJECT** and **SERVER-PROGID** attributes are set for the OLE container control, the value of the EMBEDDED-OBJECT attribute is removed.

Applies to	OLE container control.
Data Type	A253
Default Value	None
Possible Values	All file names (if a file name is specified without an extension, the extension ".neo" is added by default)

89 ENABLED

Determines whether the end user may use the dialog element (for input, for example) or whether the dialog element is disabled (greyed, for example).

Applies to	ActiveX control, bitmap control, canvas control,context menu, control box control, date/time picker control, dialog (all types), dialog bar control, edit area control, group frame control, input field control, list box control, list view control, menu item, OLE container control, progress bar control, push button control, radio button control, scroll bar control, selection box control, signal, status bar control, status bar pane, slider control, spin control, tab control, table control, text constant control, toggle button control, tool bar control, tool bar item, tree view control.
Data Type	BOOLEAN
Default Value	TRUE
Possible Values	TRUE / FALSE

90 EVENT-QUEUEING

Determines whether messages received from the windowing system for this dialog should be processed immediately. By default, certain messages are queued in order to be compatible with the behaviour of older Natural versions under Windows 3.x. This attribute allows event queueing to be disabled at the dialog level, causing these messages to be processed immediately.

Event queueing was introduced under Windows 3.x to serialize messages dispatched via nested message loops. Such nested message loops were common under Windows 3.x in order to wait for a potentially lengthy operation (such as a remote database access) without hanging Windows. This is no longer a problem under newer versions of Windows, because these are pre-emptive operating systems. As such, it is no longer possible for an application that is not processing its messages to hang the system. Therefore, it is usually preferable (especially for newly written applications) to turn event queueing off.

Applies to	Dialog (all types).
Data Type	BOOLEAN
Default Value	TRUE
Possible Values	TRUE / FALSE

91 EXPANDED

Determines whether a tree view item is expanded (child items shown) or collapsed (child items hidden).

This attribute can be used to both obtain or modify the expansion status.

For more information, please refer to the article Working with Tree View Controls.

Applies to	Tree view item.
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	TRUE (expanded) / FALSE (collapsed).

92 FIRST-CHILD

The FIRST-CHILD attribute serves as a tool to query the first created child dialog element, that is, the dialog element one level below the (parent) dialog element. An example of such a child-parent relationship between dialog elements is the relationship between a dialog and several push button controls inside. With the FIRST-CHILD attribute, you query the push button control that has been created first inside the dialog. You use this attribute in conjunction with the **SUCCESSOR** attribute to "travel" through the hierarchy of dialog elements.

Applies to	Date/time picker control, dialogs (all types) and all dialog elements, image control, image list control, list view column, list view control, list view item, progress bar control, slider control, spin control, tree view control, tree view item.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE (no child existing) / any dialog element handle

93 FIRST-COLUMN-WIDTH

Determines the width in pixels of the first column in a table control. The attribute is used if the **HAS-FIRST-COLUMN** attribute is TRUE for the table control.

Applies to	Table control.
Data Type	I4
Default Value	50
Possible Values	0 - 9999

94 FIRST-VISIBLE-COLUMN

Determines the index of the first visible column.

- If no frozen columns are defined, the first visible column is the leftmost column of the table.
- If frozen columns *are* defined, the first visible column is the first column displayed to the right of the frozen columns.

Applies to	Table control.
Data Type	I4
Default Value	1
Possible Values	0 - 9999

95 FIRST-VISIBLE-ITEM

Determines the first visible item in a dialog element.

The first visible item is the item displayed at the top of the dialog element. If the value is being set, the handle specified must belong to an item within the dialog element. In this case, the specified item will only become the first visible item if the scroll amount required to achieve this is within the bounds of the dialog element's scroll range (if any). For list view controls, the attribute may only be queried

Applies to	List box control, list view control, tree view control.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE / list box item handle

96 FIRST-VISIBLE-ROW

Determines the index of the row which is displayed at the top of the table.

Applies to	Table control.
Data Type	I4
Default Value	1
Possible Values	0 - 9999

97 FOLLOWS

Specifies the preceding dialog element in the (new) control sequence. The control sequence determines the order in which dialog elements receive the focus on repeated pressing of the TAB key (or arrow keys, for controls with the same group ID). It also determines which dialog element gets the focus on input of a mnemonic corresponding to a dialog element which cannot receive the focus itself (for example, a text-constant or group frame). By default, the FOLLOWS attribute is based on the SUCCESSOR chain. You can use this attribute to move a dialog element from its default position. Assigning a value of NULL-HANDLE moves the dialog element to the end of the control sequence, whereas assigning the handle of the parent dialog itself moves the dialog to the front of the control sequence. Assigning the handle of the control itself has the effect of removing the control from the control sequence such that it can no longer be reached via the TAB key.

When applied to dialogs, the FOLLOWS attribute returns the last dialog element in the control sequence. This enables complete enumeration of the control sequence in reverse order.

Applies to	ActiveX control, bitmap control, canvas control, control box control, date/time picker control, dialog (all types), dialog bar control, edit area control, group frame control, input field control, list box control, list view control, OLE container control, push button control, progress bar control, radio button control, scroll bar control, selection box control, slider control, spin control, status bar control, tab control, table control, text constant control, toggle button control, tool bar control, tree view control.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE / handle value of the desired predecessor in the control sequence.

98 FONT-HANDLE

Serves as a handle to assign a certain combination of font face, font size, and font style to a dialog element. Assigning an existing font handle value to this attribute leads to the STRING of the dialog element being displayed in this font. You can use the same FONT-HANDLE for several dialog elements.

Applies to	Dialog (all types), date/time picker control, edit area control, graphic text control, group frame control, input field control, list box control, list view control, push button control, radio button control, selection box control, status bar control, tab control, table control, text constant control, toggle button control, tree view control.
Data Type	HANDLE
Default Value	System font
Possible Values	System font / any other defined fonts

99 FONT-STRING

Sets the default font throughout a dialog. The default font must be set as a parameter while the dialog is being created with PROCESS GUI action ADD. If a dialog element inside the dialog contains text with no particular font assigned to it, this text will be displayed in the font specified by FONT-STRING. When you assign a FONT-STRING to a dialog, a font control is created internally, with the dialog being the parent of the font control. The font's handle can be queried from the dialog's **FONT-HANDLE** attribute.

Note: If the FONT-STRING attribute has created fonts that are too large or too small for the dialog window layout, you can use this attribute to resize all fonts evenly throughout the dialog.

Applies to	Dialog (all types).
Data Type	A253
Default Value	System font
Possible Values	Encoded string which can be edited in the dialog editor's dialog attributes window. Font strings consist of several fields separated by a delimiter character.

Possible Delimiter Characters

The default delimiter character is the slash "/".

Possible delimiter characters are:

"/", ",", "#", ":", ";", "|", "@", "$", "%", "&".

Fields

1. The font's face name.

2. The font style: Empty, or equivalently, the word "Regular", or a combination of the keywords "Bold", "Italic", "Strikeout", and "Underlined", separated by blanks.

3. The font size: Either the size in points (here, an optional decimal point followed by one digit is allowed), or the size as *width* x *height*. A positive value indicates character size, a negative value indicates cell size. A zero indicates that the system should choose a value for the field. The *width* x *height* notation allows for better portability across screen resolutions.

4. Character set information. A combination of keywords, one from each of the following subgroups, separated by blanks:

Character sets:	ANSI or SYMBOL or SHIFTJIS or OEM.
Character sets:	ANSI or SYMBOL or SHIFTJIS or OEM.
Fixed or variable-width font:	FIXED or VARIABLE.
Font family:	ROMAN or SCRIPT or SWISS or MODERN or DECORATIVE.
Print quality:	DRAFT or PROOF.

5. A character indicating the platform. For fonts selected under windows, this character is “W”.

6. Platform-dependent field. For Windows, this is the clipping precision.

7. Platform-dependent field. For Windows, this is the output precision.

Sample String

```
/Arial/Bold/0 x -19/ ANSI VARIABLE SWISSDRAFT/W/2/0/
```

100 FOREGROUND-COLOUR-NAME

Provides a choice of existing foreground colors. Foreground colors also refer to text.

When setting foreground colors, you have four possibilities:

- Use the FOREGROUND-COLOUR-NAME attribute and leave the value at DEFAULT (0). The color will then be determined by your color settings in the operating system.
- Use the FOREGROUND-COLOUR-NAME attribute and choose from one of the dialog editor's predefined colors. These are represented by the values 1 to 18.
- Define your own color by using the **FOREGROUND-COLOUR-VALUE** attribute in a dialog editor attributes window. Here, you must first set the FOREGROUND-COLOUR-NAME attribute to CUSTOM (50). A dialog box appears where you can set values for the red, green, and blue elements of your color (RGB model).
- Set the color dynamically in the event handler code by assigning a value to the attribute.

Note: Due to consistency constraints imposed by Windows, the specified setting may not be effective for particular dialog element types, depending on the version of Windows in use and whether visual themes are enabled.

Applies to	Canvas control, edit area control, graphic text control, group frame control, input field control, line control, list view control, progress bar control, push button control, radio button control, rectangle control, scrollbar control, selection box control, table control, text constant control, toggle button control, tree view control.
Data Type	I4
Default Value	DEFAULT (0)

Possible Values

DEFAULT (0)	WHITE (1)	BLACK (2)
LTGRAY (3)	GRAY (4)	DKGRAY (5)
RED (6)	GREEN (7)	BLUE (8)
CYAN (9)	MAGENTA (10)	BROWN (11)
YELLOW (12)	LIGHTRED (13)	LIGHTGREEN (14)
LIGHTBLUE (15)	LIGHTCYAN (16)	LIGHTMAGENTA (17)
BRIGHTWHITE (18)	CUSTOM (50)	

Note: The text representation can be substituted for the integer representation because the local data area NGULKEY1 (supplied in library SYSTEM) is automatically included in the event handler code.

101 FOREGROUND-COLOUR-VALUE

Provides a facility to define customized foreground colors. Foreground colors also refer to text.

When setting foreground colors, you have four possibilities:

- Define your own color by using the FOREGROUND-COLOUR-VALUE attribute in a dialog editor Attributes window. Here, you must first set the **FOREGROUND-COLOUR-NAME** attribute to CUSTOM (50). A dialog box appears where you can set values for the red, green, and blue elements of your color (RGB model).
- Use the **FOREGROUND-COLOUR-NAME** attribute and leave the value at DEFAULT (0). Your color will then be determined by your color settings in the operating system.
- Use the **FOREGROUND-COLOUR-NAME** attribute and choose from one of the dialog editor's predefined colors. These are represented by the values 1 to 18.
- Set the color dynamically in event handler code by assigning a set of three RGB values to the attribute. To do so, you must first set the **FOREGROUND-COLOUR-NAME** attribute to CUSTOM (50). You enter the three RGB values in hexadecimal form, such as H'FF0000'.

Note: Due to consistency constraints imposed by Windows, the specified setting may not be effective for particular dialog element types, depending on the version of Windows in use and whether visual themes are enabled.

Applies to	Canvas control, edit area control, graphic text control, group frame control, input field control, line control, list view control, progress bar control, push button control, radio button control, rectangle control, scrollbar control, selection box control, table control, text constant control, toggle button control, tree view control.
Data Type	B3
Default Value	None
Possible Values	Any red, green, and blue value from 1 to 253

102 FORMAT

Determines the format in which data associated with a dialog element is to be stored internally.

The selection of the appropriate format ensures that sorting works as intended. For example, if a list view column may contain numeric data, the choice of one of the numeric FORMAT values ensures that the value "10" is collated after the value "1", for example. In addition, specification of a format allows an **EDIT-MASK** to be used, if required, since edit masks in Natural contain type-specific special characters. Lastly, choosing the correct format can help save space and can change the default display representation in the case where an edit mask is not being used.

If the format is set to FT-ALPHA, only the actual number of characters used (up to a maximum of 253) are stored internally.

For more information, please refer to the articles Working with List View Controls and Working with Tree View Controls.

Applies to	List view column, tree view item.
Data Type	I4
Default Value	FT-ALPHA (0)
Possible Values	FT-ALPHA (0) = format An (where n = actual length) FT-BINARY (1) = B8 FT-SHORTBIN (2) = B4 FT-LONGBIN (3) = B16 FT-DATE (4) = D FT-FLOAT (5) = F8 FT-SHORTFLOAT (6) = F4 FT-INTEGER (7) = I4 FT-SHORTINT (8) = I2 FT-LOGICAL (9) = L FT-NUMERIC (10) = P15 FT-LONGNUM (11) = P31 FT-DECIMAL (12) = P10.5

	FT-SHORTDEC (13) = P4.3 FT-LONGDEC (14) = P21.7 FT-CURRENCY (15) = P13.2 FT-FRACTION (16) = P0.7 FT-TIME (17) = T

103 FROZEN-COLUMNS

Determines the number of columns on the left side of the table control that cannot be scrolled horizontally.

Applies to	Table control.
Data Type	I4
Default Value	0
Possible Values	0 - 9999

104 GROUP-ID

Assigns a group ID to a radio button control. Radio button controls with the same GROUP-ID value are regarded as a group; that is, only one radio button control out of this group may be CHECKED at any time.

Applies to	Radio button control.
Data Type	I4
Default Value	0
Possible Values	0 or any other integer value

105 HANDLE-VARIABLE

This attribute is only used in the context of the PROCESS GUI statement action ADD. It identifies the Natural variable which receives the handle of the newly created dialog element.

Note: The value of this attribute is set at creation time of the corresponding dialog element; it cannot be queried or modified.

Data Type	HANDLE
Data Type	HANDLE
Possible Values	Any Natural variable

Example:

```
DEFINE DATA LOCAL 1
#NEW2 HANDLE OF INPUTFIELD END-DEFINE...
PROCESS GUI ACTION ADD WITH PARAMETERS HANDLE-VARIABLE
 = #NEW2 TYPE = INPUTFIELD  STRING
 = 'NEW2' RECTANGLE-X
 = 24 RECTANGLE-Y
 = 30 RECTANGLE-W
 = 176 RECTANGLE-H
 = 28 ENABLED
 = TRUE VISIBLE
 = TRUE PARENT
 = #DLG$WINDOW
 END-PARAMETERS
```

106 HAS-DIL

Determines whether a status bar control or the status line of a dialog is updated with DIL-TEXTs. When applied to a dialog, the following interdependencies exist for the display of DIL-TEXTs and STATUS-TEXTs:

HAS-DIL	HAS-STATUS-BAR	DIL-TEXT	STATUS-TEXT
TRUE	TRUE	displayed	displayed
TRUE	FALSE	-	-
FALSE	TRUE	-	displayed
FALSE	FALSE	-	-

Applies to	Dialog (window, MDI frame, MDI child), status bar control.
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	TRUE / FALSE

107 HAS-FIRST-COLUMN

Determines whether the table control displays a first column to the left of the other columns. This first column consists of a button series to the left of each row. Each button is inscribed with the number of the row.

Applies to	Table control.
Data Type	BOOLEAN
Default Value	TRUE
Possible Values	TRUE / FALSE

108 HAS-HELP-BUTTON

Determines whether the dialog title bar has a help button or not. Selection of this button invokes help mode. You can then display help for a control simply by clicking on it with the left mouse button.

Note: Even if this attribute is set to TRUE, Windows will not display the help button if the maximize and minimize buttons are also present.

Applies to	Dialog (window, MDI frame, MDI child).
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	TRUE / FALSE

109 HAS-MENU-BAR

Indicates whether the dialog has a menu bar.

Applies to	Dialog (window, MDI frame, MDI child).
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	TRUE / FALSE

110 HAS-STATUS-BAR

Determines whether a status bar is displayed at the bottom of the dialog window. This status bar displays the value of the **STATUS-TEXT** attribute or, if applicable, of the **DIL-TEXT** attribute of the dialog element that currently has the focus. For the display of DIL-TEXTs and STATUS-TEXTs, the following interdependencies exist:

HAS-DIL	HAS-STATUS-BAR	DIL-TEXT	STATUS-TEXT
TRUE	TRUE	displayed	displayed
TRUE	FALSE	-	-
FALSE	TRUE	-	displayed
FALSE	FALSE	-	-

Applies to	Dialog (window, MDI frame, MDI child).
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	TRUE / FALSE

111 HAS-SYSTEM-BUTTON

Determines whether the dialog has a system button or not. The system button is in the top left corner of the window and provides a menu with the platform's standard window functions (like Close or Minimize).

Applies to	Dialog (window, MDI frame, MDI child).
Data Type	BOOLEAN
Default Value	TRUE
Possible Values	TRUE / FALSE

112 HAS-TOOLBAR

Determines whether the tool bar of a dialog (that is, the tool bar plus the tool bar items) is to be displayed or not. You use this attribute to dynamically switch the tool bar on and off.

Note: The handle value of the tool bar must be assigned to the **TOOLBAR-HANDLE** attribute.

Applies to	Dialog (window, MDI frame, MDI child).
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	TRUE / FALSE

113 HAS-TOOLTIP

Indicates whether tool tips are to be displayed for a tool bar or status bar control. If this attribute is set to FALSE, tool tip display is suppressed.

Applies to	List view control, status-bar control, tab control, tool bar control, tree view control.
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	TRUE / FALSE

114 HEADER-FONT-HANDLE

Serves as a handle to assign a certain combination of font face, font size, and font style to the first column and the header line of a table control. Assigning an existing font handle value to this attribute leads to the header being displayed in this font.

Applies to	Table control.
Data Type	HANDLE
Default Value	System font
Possible Values	System font / any other defined fonts

115 HEADER-HEIGHT

Determines the height (in pixels) of the header line in a table control. To add a header line, the table control's **STYLE** attribute must include the letter "h".

Applies to	Table control.
Data Type	I4
Default Value	0
Possible Values	0 - 9999

116 HELP-FILENAME

Determines a dialog's help file name usually without extension. It is assumed that the extension is ".chm" or ".hlp". If the end user presses F1 in an active dialog, Natural first queries for a file with the value of HELP-FILENAME plus the extension ".chm". If it does not find such a file, it queries for a file with the value of HELP-FILENAME plus the extension ".hlp". If HELP-FILENAME is not specified for the current dialog, the HELP-FILENAME of the parent dialog (if any) is used, if specified. This process is repeated as necessary until a top-level dialog is reached. If no non-empty HELP-FILENAME attribute is found for any of these dialogs, Natural searches for the file *libraryname*.chm first, then (if not found) *libraryname*.hlp, where *libraryname* is the name of the logon library.

The HELP-FILENAME attribute can also contain path information. If this is the case, then only that path is searched. If no path information is supplied, the help file is searched for in the logon library's RES subdirectory first, then in the RES subdirectory of each steplib, then in the directory assigned to the environment variable NATGUI_BMP.

Applies to	Dialog (all types).
Data Type	A253
Default Value	None
Possible Values	Any file name

117 HELP-ID

The help ID to be passed to the help system whenever the end user requests help for the dialog or dialog element that has the focus by pressing F1. In most help systems, this help ID is referred to as topic number.

If the dialog or dialog element with the focus does not provide a HELP-ID, the HELP-ID of its parent is taken.

Applies to	ActiveX control, bitmap control, canvas control, column specification control, context menu, date/time picker control, dialog (all types), dialog bar control, edit area control, input field control, list box control, list view control, menu bar, OLE container control, push button control, radio button control, scroll bar control, selection box control, slider control, spin control, submenu control, tab control, table control, toggle button control, tree view control.
Data Type	I4
Default Value	0
Possible Values	0 or any other integer value

118 HORIZ-SCROLLABLE

Determines whether the dialog or dialog element has a horizontal scroll bar.

Note: This attribute can be dynamically enabled, but not dynamically disabled.

Applies to	Dialog (all types), edit area control, table control.
Data Type	BOOLEAN
Default Value	TRUE
Possible Values	TRUE / FALSE

119 ICONIZED

Determines whether the OLE container control represents data visually inside the control's borders or whether data are displayed as the server application's icon (useful for sound or video).

Applies to	OLE container control.
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	TRUE / FALSE

120 IMAGE

Specifies the **Image Control** that contains the base image (if any) used by an item. If this attribute is not set to NULL-HANDLE, the image handle specified must refer to an image from the parent control's image list, as specified by its **IMAGE-LIST** attribute value. An image control referenced by this attribute cannot be an overlay image (i.e., cannot have the "O" **STYLE**).

If the referenced image control contains more than one image, the image used is determined by the item's **IMAGE-INDEX** attribute value.

For more information, please refer to the article Working with Image List Controls.

Applies to	List view item, tree view item.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE / handle value of image.

121 IMAGE-INDEX

Specifies an item's image index. This index determines the base image to be used from the **IMAGE** control associated with the item, or from its parent control's **IMAGE-LIST** if the former attribute is set to NULL-HANDLE.

The first case above, the image index is the zero-based relative index of the image within the specified image control, with wrap-around if the index is out of range. For example, if the referenced image control contains three images, the image indexes of 0, 3, 6, 9, and so on, will all select the first image.

In the second case above, the image index is the one-based absolute position of the required image in the parent control's image list, with no wrap-around for out-of-range values. For example, if the parent control's image list contains 10 images, an image index of 1 will select the first image from the first image control belonging to the image list. Similarly, an image index of 10 will select the last image from the last image control belonging to the image list. Out-of-range image index values, such as 11 in this case, are simply ignored.

Note: With this absolute indexing mechanism, Natural does not automatically track changes made to to the image list. For example, if image 9 is removed from a 10-image image list, causing the 10th image to suddenly become the 9th image, items with an image index of 10 are neither automatically updated to 9, nor automatically refreshed to reflect the change.

For more information, please refer to the article Working with Image List Controls.

Applies to	Image control, list view item, tree view item.
Data Type	I4
Default Value	0
Possible Values	Any positive value.

122 IMAGE-LIST

Specifies the image list containing the images used by the dialog element. A value of NULL-HANDLE indicates that the dialog element has no associated image list (and hence does not display any images). You can use the same image list for several dialog elements.

For more information, please refer to the article Working with Image List Controls.

Applies to	List view control, tree view control.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE / handle value of image list.

123 INPLACE-ACTIVE

Determines whether the server of the OLE container control is in-place active or not.

Applies to	OLE container control.
Data Type	BOOLEAN
Default Value	TRUE
Possible Values	TRUE / FALSE

124 ITEM

Used in particular event handlers for list view and tree view controls in order to determine the item that caused the event to be raised.

The following table indicates the events for which this attribute is set:

Event	Interpretation
Activate Event	Item being activated, or NULL-HANDLE in the case of **MULTI-SELECTION** controls.
After-Edit Event	Item whose label has just been edited.
Before-Edit Event	Item whose label the user is attempting to edit.
Check Event	Item that is being checked or unchecked.
Click Event	Item that is being selected or deselected.
Collapse Event	Tree view item that is being collapsed.
Expand Event	Tree view item that is being expanded.

For more information, please refer to the articles Working with List View Controls and Working with Tree View Controls.

Applies to	List view control, tree view control.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE / handle of list view item or tree view item.

125 ITEM-H

Determines the height of each item for the dialog element. For status bar controls, this is the minimum pane height.

For image list controls, this attribute specifies the height of the images in the image file(s) providing the image list's images. If neither the “Large images (L)” nor the “Small images (S)” **STYLE** flags are set for the image list, this is also the height of the images stored in the image list control. If either either (or both) of these styles are specified, the ITEM-H attribute is usually left set to its default value of zero, implying the use of the the system-defined small and/or large icon height(s). For more information, please refer to the article Working with Image List Controls.

Applies to	Image list control, status bar control, tab control, tool bar, tool bar control, tree view control.
Data Type	I4
Default Value	0 (system default)
Possible Values	0 - 9999

126 ITEM-W

Determines the width of each item for the dialog element.

For image list controls, this attribute specifies the width of the images in the image file(s) providing the image list's images. If neither the “Large images (L)” nor the “Small images (S)” **STYLE** flags are set for the image list, this is also the width of the images stored in the image list control. If either either (or both) of these styles are specified, the ITEM-W attribute is usually left set to its default value of zero, implying the use of the the system-defined small and/or large icon width(s). For more information, please refer to the article Working with Image List Controls.

Applies to	Image list control, tab control, tool bar, tool bar control.
Data Type	I4
Default Value	0 (system default)
Possible Values	0 - 9999

127 LAST-CHILD

Contains the handle of the dialog element's last child dialog element, or NULL-HANDLE if the dialog element has no children. You can use this attribute in conjunction with the **PREDECESSOR** and **PARENT** attributes to traverse the parent/child hierarchy in the reverse direction.

Applies to	Date/time picker control, dialogs (all types) and all dialog elements, image control, image list control, list view column, list view control, list view item, progress bar control, slider control, spin control, tree view control, tree view item.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE (no child existing) / any dialog element handle.

128 LENGTH

Specifies the maximum number of characters that can be entered by the end user for the dialog element. It does not limit the number of characters settable programmatically.

For list view items and tree view items, this attribute specifies the maximum enterable label length, in characters.

Applies to	Column specification control, edit area control, input field control, list view item, selection box control, tree view item.
Data Type	I4
Default Value	0 (= unlimited) 253 for input field controls and selection box controls. Maximum system limits for edit area controls.
Possible Values	0 - 30000 for edit area controls, or 0 - 253 for input field controls and selection box controls (including column specification controls based on these two types), list view items and tree view items

129 LINE

For scroll bar controls, this attribute determines the number of logical units by which the control's thumb (slider) moves in response to either pressing the corresponding arrow keys on the keyboard or by clicking the arrow buttons at either end of the scroll bar itself.

For slider controls, this attribute determines the number of logical units by which the control's thumb moves in response to pressing the corresponding arrow keys on the keyboard.

Inside a numeric scale limited by the values of the control's **MIN** and **MAX** attributes, the value of this attribute determines the size of the steps by which the thumb will be moved. If you set a small value in relation to the **MIN** and **MAX** attributes, the thumb will move continuously. If you set a larger value, the thumb will move in visible steps.

Applies to	Scrollbar control, slider control.
Data Type	I4
Default Value	0
Possible Values	0 or any integer value

130 LINKED

Determines whether the variable specified for the dialog element is linked or not. Setting this attribute to TRUE and then assigning a Natural variable to the **VARIABLE** attribute has the same effect as specifying a “Linked Variable” in a source dialog box of a “String” attribute window.

Applies to	Input field control, selection box control.
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	TRUE / FALSE

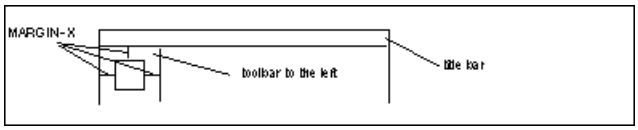
131 LOCATION

Specifies the position (top, bottom, left, right) of a status bar or tool bar control relative to the associated dialog. For dockable tool bars floated in a separate window, the location is represented by the value TB-FLOAT. Status-bar controls may only be positioned at the top or bottom of a dialog.

Applies to	Dialog bar control, status-bar control, tool bar control.
Data Type	I4
Default Value	TB-TOP
Possible Values	TB-TOP / TB-BOTTOM / TB-LEFT / TB-RIGHT / TB-FLOAT

132 MARGIN-X

For tool bars, this attribute determines the margin to the left, to the right and above the tool bar items displayed in the tool bar area. This attribute only applies if **TOOLBAR-POS** is set to TB-LEFT or TB-RIGHT.



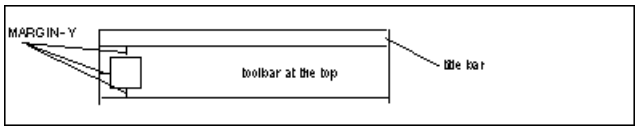
For tool bar controls, this attribute determines the margin on the left and right sides of the control, if the control's **LOCATION** is set to TB-TOP, TB-BOTTOM or TB-FLOAT. Otherwise, the attribute determines the margin at the top and bottom of the control.

For the remaining dialog element types, this attribute always determines its right and left margins.

Applies to	Dialog bar control, status bar control, tab control, tool bar, tool bar control.
Data Type	I4
Default Value	0
Possible Values	0 - 9999

133 MARGIN-Y

For tool bars, this attribute determines the margin to the left of, above and below the tool bar items displayed in the tool bar area. This attribute only applies if **TOOLBAR-POS** is set to TB-TOP or TB-BOTTOM.



For tool bar controls, this attribute determines the margin at the top and bottom of the control, if the control's **LOCATION** is set to TB-TOP, TB-BOTTOM or TB-FLOAT. Otherwise, the attribute determines the margin on the left and right sides of the control.

For the remaining dialog element types, this attribute always determines its top and bottom margins.

Applies to	Dialog bar control, status bar control, tab control, tool bar, tool bar control.
Data Type	I4
Default Value	0
Possible Values	0 - 9999

134 MAX

Determines the maximum limit for a dialog element's (numeric) range. Because both **MIN** and MAX attributes default to zero, it is necessary to explicitly specify a value for one or both of these attributes in order to ensure that the dialog element has a non-zero range.

Note: The difference between the values of the MAX and **MIN** attributes must be less than 32767.

Applies to	Progress bar control, scrollbar control, slider control, spin control.
Data Type	I4
Default Value	0
Possible Values	0 or any integer value between -32767 and 32767; must be larger than the value of the MIN attribute.

135 MAXIMIZABLE

Determines whether the dialog has a maximize button in the top right corner. This button enables you to maximize the dialog so that it covers the whole screen.

Applies to	Dialog (window, MDI frame, MDI child), dialog bar control.
Data Type	BOOLEAN
Default Value	TRUE
Possible Values	TRUE / FALSE

136 MAXIMIZED

If this attribute is set, the dialog is maximized to full screen size and the dialog's size event occurs. If you have specified a size event handler code section, this code section will be triggered. An MDI child window will be maximized to fill the MDI frame window's client area.

Applies to	Dialog (all types), dialog bar control.
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	TRUE / FALSE

137 MENU-HANDLE

When used with a dialog, it creates a link between a menu bar plus its submenu controls, its items, and this dialog. The menu is then displayed at the top of the dialog. When the MENU-HANDLE attribute is modified to NULL-HANDLE, the menu disappears without being destroyed. When used with a menu item, it creates a link between a submenu control and this menu item. The menu structure is then nested by one more level. Several dialogs may be linked to the same menu bar, but you may only link one menu bar to a dialog.

Applies to	Dialog (window, MDI frame, MDI child), menu item.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE / any handle of a menu bar

138 MENU-ITEM-OLE

Determines whether a top-level menu item or a submenu control represents the OLE Container or File or Window group or not. If a dialog has a menu bar and an OLE container control is being edited in-place, the submenus of the OLE server application are merged into the Natural menu bar. The Natural contribution to the merged menu consists of the submenu controls called File, Container, and Window, depending on what you set with this attribute. The server application's contribution are the submenu controls called Edit, Object, and Help.

Applies to	Menu item, submenu control.
Data Type	I4
Default Value	MO-NORMAL

Possible Values

MO-NORMAL	No OLE menu item.
MO-NORMAL	No OLE menu item.
MO-CONTAINER	OLE Container group.
MO-FILE	OLE File group.
MO-WINDOW	OLE Window group.

139 MENU-ITEM-TYPE

Determines the function of a menu item.

The values MT-CUT, MT-COPY, MT-PASTE, MT-DELETE and MT-UNDO are predefined functions that are executed by the windowing system. They are automatically enabled or disabled depending on the focus and on the selection.

Applies to	Menu item, signal, tool bar item.
Data Type	I4
Default Value	MT-NORMAL

Possible Values

MT-NORMAL	This is a menu item on the lowest level of the menu structure. Only items of this type can have an event handler.
MT-NORMAL	This is a menu item on the lowest level of the menu structure. Only items of this type can have an event handler.
MT-SEPARATOR	Items of this type are displayed as horizontal line in a submenu to separate menu items optically.
MT-EDITCUT	Items of this type allow the end user to cut a selected portion of text and copy it to the clipboard. (The cut text is in the dialog element that currently has the focus).
MT-EDITCOPY	Items of this type allow the end user to copy a selected portionof text to the clipboard. (The copied text is in the dialog element that currently has the focus).
MT-EDITPASTE	Items of this type allow the end user to paste the portion of text that is in the clipboard. (The text is pasted into the dialog element that currently has the focus).
MT-EDITDELETE	Items of this type allow the end user to delete a selected portion of text. (The deleted text is in the dialog element that currently has the focus).
MT-EDITUNDO	Items of this type allow the end user to undo a text input operation. The previous text input operation is undone in the dialog element that currently has the focus.
MT-SUBMENU	Items of this type represent a submenu. When selected, a (*) submenu drops down.

MT-WINDOWMENU	Items of this type also represent a submenu, only that all MDI (*) children are entered automatically into it.
MT-MDITILE	Arranges the MDI children of an MDI frame dialog in a tile (**) manner.
MT-MDICASCADE	Arranges the MDI children of an MDI frame dialog in a cascade (**) manner.
MT-MDIARRANGE (**)	Arranges the iconized MDI children of an MDI frame dialog.
MT-OBJECTVERBS	Displays and enables the verbs (actions) available for an OLE container control in the current dialog if the OLE container control has the focus and is assigned to an OLE server application. If the focus is on another dialog element, the disabled entry “Object” appears.

* If you create a menu item of this type, you must assign the handle value of the submenu control to the MENU-HANDLE attribute.

** Menu items of this type must be contained in a menu item of type MT-WINDOWMENU.

Note: The local data area NGULKEY1in library SYSTEM lists reserved keywords to be used in any event handler code. This enables you to refer to the above attribute values by the more meaningful keywords rather than by the IDs. It also enables you to use meaningful dialog element names as parameters in a CALLNAT to an NGU-prefixed subprogram or in an OPEN DIALOG to an NGU-prefixed Dialog or in a PROCESS GUI statement action.

140 MIN

Determines the minimum limit for a dialog element's (numeric) range. Because both the MIN and **MAX** attributes default to zero, it is necessary to explicitly specify a value for one or both of these attributes in order to ensure that the dialog element has a non-zero range.

Note: The difference between the values of the **MAX** and MIN attributes must be less than 32767.

Applies to	Progress bar control, scrollbar control, slider control, spin control.
Data Type	I4
Default Value	0
Possible Values	0 or any integer value between -32767 and 32767; must be smaller than the value of the MAX attribute.

141 MINIMIZABLE

Determines whether the dialog has a minimize button in the top right corner. This button enables you to minimize the dialog to an icon.

Applies to	Dialog (window, MDI frame, MDI child).
Data Type	BOOLEAN
Default Value	TRUE
Possible Values	TRUE / FALSE

142 MINIMIZED

If this attribute is set, the dialog is minimized and appears as an icon and the dialog's size event occurs. If you have specified a size event handler code section, this code section will be triggered.

Applies to	Dialog (all types), dialog bar control.
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	TRUE / FALSE

143 MODIFIED-SUCCESSOR

Gets all modified dialog elements of a dialog one after the other. To get the first modified dialog element in a dialog, you use the handle of the dialog to query the attribute value. To get the next modified dialog element, you use the handle of the dialog element queried last. If no more modified dialog elements are found, a NULL-HANDLE is returned and the query for a modified successor is ended. If you query the MODIFIED-SUCCESSOR of a dialog (window, MDI frame, MDI child), the handle value of the dialog's first modified child will be returned.

To get the selected items of a list box control, you use the **SELECTED-SUCCESSOR** attribute.

Applies to	Date/time picker control, dialog (all types), edit area control, input field control, list box control, radio button control, scroll bar control, selection box control, slider control, spin control, table control, toggle button control.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE / any dialog element handle

144 MODIFIABLE

Determines whether a dialog element can be edited or not.

Note that, although disabling the control also prevents any modifications being made, the use of the **ENABLED** attribute (if available) is usually more severe in its effect than the use of this attribute. For example, disabled controls usually have a more radical visual appearance than non-modifiable (enabled) ones do, being typically completely grayed-out. Furthermore, disabled controls cannot receive the focus, so it is not possible to scroll text in a disabled control, or to make a selection and copy it to the clipboard.

If this attribute is set to FALSE for an OLE container control that has been activated for in-place editing, the OLE object cannot be modified.

For tab controls, this attribute has a different meaning than provided above, and determines whether the user is allowed to switch between tabs. For more information, please refer to the article Working with Tab Controls.

For list view items and tree view items, this attribute refers to the ability to change the item's text label. However, if the MODIFIABLE attribute is set to FALSE for the list view control or tree view control itself is set to FALSE, no editing of any items is allowed, regardless of the setting of the MODIFIABLE attribute for the individual items. For more information, please refer to the article Label Editing in Tree View and List View Controls.

Applies to	Column specification control, edit area control, input field control, list view control, list view item, OLE container control, selection box control, tab control, table control, tree view control, tree view item.
Data Type	BOOLEAN
Default Value	TRUE
Possible Values	TRUE / FALSE

145 MODIFIED

If set to TRUE, the content of a dialog element (e.g., **STRING** attribute for an input field control, **CHECKED** attribute for a toggle button control, etc.) has been modified since the last resetting of the MODIFIED attribute. It may be modified either directly by user interaction or indirectly by an event handler: either the open event handler resets it indirectly on opening the dialog or the user resets it directly in the attributes window of the dialog that contains the dialog element in question.

Applies to	Date/time picker control, dialog (all types), edit area control, input field control, list box control, radio button control, scroll bar control, selection box control, slider control, spin control, table control, toggle button control.
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	TRUE / FALSE

146 MULTI-SELECTION

Determines whether a dialog element allows multiple items to be selected or not. If it does, the selected items may be retrieved by using the **SELECTED-SUCCESSOR** attribute.

Applies to	List box control, list view control.
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	TRUE / FALSE

147 NAME

Specifies the dialog's object name, without the ".NS3" file extension.

Applies to	Dialog (all types).
Data Type	A8
Default Value	None
Possible Values	Name of dialog object.

148 OBJECT-SIZE

Determines the size of the object in the OLE container control.

Applies to	OLE container control.
Data Type	I4
Default Value	0
Possible Values	0 or any other integer value.

149 OFFSET-X

For canvas controls, this attribute determines the horizontal offset of the coordinate origin relative to the origin of the control's client area (the region of the control excluding its border, if any).

For example, changing this attribute to 100 from its default value of 0 causes all objects displayed within the canvas to be shifted 100 pixels (=coordinate units) to the right. This may mean that some objects disappear off the right hand side of the canvas, whilst other objects that were previously invisible may appear on the left.

For list view controls in one of the icon view modes, this attribute may (only) be queried and returns the horizontal offset of the origin of the control's client area relative to the (view) coordinate origin (i.e., the x-coordinate of the client area origin). Note that this is the inverse of the definition used in the case of canvas controls. Normally, the list view origin is at view coordinate (0, 0). However, if the list view is scrolled, this may no longer be the case. For more information, please refer to the article Working with List View Controls.

Applies to	Canvas control, list view control.
Data Type	I4
Default Value	0
Possible Values	0 - 9999

150 OFFSET-Y

For canvas controls, this attribute determines the vertical offset of the coordinate origin relative to the origin of the control's client area (the region of the control excluding its border, if any).

For example, changing this attribute to 100 from its default value of 0 causes all objects displayed within the canvas to be shifted 100 pixels (=coordinate units) downwards. This may mean that some objects disappear off the bottom side of the canvas, whilst other objects that were previously invisible may appear at the top.

For list view controls in one of the icon view modes, this attribute may (only) be queried and returns the vertical offset of the origin of the control's client area relative to the (view) coordinate origin (i.e., the y-coordinate of the client area origin). Note that this is the inverse of the definition used in the case of canvas controls. Normally, the list view origin is at view coordinate (0, 0). However, if the list view is scrolled, this may no longer be the case. For more information, please refer to the article Working with List View Controls.

Applies to	Canvas control, list view control.
Data Type	I4
Default Value	0
Possible Values	0 - 9999

151 OVERLAY

Specifies the **Image Control** that contains the overlay image (if any) used by an item. If this attribute is not set to NULL-HANDLE, the image handle specified must refer to an image from the parent control's image list, as specified by its **IMAGE-LIST** attribute value. An image control referenced by this attribute must be an overlay image (i.e., must have the "O" **STYLE**).

If the referenced image control contains more than one image, the image used is determined by the item's **OVERLAY-INDEX** attribute value.

An overlay image is a image that is superimposed on the item's base image. Since they belong to the same image list, the base and overlay images have the same dimensions. However, the transparent pixels of the overlay image allow the corresponding pixels in the base image to show through. For more information, please refer to the article Working with Image List Controls.

Applies to	List view item, tree view item.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE / handle value of image.

152 OVERLAY-INDEX

Specifies an item's overlay image index. This index determines the overlay image to be used from the **OVERLAY** image control associated with the item, or from its parent control's **IMAGE-LIST** if the former attribute is set to NULL-HANDLE.

The first case above, the overlay image index is the zero-based relative index of the image within the specified overlay image control, with wrap-around if the index is out of range. For example, if the referenced overlay image control contains three images, the overlay image indexes of 0, 3, 6, 9, and so on, will all select the first image.

In the second case above, the overlay image index is the one-based absolute position of the required image in the parent control's image list, with no wrap-around for out-of-range values. For example, if the parent control's image list contains 10 images, an overlay image index of 1 will select the first image from the first image control belonging to the image list. Similarly, an overlay image index of 10 will select the last image from the last image control belonging to the image list. Out-of-range image index values, such as 11 in this case, are simply ignored. Note that with this absolute indexing mechanism, Natural does not automatically track changes made to to the image list. For example, if image 9 is removed from a 10-image image list, causing the 10th image to suddenly become the 9th image, items with an image index of 10 are neither automatically updated to 9, nor automatically refreshed to reflect the change.

For more information, please refer to the article Working with Image List Controls.

Applies to	Image control, list view item, tree view item.
Data Type	I4
Default Value	0
Possible Values	Any positive value.

153 OWNER

Specifies a “belongs to” relationship between two objects other than a containment relationship (for which the **PARENT** attribute is used).

In particular, this is currently used to assign controls to tab control tabs, where the control's OWNER attribute is set to the handle of the tab control tab to which the control belongs. If this tab is selected, the control is automatically made visible if its **VISIBLE** attribute is set to FALSE. If the tab is deselected, the control is automatically hidden. If the tab is deleted, the control is automatically deleted. The foregoing only applies to controls that are direct children of the tab control. For indirect descendant controls, or for controls for which the OWNER attribute is not set, no automatic show, hide or delete occurs.

Applies to	ActiveX control, bitmap control, canvas control, control box control, date/time picker control, edit area control, group frame control, input field control, list box control, list view control, OLE container control, push button control, progress bar control, radio button control, scroll bar control, selection box control, slider control, spin control,tab control, table control, text constant control, toggle button control, tree view control.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE / owner handle.

154 P1-X

Determines a line control's start position on the x-axis.

Applies to	Line control.
Data Type	I4
Default Value	0
Possible Values	0 - 9999

155 P1-Y

Determines a line control's start position on the y-axis.

Applies to	Line control.
Data Type	I4
Default Value	0
Possible Values	0 - 9999

156 P2-X

Determines a line control's end (target) position on the x-axis.

Applies to	Line control.
Data Type	I4
Default Value	0
Possible Values	0 - 9999

157 P2-Y

Determines a line control's end (target) position on the y-axis.

Applies to	Line control.
Data Type	I4
Default Value	0
Possible Values	0 - 9999

158 PAGE

For scroll bar controls, this attribute determines the number of logical units by which the control's thumb (slider) moves in response to either pressing the PageUp or PageDown keys on the keyboard or by clicking within the control's shaft.

For slider controls, this attribute determines the number of logical units by which the control's thumb moves in response to either pressing the PageUp or PageDown keys on the keyboard or by clicking on the slider shaft on either side of the thumb.

Inside a numeric scale limited by the values of the control's **MIN** and **MAX** attributes, the value of this attribute determines the size of the steps by which the thumb will be moved. If you set a small value in relation to the **MIN** and **MAX** attributes, the thumb will move continuously. If you set a larger value, the thumb will move in visible steps.

Applies to	Scrollbar control, slider control.
Data Type	I4
Default Value	0
Possible Values	0 or any other integer value

159 PARENT

Determines the parent handle of a dialog or a dialog element in the hierarchy. At the top of the hierarchy are one or several base dialogs; these are the parents of the dialog elements contained in them. These children may themselves contain children, such as controls in a child dialog. Children on the same level are called siblings. They may be enumerated using the **SUCCESSOR** attribute with the same parent.

The parent/child relationship can only be changed by "deleting" a child and creating a new one.

Applies to	Date/time picker control, dialogs (all types) and all dialog elements, dialog bar control, image control, image list control, list view column, list view control, list view item, tab control, tab control tab, progress bar control, slider control, spin control, tree view control, tree view item, wallpaper control.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE / any dialog element

160 POSITION

Determines a dialog element's current value within the range of values supported by the dialog element, as defined by its **MIN** and **MAX** attributes.

An attempt to set the position to an out-of-range value causes an error to be raised.

Applies to	Progress bar control, spin control.
Data Type	I4
Default Value	0
Possible Values	Any in-range integer value.

161 POPUP-HELP

Determines whether help invoked for the dialog or one of its controls via the title bar help button or pressing F1 should be displayed in a popup window. A popup window is a tool tip style window usually containing only text, which is automatically removed when the window loses the focus.

Help for popup windows can be created via the Help Organizer in the same way as for non-popup help. For HTML help, it is currently necessary to enter popup help text into a separate text file, which is then compiled into the help file together with the HTML topics.

Note: To invoke the popup help in the latter case, it is necessary to explicitly specify the name of the text file in the **HELP-FILENAME** attribute via the syntax *helpfilename.chm::/textfilename.txt*. Please refer to your HTML authoring tool help for further information.

Applies to	Dialog (all types).
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	TRUE / FALSE

162 PREDECESSOR

Contains the handle of the previous dialog element with the same **PARENT** in the parent/child hierarchy, or NULL-HANDLE if no such dialog element exists. You can use this attribute to go through all dialog elements in one parent dialog or dialog element in reverse order.

Applies to	Date/time picker control, dialogs (all types) and all dialog elements, image control, image list control, list view column, list view control, list view item, progress bar control, slider control, spin control, tree view control, tree view item.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE / handle value of the predecessor.

163 RECTANGLE-H

Represents the width of a dialog element. If used with a column specification control or list view column, it represents the width of the column in pixels.

For list view columns, two special negative values for this attribute may be set after the column has been created:

-1 = Autosize the column to fit longest value displayed in the column. -2 = Autosize the (last) column to fit the remaining header width.

Applies to	ActiveX control, bitmap control, canvas control, control box control, date/time picker control, dialog (all types), dialog bar control, edit area control, graphic text control, group frame control, input field control, list box control, list view control, OLE container control, push button control, progress bar control, radio button control, rectangle control, scroll bar control, selection box control, slider control, spin control, status bar control, tab control, table control, text constant control, tool bar control, toggle button control, tree view control.
Data Type	I4
Default Value	0
Possible Values	0 - 9999

164 RECTANGLE-W

Represents the width of a dialog element. If used with a column specification control, it represents the width of the column in pixels.

Applies to	ActiveX control, bitmap control, canvas control, column specification control, control box control, date/time picker control, dialog (all types), dialog bar control, edit area control, graphic text control, group frame control, input field control, list box control, list view column, list view control, OLE container control, push button control, progress bar control, radio button control, rectangle control, scroll bar control, selection box control, slider control, spin control, status bar control, status bar pane, tab control, table control, text constant control, tool bar control, toggle button control, tree view control.
Data Type	I4
Default Value	0
Possible Values	0 - 9999

165 RECTANGLE-X

Represents the dialog element's location on the x-axis relative to the top left corner of its parent's client area.

Applies to	ActiveX control, bitmap control, canvas control, control box control, date/time picker control, dialog (all types), dialog bar control, edit area control, graphic text control, group frame control, input field control, list box control, list view control, list view item, OLE container control, push button control, progress bar control, radio button control, rectangle control, scroll bar control, selection box control, slider control, spin control, status bar control, tab control, table control, text constant control, tool bar control, toggle button control, tree view control.
Data Type	I4
Default Value	0
Possible Values	0 - 9999

166 RECTANGLE-Y

Represents the dialog element's location on the y-axis relative to the top left corner of its parent's client area.

Applies to	ActiveX control, bitmap control, canvas control, control box control, date/time picker control, dialog (all types), dialog bar control, edit area control, graphic text control, group frame control, input field control, list box control, list view control, list view item, OLE container control, push button control, progress bar control, radio button control, rectangle control, scroll bar control, selection box control, slider control, spin control, status bar control, tab control, table control, text constant control, tool bar control, toggle button control, tree view control.
Data Type	I4
Default Value	0
Possible Values	0 - 9999

167 ROW

A ROW attribute value corresponds to exactly one **COLUMN** attribute value. The combination of these two denotes a cell inside a table control.

- To denote a cell of the table: COLUMN must be >= 1 and ROW must be >=1.
- To denote the first column in the table: COLUMN must be 0 and ROW must be 1.
- To denote the header row in the table: COLUMN must be 1 and ROW must be 0.

Applies to	Table control.
Data Type	I4
Default Value	1
Possible Values	0 - 9999

168 ROW-COUNT

Enables you to initialize or query the current number of rows in a table control, query the current number of rows of tabs in a tab control, or query the current number of items in a list box, list view control, or selection box control.

Applies to	List box control, list view control, selection box control, tab control, table control.
Data Type	I4
Default Value	1 (for table controls)
Possible Values	0 - 9999 (for table controls)

169 ROW-HEIGHT

Determines the height of all rows in a table control (except the header row).

Applies to	Table control.
Data Type	I4
Default Value	0
Possible Values	0 - 9999

170 RTL

If this attribute is set, the dialog element direction is right-to-left. You can use this attribute to change the direction for elements which are designated for the input of characters from bidirectional languages.

This attribute has only an effect if both the Natural default code page and the Windows system code page are defined as bidirectional code page. If Natural does not define a specific code page, it is sufficient to have defined a bidirectional Windows system code page.

Applies to	Dialogs (all types), edit area control, group frame control, input field control, list box control, radio button control, scrollbar control, selection box control, table control, toggle button control.
Data Type	BOOLEAN
Default Value	depends on profile parameter `PM`
Possible Values	TRUE/FALSE

171 SAME-AS

Specifies an equivalence relationship between two controls (“objects”), such that both objects are considered to represent the same user action (“command”). This allows Natural to ensure the coherency of the user interface implicitly. Coherency implies that all user interface elements representing the same command have the same state at all times. For example, if a menu item and a tool bar item represent the same command, then, at any one time, either both should be enabled, or both disabled. Furthermore, Natural ensures that the same event-handling code is executed when a command is invoked, regardless of which menu or tool bar item invoked it.

The referenced object may be a signal or a menu item. The referencing object may be a tool bar item or a menu item. A SAME-AS relationship between two menu items is not allowed. A two-step approach, whereby a tool bar item references a menu item, which itself references a signal, is permitted, but not recommended. Instead, the tool bar item should, like the menu item, reference the signal directly. Note that a SAME-AS relationship between objects belonging to different dialogs is permitted. Thus, objects in MDI child dialogs may reference objects in the MDI frame dialog.

For each command, the SAME-AS relationships form a dependency tree. The object within the tree that does not have a SAME-AS relationship to another object is the *root* object. Any object in the tree to which one or more SAME-AS relationships exist is referred to as a *referenced* object, and the objects which (directly or indirectly) reference it are referred to as that object's *dependent* objects.

- **ACCELERATOR**
- **BACKGROUND-COLOUR-NAME**
- **BACKGROUND-COLOUR-VALUE**
- **BITMAP-FILE-NAME**
- **CHECKED**
- **DIL-TEXT**
- **ENABLED**

- **MENU-ITEM-TYPE**
- **SHARED**
- **TOOLTIP**
- **VISIBLE**

Propagation implies that the same attribute is implicitly set to the same value for each object to which the attribute setting is propagated. Furthermore, when a SAME-AS relationship is set to any value other than NULL-HANDLE, the values of all the above attributes are immediately copied from the referenced object to the referencing object. Setting the SAME-AS attribute to NULL-HANDLE breaks the relationship and restores the object's creation-time attribute values.

In addition, a click received for a non-root object results in a CLICK event being raised for the root object instead. This ensures that the same code is executed regardless of how the command is accessed.

As an example, suppose a command may be accessed via a menu bar menu item, a context menu item, or a tool bar item. Instead of maintaining a set of state attributes for each of these three objects, it is easier to create a signal to represent the command, and to set the SAME-AS attribute of all three objects to the handle of this signal. Now, only the signal's attributes need to be maintained by the program. For example, setting the **CHECKED** attribute of the signal to TRUE implicitly also sets the **CHECKED** attribute of both menu items and the toolbar item to TRUE, thus checking them. Furthermore, regardless of which one of the menu or tool bar items the user clicks, a CLICK event is received for the signal instead.

Applies to	Menu item, tool bar item.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE / any signal or (for tool bar items only) menu item handle

172 SCROLLRANGE-X

Determines the horizontal range within which the end user may scroll in a dialog window. The dialog editor automatically determines the scroll range so that it encloses all dialog elements. If the dialog's scroll bars are enabled, they are displayed if the dialog's size is smaller than the scroll range.

Applies to	Dialog (all types).
Data Type	I4
Default Value	0
Possible Values	0 - 9999

173 SCROLLRANGE-Y

Determines the vertical range within which the end user may scroll in a dialog window. The dialog editor automatically determines the scroll range so that it encloses all dialog elements. If the dialog's scroll bars are enabled, they are displayed if the dialog's size is smaller than the scroll range.

Applies to	Dialog (all types).
Data Type	I4
Default Value	0
Possible Values	0 - 9999

174 SELECTED-SUCCESSOR

If applied to a control (rather than to an item), this attribute returns the handle of the first selected item (or tab, in the case of tab controls) within the control. For controls supporting multiple selection (list box and list view controls), the attribute may also be applied to an item, in which case the handle of the next selected item (if any) is returned.

The value NULL-HANDLE indicates that no (further) selected items exist.

For programming techniques related to the SELECTED-SUCCESSOR attribute, please refer to Working with List Box Controls and Selection Box Controls in the Event-driven Programming Techniques documentation.

Applies to	List box control, list box item, list view control, list view item, selection box control, tab control, tree view control.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE / handle of list box item, selection box item or tab.

175 SELECTED

Determines whether the specified item is selected.

Applies to	List box item, list view item, selection box item, tab control tab, tree view item.
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	TRUE / FALSE

176 SHARED

Determines the MDI dialog which is to receive the event notifications for a dialog element. If this attribute is set to FALSE, the events are sent to the owning dialog (typically the MDI frame dialog). If this attribute is set to TRUE, the events are sent to the active MDI child dialog (if any), or to the MDI frame dialog if no MDI child dialog is active. For non-MDI dialogs, this attribute has no effect.

Applies to	Menu item, signal, status bar pane, tool bar item.
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	TRUE / FALSE

177 SIZE-MODIFIABLE

Determines whether the dialog's size is modifiable by the end user or not. This implies that the dialog is surrounded by a thick border.

Applies to	Dialog (window, MDI frame, MDI child).
Data Type	BOOLEAN
Default Value	TRUE
Possible Values	TRUE / FALSE

178 SLIDER

Determines the position of the thumb (slider) for a dialog element, within the numeric range specified by the values of the **MIN** and **MAX** attributes.

Applies to	Scrollbar control, slider control.
Data Type	I4
Default Value	0
Possible Values	All integer values between the MIN and the MAX value of the dialog element.

179 SORTED

For list box controls and selection box controls, this attribute determines whether items are inserted into the control in sorted sequence.

For tree view controls, this attribute determines whether items are inserted into the control in ascending alphabetic sequence.

For tree view items, this attribute determines whether the item is inserted into the control in ascending alphabetic sequence. This is therefore only useful if the SORTED attribute is not set for the tree view control itself, and allows tree views to be partially sorted. In this case, the program should ensure that all tree view items with the same **PARENT** (i.e., sibling items) should have the same SORTED attribute value, as insertion based on sort sequence requires that the existing sibling items are already in sorted sequence.

For list view controls, this attribute determines whether items are inserted into the control in alphabetic sequence. If so, the value of the **DESCENDING** attribute determines whether the items are inserted in ascending or descending sequence.

For list view columns, this attribute may only be queried, and returns TRUE if the column is sorted (i.e., if the header, if any, contains a sort indicator). This does not necessarily imply that the column's values are still in sorted sequence, however, because changes could have been made to the column items or their values since the column sort was performed. The **DESCENDING** attribute determines whether an ascending or descending sort was performed on the column (i.e., the direction of the sort indicator, if any).

For more information, please refer to the articles Working with List View Controls and Working with Tree View Controls.

Applies to	List box control, list view column, list view control, selection box control, tree view control, tree view item.
Data Type	BOOLEAN
Default Value	FALSE
Possible Values	TRUE / FALSE

180 SERVER-OBJECT

Determines the file name of an external OLE object associated with an OLE server application. If you assign this value, the corresponding object is loaded into the OLE container control. When this object is loaded, the current value of the attributes **SERVER-PROGID** and **EMBEDDED-OBJECT** is removed.

Applies to	OLE container control.
Data Type	A253
Default Value	None
Possible Values	Any file name

181 SERVER-PROGID

Determines the programmatic identifier of an OLE server application. If you assign this value, a specific OLE server application is selected for the OLE container control. Any object previously loaded into the OLE container control is removed. The current values of the attributes **SERVER-PROGID** and **EMBEDDED-OBJECT** are also removed.

Applies to	OLE container control.
Data Type	A253
Default Value	None
Possible Values	Any program ID

182 SPACING

Specifies the horizontal spacing (in pixels) between adjacent columns of list view items for list-based list view controls (i.e., where the **VIEW-MODE** attribute is set to VM-LIST), or the horizontal indentation of a tree view control's items relative to their respective parent. A value of zero is interpreted as “unspecified”, implying that the control's default spacing is used.

Note that the spcified value is automatically scaled on control creation, if necessary, if the parent dialog's **AUTO-ADJUST** attribute is set to TRUE.

For more information, please refer to the articles Working with List View Controls and Working with Tree View Controls.

Applies to	List view control, tree view control.
Data Type	I4
Default Value	0
Possible Values	0 - 9999

183 SPACING-X

Specifies the horizontal spacing (in pixels) between adjacent columns of arranged list view items for icon-based list views. A value of zero is interpreted as “unspecified”, implying that the control's default horizontal spacing is used.

This attribute, together with the **SPACING-Y** attribute, thus determines the size of the logical grid used by the **ARRANGE** action.

Note that the spcified value is automatically scaled on control creation, if necessary, if the parent dialog's **AUTO-ADJUST** attribute is set to TRUE.

For more information, please refer to the article Working with List View Controls.

Applies to	List view control.
Data Type	I4
Default Value	0
Possible Values	0 - 9999

184 SPACING-Y

Specifies the vertical spacing (in pixels) between adjacent rows of arranged list view items for icon-based list views. A value of zero is interpreted as “unspecified”, implying that the control's default vertical spacing is used.

This attribute, together with the **SPACING-X** attribute, thus determines the size of the logical grid used by the **ARRANGE** action.

Note that the spcified value is automatically scaled on control creation, if necessary, if the parent dialog's **AUTO-ADJUST** attribute is set to TRUE.

For more information, please refer to the article Working with List View Controls.

Applies to	
Data Type	
Default Value	
Possible Values	

185 STATUS-HANDLE

Specifies the status bar control (if any) which is to receive output in the case where no status bar control handle is explicitly specified (for example, when the **STATUS-TEXT** attribute of the dialog is changed). If the value of this attribute is NULL-HANDLE, the output will be sent to the implicit status bar (if any) that is created when the dialog's **HAS-STATUS-BAR** attribute is set to TRUE.

Applies to	Dialog (window, MDI frame, MDI child).
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE / status bar control handle

186 STATUS-TEXT

By modifying the value of this attribute, you display the assigned text in the status bar of a dialog. The text will only be displayed if **HAS-STATUS-BAR** is TRUE. Then the **DIL-TEXT** will overlap the STATUS-TEXT value or vice versa, depending on which was modified last. For MDI child windows, the status bar is located in the MDI frame window.

Applies to	Dialog (window, MDI frame, MDI child).
Data Type	A253
Default Value	Empty string
Possible Values	Empty string / any text string

187 STRING

Determines the text string associated with a dialog element, for example, the title of a dialog window or the label of a push button control. For a font control, this text string is invisible to the end user.

By default, the system font is used to display the STRING. You should, however, be aware that the size of the system font depends on your operating system and on your screen driver. As a result of this, a STRING may not fit in the dialog element's rectangle.

Applies to	Column specification control, dialog (all types), dialog bar control, edit area control, font control, graphic text control, group frame control, input field control, list box item, list view column, list view item, menu item, push button control, radio button control, selection box control, selection box item, status bar control, status bar pane, tab control tab, table control, text constant control, tool bar control, toggle button control, tree view item.
Data Type	A253
Default Value	Empty string
Possible Values	Empty string / any text string

188 STYLE

Determines a set of properties of the above-mentioned dialog elements. For one instance of a dialog element, several properties may be set. The style of a dialog element is represented by a set of characters. Each character determines a property specific to the dialog element.

Applies to	ActiveX control, bitmap control, canvas control, column specification control, date/time picker control, dialog (all types), dialog bar control, edit area control, graphic text control, image control, image list control, input field control, line control, list view column, list view control, list view item, OLE container control, progress bar control, push button control, rectangle control, scroll bar control, selection box control, slider control, spin control, status bar control, status bar pane, tab control, table control, text constant control, tool bar, tool bar control, tool bar item, tree view control, tree view item, wallpaper control.
Data Type	A253

Possible Values for ActiveX Controls

Value	Meaning
O	OK button: is pushed if the end user presses ENTER.
C	Cancel button: is pushed if the end user presses ESC.
x	Disable accelerators. Disables usage of any accelerator keys defined by the control.

Note: The styles "O" and "C" apply only to ActiveX controls that behave like buttons (i.e., are marked in the system registry with the style OLEMISC_ACTSLIKEBUTTON).

Possible Values for Bitmap Controls

Value	Meaning
c	Align the bitmap to the center of the dialog element's rectangle.
l	Align to the left.
r	Align to the right.
v	Align to the vertical center.
t	Align to the top.
b	Align to the bottom.
s	Scale the bitmap to fit into the dialog element's rectangle.
F	Three-dimensional frame.
T	Transparent: Bitmap pixels matching the background color are not drawn. If the background color is specified as “default”, the color of the first pixel in the bitmap is assumed to be the background color.

Possible Value for Canvas Controls

Value	Meaning
F	Frame (border) around the canvas control (reduces the visible area of the canvas).

Possible Values for Control Box Controls

Value	Meaning
F	Framed: Frame around the control box control.
L	Lowered: Creates a 3-D border with a sunken appearance.
E	Exclusive: Hide all other sibling control boxes with this style automatically when this control is made visible.
T	Transparent: (The control itself is not visible, but children may be).
z	Size to parent: Resize control to fill parent's client area whenever parent is resized.

Possible Values for Date and Time Picker (DTP) Controls

Value	Meaning
s	Short date (default). The control displays date information in the system-defined short date format, which may or may not include century information.
c	Century date. The control displays date information in the system-defined short date format, but with century information. **Note:** Depending on the current regional settings, this may be identical to the short date format.
1	Long date. The control displays date information in the system-defined long date format.

Possible Values for Dialog Bar Controls

Value	Meaning
g	Gripper: Control has a gripper bar.
Y	Dynamic: Control can be resized when floated or docked.
3	3-D border: Dialog interior drawn with sunken 3-D appearance.
R	Raised: Control has a raised interior.
x	Control has a close button.
z	Control has a zoom button.
T	UI Transparent: The control can only be dragged via the gripper bar (if present) and not via clicking elsewhere in the control.

Possible Values for Dialogs (WINDOW)

Value	Meaning
c	Center dialog on screen.
P	Modeless (Pop-up; window is independent of its parent).
m	Modal.
x	Dialog box.
r	Relative position: The position is interpreted as being relative to the parent (owner) dialog.
d	Default position; will be ignored if “Dialog box” is set as well.
D	Default rectangle; will be ignored if “Dialog box” is set as well.
Z	Control clipping. Prevents controls overprinting other controls which are “in front” in the so-called Z order. The position of a control in the Z-order sequence is determined via the control's SUCCESSOR attribute.
3	3-D border: Dialog interior drawn with sunken 3-D appearance.
p	Property sheet-like navigation: If the dialog contains a tab control, the Ctrl-Tab and Ctrl-Shift-Tab key combinations are used for browsing forwards and backwards (respectively) between the tab control's tabs.

Possible Values for Edit Area Controls

Value	Meaning
w	Wordwrapped (if text exceeds the width of the edit area control, it is automatically wrapped to the next line).
F	Frame around the edit area control.
v	Autoscroll: Text is automatically scrolled upwards if ENTER key is pressed on the last displayed line. This style also enables scrolling in the absence of a vertical scroll bar (e.g. via cursor keys or mouse wheel). **Note:** An edit area with a vertical scroll bar is implicitly autoscrollable.

Possible Value for Group Frame Controls

Value	Meaning
p	Group frame becomes parent of any controls placed within it.

Possible Value for Image Controls

Value	Meaning
s	Scaled. If the required width or height of the rendered image differ from the width or height of the image(s) taken from the image file, the images are scaled, rather than being either extended in the background color or truncated.
T	Transparent. Image is rendered transparently. For bitmap image files (*.bmp), bitmap pixels matching the background color are not drawn. If the background color is specified as "default", the color of the first pixel in the bitmap is assumed to be the background color. For icon image files (*.ico), where the background color is "default" and no scaling needs to be performed on the image, the transparency mask provided with the icon is used. Otherwise, the icon's bitmap is handled as for bitmap image files and the transparency mask is ignored.
C	Composite image. Image bitmap file assumed to consist of multiple images joined together horizontally into one large bitmap, based on the image list's **ITEM-H** and **ITEM-W** attributes, or (if zero) the system small icon size if the image list's "Small images (S)" style is specified, or the system large icon size (typically 32 by 32 pixels) if the image list's "Large images (L)" style is specified. If none of these are specified, the system small icon size (typically 16 by 16 pixels) is assumed. This style cannot be used for icon image files.
O	Overlay image. An overlay images is an images that may be superimposed on the base image displayed for an item. Overlay images may be specified via the **OVERLAY** and/or **OVERLAY-INDEX** attributes, and are always implicitly rendered transparently.

Possible Value for Image List Controls

Value	Meaning
L	Large images. Indicates that the image list consists of images corresponding to the system large icon size (typically 32 by 32 pixels). This should be set if the image list is used to source large icons for a control (e.g., for the Icon view mode of a list view control).
S	Small images. Indicates that the image list consists of images corresponding to the system small icon size (typically 16 by 16 pixels). This should be set if the image list is used to provide small icons for a control (e.g., for a tree view control).

Notes:

1. Both of the above styles may be specified, in which case the image list control consists of both large and small images (e.g. for list view controls).
2. If neither of the above styles are specified, the image size is determined by the **ITEM-W** and **ITEM-H** attributes. If either of these is zero, the system small icon width or height (respectively) is assumed.

Possible Values for Input Field Controls

Value	Meaning
l	Left-justified input.
r	Right-justified input.
c	Horizontally centered input.
N	Non-displayed input (for example, for passwords).
U	Upper-case input. Input is converted to upper case.
L	Lower-case input. Input is converted to lower case.
M	Input is mandatory.
d	Digits only. Only the digits 0.. 9 may be input.

Possible Values for Line Controls and Rectangle Controls

Value	Meaning
S	Solid line (default).
-	Dashed line pattern.
.	Dotted line pattern.
!	Dash-dot line pattern.
:	Dash-dot-dot line pattern.

Possible Value for List Box Control

Value	Meaning
3	3-D border: Border with sunken 3-D appearance around the list box control.
I	Integral height. Partial rows are not displayed.
i	Insertion mark. An insertion mark appears indicating the would-be insert position when the list box is used as a target in a drag-drop operation and a drop is allowed.

Possible Values for List View Controls

Value	Meaning
3	3-D border: Border with sunken 3-D appearance around the list view control.
v	Align vertically. Items are arranged vertically (instead of horizontally) in the icon view modes.
a	Auto-arrange. Items are always automatically kept arranged in the icon view modes.
r	Snap to grid. (Re-)positioned items are automatically snapped to their nearest grid position in the icon view modes.
n	No scroll. Disables both horizontal and vertical scrolling in the control. The control will not display any scroll bars.
x	No header. No column headers are displayed in report view mode.

Value	Meaning
y	No sort header. List view columns may not be sorted by clicking their header (i.e., the column headers are not click-sensitive).
c	Check boxes. Check boxes are displayed for all items. The checked state may be queried or changed via the **CHECKED** attribute for a list view item.
f	Full row select. Selection emphasis extends across all columns in report view mode, and it is possible to select an item by selecting anywhere within the row.
g	Grid lines. Grid lines are displayed in the report view.
d	Header drag. The user may re-order columns in the report view by dragging their headers.
p	Label tip. If not enough space exists to display an items label completely, the full label is displayed in a tooltip window when the mouse cursor hovers over the label. **Note:** An item's label tip (if any) is only displayed if there is no custom tooltip text associated with the item via its **TOOLTIP** attribute, or if custom tooltip display has been suppressed via the control's **HAS-TOOLTIP** attribute.
w	Wrap icon labels (default). Long item labels may be wrapped to spread over multiple lines.
u	Underline hot. Label of item under mouse cursor (if any) is underlined and emphasized in a different color. Only a single click is needed to activate items.
U	Underline cold. All items labels are underlined. Only a single click is needed to activate items.
s	No hide selection. Keep selection visible even when the control does not have the focus.
b	Border select. Use thick border to highlight selected items in icon view mode. No dot screen is applied over the selected icons in any mode, thus making them more legible.
t	Hot-track select. The item under the mouse cursor is selected automatically if the mouse cursor hovers over it for a short time.
m	Marquee select (default). In multiple selection list views, items may be selected by "rubber banding" (i.e., defining a drag rectangle containing the items to be selected).
z	Size to parent: Resize control to fill parent's client area whenever parent is resized.

Possible Values for List View Columns

Value	Meaning
1	Left. Column title is left-aligned (default).
c	Center. Column title is centered. This option has no effect on the primary column (the column containing the item labels).
r	Right. Column title is right-aligned. This option has no effect on the primary column (the column containing the item labels).
i	Case insensitive. Columns containing alphanumeric data are sorted case-insensitively. By default, the sort is case-sensitive.
w	Word compare. Columns containing alphanumeric data are sorted using a word compare. By default, a simple string comparison is made. In a word compare, the hyphen and the apostrophe characters are treated differently, in order to ensure that words such as "coordinate" and "co-ordinate" stay together within a sorted list.

Possible Value for List View Items

Value	Meaning
U	Upper case. When an item label is edited, lower-case characters are automatically converted to upper-case. Note that this option does not apply to text that is set programatically, via the item's **STRING** attribute.

Possible Values for Menu Items

Value	Meaning
s	Scaled: Bitmaps are scaled to match the image height and width specified for the parent menu.
T	Transparent: Bitmap pixels matching the background color are not drawn. If the background color is specified as “default”, the color of the first pixel in the bitmap is assumed to be the background color.
D	Default: Menu item text is displayed in a bold font.

Possible Value for OLE Container Controls

Value	Meaning
F	Framed: Frame around the OLE container control.

Possible Values for Progress Bar Controls

Value	Meaning
s	Smooth. The control's progress bar is non-segmented. **Note:** This option is ignored if Windows XP styles are in use. The progress bar is always segmented in this case.
v	Vertical. The control is oriented vertically, rather than horizontally.

Possible Values for Push Button Controls

Value	Meaning
O	OK button: is pushed if the end user presses ENTER.
C	Cancel button: is pushed if the end user presses ESC.

Possible Values for Scrollbar Controls

Value	Meaning
h	Slider will scroll horizontally.
v	Slider will scroll vertically.

Note: When you edit the STYLE attribute value in the scroll bar control attributes window, setting "h" instead of "v" and vice versa, the **RECTANGLE-H** and **RECTANGLE-W** attribute values are exchanged. The dialog editor thus ensures that the scroll bar control will not provide for vertical scrolling in a horizontal shape and vice versa.

Possible Values for Selection Box Controls

Value	Meaning
M	Input into this selection box control is mandatory.
X	Box is pulled down all the time.
U	Upper-case input.

Possible Values for Slider Controls

Value	Meaning
a	Auto ticks (default). Tick marks are displayed automatically, at intervals determined by the control's **SPACING** attribute. If this option is not specified, tick marks must be explicitly added via the **SET-TICKS** action. **Note:** The tick marks at the lower and upper end of the slider range are permanent, and are not affected by this option.
1	Side 1 ticks (default). Tick marks are displayed at the top or left side of the control, if the control is oriented horizontally or vertically (respectively).
2	Side 2 ticks. Tick marks are displayed at the bottom or right side of the control, if the control is oriented horizontally or vertically (respectively).
v	Vertical. The control is oriented vertically, rather than horizontally.
p	Position tip. When the slider is used, a toolip window is shown to indicate the current position.
n	No thumb. The control does not display a thumb.

Possible Values for Spin Controls

Value	Meaning
h	Horizontal. The control is oriented horizontally rather than vertically.
1	Left align. The control has an implicit buddy input field control that is displayed to the right of the control's Up and Down buttons.
r	Right align (default). The control has an implicit buddy input field control that is displayed to the left of the control's Up and Down buttons.
s	Set buddy (default). The contents of the buddy input field control (if any) are automatically updated to reflect the spin control's position.
w	Wrap. When the control's upper limit has been reached, the control's value automatically wraps around to the lower limit, and vice versa.
k	Arrow keys (default). Causes the control to increment or decrement the position when the Up Arrow and Down Arrow keys are pressed, respectively.
n	No thousands (default). Thousands separators are not used. For example, “1,200,000” is displayed as “1200000” in the buddy input field control (if any). **Note:** This option only has an effect if the “Set buddy (s)” STYLE is also set.
t	Hot tracking. Causes the up and down arrow buttons to be highlighted when the mouse cursor moves over them. **Note:** This option is implicitly set if Windows XP styles are in use.
x	Hexadecimal. The buddy input field control (if any) displays values in hexadecimal (rather than decimal) notation (e.g., “0x001E” instead of “30”). **Note:** This option only has an effect if the “Set buddy (s)” STYLE is also set.

Possible Values for Status Bar Controls

Value	Meaning
3	3-D border: border will be drawn with a 3-D appearance.
t	Top border: border will be drawn along top edge of control.
b	Bottom border: border will be drawn along bottom edge of control.
g	Gripper: control has a sizing grip.

Possible Values for Status Bar Panes

Value	Meaning
c	Centered: text will be centered within the pane.
H	Hide pane text (instead of graying it out) if pane is disabled.
R	Raised: pane will be drawn with a raised 3-D appearance.
n	No borders: pane will be drawn without borders.

Possible Value for Submenus and Context Menus

Value	Meaning
c	Cool menu: menu items are drawn with their bitmap (if any).
F	Keep focus: the focus is not automatically transferred to the clicked dialog element. This style applies to context menus only.

Possible Values for Tab Controls

Value	Meaning
b	Bottom tabs: Tabs are displayed at the bottom of the control instead of at the top.
f	Fixed-width tabs: All tabs have a constant width, as specified by the ITEM-W attribute. **Note:** This style does not apply if the tabs are aligned on both sides of the control.
m	Multi-row: Tabs are arranged in multiple rows, if necessary, instead of being on a single (scrollable) row.
R	Ragged: Tab widths are not automatically extended for tab controls with the “multi-row” style to ensure that the left and right edges of each tab row are aligned to the corresponding edges of the control.
x	Left icon: The tab's icon (if any) is left-aligned.
y	Left label: The tab's label is left-aligned. This style automatically implies the “left icon” style.
z	Browsable: The control can receive the focus, whereupon the active tab is drawn with focus emphasis. It is then possible to navigate (browse) between the tabs using the cursor (arrow) keys.
U	UI active: Switching between tabs is possible in the dialog editor. In addition, any controls created within the tab control are owned by the currently selected tab (see **OWNER** attribute) and are automatically hidden and shown when this tab is deselected and re-selected, respectively (both in the editor and at run-time).
-	Disable theme: Disables the standard tab control theme (if any) used for drawing the control. The control appears in the “Classic” Windows style. Note that this style has no effect on Windows versions prior to Windows XP.

Possible Values for Table Controls

Value	Meaning
h	Columns header: buttons with field names are displayed at the top of each column.
I	Integral height: partial rows are not displayed.
n	No lines: the table control is displayed without the lines that normally separate the cells.
s	Single cell selection: If this attribute is set, end users may only select single cells. If not set, end users may select ranges of cells.
w	Whole row selection: selecting an individual cell sets the selection to the entire row.
e	Extendable: end users can delete and insert rows using DEL and INS.
c	Resize columns: end users may resize the columns horizontally.
r	Resize rows: end users may resize the rows vertically.
d	Draggable columns: If this attribute is set, end users may drag the columns.

Possible Values for Text Constant Controls

Value	Meaning
l	Left-justified input.
r	Right-justified input.
c	Horizontally centered input.
F	Frame around the text constant control.

Possible Values for Tool Bars

Value	Meaning
w	Wrapped: if set and there are more tool bar items than can be displayed on the top of the dialog, the tool bar wraps around to a new line. (The default: the tool bar can be scrolled with the two small arrow push buttons on the left of the tool bar).

Possible Values for Tool Bar Controls

Value	Meaning
3	3-D border: border will be drawn with a 3-D appearance.
t	Top border: border will be drawn along top edge of control. This style is implicitly set for dockable tool bar controls.
b	Bottom border: border will be drawn along bottom edge of control. This style is implicitly set for dockable tool bar controls.
l	Left border: border will be drawn along left edge of control. This style is implicitly set for dockable tool bar controls.
r	Right border: border will be drawn along right edge of control. This style is implicitly set for dockable tool bar controls.

Value	Meaning
g	Gripper: control has a gripper bar.
L	Tool bar items are drawn with a flat (rather than raised) appearance.
Y	Dynamic: tool bar control can be resized when floated, but cannot contain any child controls.

Possible Values for Tool Bar Items

Value	Meaning
s	Scaled: bitmaps are scaled to fit on the tool bar items.
T	Transparent: Bitmap pixels matching the background color are not drawn. If the background color is specified as “default”, the color of the first pixel in the bitmap is assumed to be the background color.

Possible Values for Tree View Controls

Value	Meaning
3	3-D border: Border with sunken 3-D appearance around the list view control.
b	+/- buttons. Displays plus (+) and minus (-) buttons next to parent items. The user can click these buttons to expand or collapse a parent item's list of child items. **Note:** These buttons are only shown for root items if the “Lines at root (l)” STYLE is also specified.
1	Lines. Lines are used to show the item hierarchy.
r	Lines at root. Lines are used to link items at the root of the item hierarchy.
n	No scroll. Disables both horizontal and vertical scrolling in the control. The control will not display any scroll bars.
x	Single expand. Items are automatically expanded when selected, and automatically collapsed when deselected (unless the >Ctrl< key is being held down). Clicking again on a selected item causes the item to be collapsed.
d	Double click expand (default). An item may be expanded by double-clicking it.
c	Check boxes. Check boxes are displayed for all items. The checked state may be queried or changed via the **CHECKED** attribute for a list view item.
f	Full row select. Selection emphasis extends across the entire control, and it is possible to select an item by selecting anywhere within the “row” containing the item.
s	No hide selection. Keep selection visible even when the control does not have the focus.
t	Hot-track select. The item under the mouse cursor is selected automatically if the mouse cursor hovers over it for a short time.
z	Size to parent: Resize control to fill parent's client area whenever parent is resized.

Possible Value for Tree View Items

Value	Meaning
U	Upper case. When an item label is edited, lower-case characters are automatically converted to upper-case. Note that this option does not apply to text that is set programatically, via the item's **STRING** attribute.
n	No check box. No check box is displayed for this item. **Note:** This style is ignored if the control's "Check boxes (c)" STYLE is not set.

Possible Values for Wallpaper Controls

Value	Meaning
c	Align the bitmap to the center of the host dialog or dialog "pattern"element's client area. Like the other alignment styles, this style does not apply if the style is set.
l	Align to the left.
r	Align to the right.
v	Align to the vertical center.
t	Align to the top.
b	Align to the bottom.
p	Pattern: The wallpaper bitmap is repeated (tiled) to fill the entire client area of the host dialog or dialog element.
T	Transparent: Bitmap pixels matching the background color are not drawn. If the background color is specified as "default", the color of the first pixel in the bitmap is assumed to be the background color.

189 SUCCESSOR

Determines the next sibling of a dialog element in the parent/child hierarchy. A SUCCESSOR refers to the next created dialog element on the same level (with the same PARENT). You can use this attribute to go through all dialog elements in one parent dialog or dialog element.

Applies to	Date/time picker control, dialogs (all types) and all dialog elements, image control, image list control, list view column, list view control, list view item, progress bar control, slider control, spin control, tree view control, tree view item.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE / handle value of the successor

190 SUPPRESS-AFTER-EDIT-EVENT

Determines whether the **After-Edit Event** will be suppressed for the dialog element or not. If it is suppressed, the dialog element will not get control after the label of one of its items has been edited.

Applies to	List view control, tree view control.
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

191 SUPPRESS-BEFORE-EDIT-EVENT

Determines whether the **Before-Edit Event** will be suppressed for the dialog element or not. If it is suppressed, the dialog element will not get control when an attempt is made to edit the label of one of its items.

Applies to	List view control, tree view control.
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

192 SUPPRESS-BEFORE-OPEN-EVENT

Determines whether or not the **Before-Open Event** will be suppressed for the context menu (or submenu). If it is suppressed, the event handler will not get control when a context menu (or submenu) is displayed. Note that this attribute cannot be used to suppress the before-open event for a dialog.

Applies to	Context menus, Submenus
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

193 SUPPRESS-BEGIN-DRAG-EVENT

Determines whether or not the **Begin-Drag Event** will be suppressed for the dialog element. If it is suppressed, the dialog element will not get control when a drag-drop operation is commenced and the dialog element is used as the drag source. This leaves the drag-drop clipboard empty and thus causes the drag-drop operation to be implicitly cancelled.

Applies to	ActiveX control, bitmap control, list box control, list view control, tree view control.
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

194 SUPPRESS-CLIENT-SIZE-EVENT

Determines whether or not the **Client-Size Event** will be suppressed for the dialog. If it is suppressed, the event handler will not get control when the dialog's interior size changes. The dialog interior is the part of the dialog that does not include the menu bar (if any), tool bars (if any) or status bar (if any).

Applies to	Dialog (all types).
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

195 SUPPRESS-DBL-CLICK-EVENT

Determines whether or not the **Double-Click Event** will be suppressed for the dialog element. If it is suppressed, the event handler will not get control when the end user double-clicks an item in a list box control.

Applies to	Bitmap control, canvas control, column specification control, list box control, OLE container control, status bar pane, table control.
Data Type	I4
Default Value	0 (NOT-SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

196 SUPPRESS-DELETE-ROW-EVENT

Determines whether or not the **Delete-Row Event** will be suppressed for the dialog element. If it is suppressed, the event handler will not get control when the end user deletes a row.

Applies to	Table control.
Data Type	I4
Default Value	0 (NOT-SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

197 SUPPRESS-INSERT-ROW-EVENT

Determines whether or not the **Insert-Row Event** will be suppressed for the dialog element. If it is suppressed, the event handler will not get control when the end user is about to insert a row.

Applies to	Table control.
Data Type	I4
Default Value	0 (NOT-SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

198 SUPPRESS-TOP-EVENT

Determines whether or not the **Top Event** will be suppressed for the dialog element. If it is suppressed, the event handler will not get control when the end user navigates to the top of a table control.

Applies to	Table control.
Data Type	I4
Default Value	0 (NOT-SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

199 SUPPRESS-ACTIVATE-EVENT

Determines whether or not the **Activate Event** will be suppressed for the dialog element. If it is suppressed, the event handler will not get control when the corresponding OLE server application has just been activated in place, or when a list view item or tree view item is double-clicked (for example).

Applies to	List view control, OLE container control, tree view control.
Data Type	I4
Default Value	0 (NOT-SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

200 SUPPRESS-CHANGE-EVENT

Determines whether or not the **Change Event** will be suppressed for the dialog element. If it is suppressed, the event handler will not get control when the content of an input field control changes, for example.

Applies to	Canvas control, column specification control, date/time picker control, edit area control, input field control, OLE container control, scroll bar control, selection box control, slider control, spin control tab control, table control.
Data Type	I4
Default Value	0 (NOT-SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

201 SUPPRESS-CHECK-EVENT

Determines whether the **Check Event** will be suppressed for the dialog element or not. If it is suppressed, the dialog element will not get control when one of its items is checked.

Applies to	List view control, tree view control.
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

202 SUPPRESS-CLIPBOARD-STATUS-EVENT

Determines whether or not the **Clipboard-Status Event** will be suppressed for the dialog element. If it is suppressed, the event handler will not get control when the dialog element has the keyboard focus and the enabled status of one or more menu or toolbar items with a **MENU-ITEM-TYPE** of MT-EDITCUT, MT-EDITCOPY, MT-EDITPASTE, MT-EDITDELETE or MT-EDITUNDO needs to be updated by Natural during idle-time processing.

Applies to	ActiveX control, list box control, list view control, tree view control.
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

203 SUPPRESS-CLICK-EVENT

Determines whether or not the **Click Event** will be suppressed for the dialog element. If it is suppressed, the event handler will not get control when the end user makes a selection.

Applies to	Bitmap control, canvas control, column specification control, list box control, list view column, list view control, OLE container control, radio button control, signal, status bar pane, table control, timer, toggle button control, tree view control.
Data Type	I4
Default Value	0 (NOT-SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

204 SUPPRESS-CLOSE-EVENT

Determines whether or not the **Close Event** will be suppressed for the dialog element. If it is suppressed, the event handler will not get control when the corresponding OLE server application is about to be terminated or when the dialog bar or tool bar control is hidden by the user via the close button (both whilst floated and, for dialog bars, when in docked state).

Applies to	Dialog bar control, OLE container control, tool bar control.
Data Type	I4
Default Value	0 (NOT-SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

205 SUPPRESS-COLLAPSE-EVENT

Determines whether the Collapse Event will be suppressed for the dialog element or not. If it is suppressed, the dialog element will not get control when an attempt is made to collapse one of its items.

Applies to	Tree view control.
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

206 SUPPRESS-COMMAND-STATUS-EVENT

Determines whether or not the **Command-Status Event** will be suppressed for the dialog. If it is suppressed, the event handler will not get control when the state of the dialog's command UI (for example, menu items and tool bar items) is to be updated during idle processing.

Applies to	Dialog (window, MDI frame, MDI child).
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

207 SUPPRESS-CONTEXT-MENU-EVENT

Determines whether the **Context-Menu Event** will be suppressed for the dialog element or not. If it is suppressed, the dialog element will not get control when an attempt is made to open a context menu for it.

Applies to	ActiveX control, bitmap control, canvas control, date/time picker control, dialog (all types), dialog bar control, edit area control, input field control, list box control, list view control, progress bar control, push button control, radio button control, scroll bar control, selection box control, slider control, spin control, status bar control, tab control, table control, toggle button control, tool bar control, tree view control.
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

208 SUPPRESS-COPY-EVENT

Determines whether or not the **Copy Event** will be suppressed for the dialog element. If it is suppressed, the event handler will not get control when the dialog element has the keyboard focus and a menu or toolbar item with a **MENU-ITEM-TYPE** of MT-EDITCOPY is clicked.

Applies to	ActiveX control, list box control, list view control, tree view control.
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

209 SUPPRESS-CUT-EVENT

Determines whether or not the **Cut Event** will be suppressed for the dialog element. If it is suppressed, the event handler will not get control when the dialog element has the keyboard focus and a menu or toolbar item with a **MENU-ITEM-TYPE** of MT-EDITCUT is clicked.

Applies to	ActiveX control, list box control, list view control, tree view control.
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

210 SUPPRESS-DELETE-EVENT

Determines whether or not the **Delete Event** will be suppressed for the dialog element. If it is suppressed, the event handler will not get control when the dialog element has the keyboard focus and a menu or toolbar item with a **MENU-ITEM-TYPE** of MT-EDITDELETE is clicked.

Applies to	ActiveX control, list box control, list view control, tree view control.
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

211 SUPPRESS-DRAG-DROP-EVENT

Determines whether or not the **Drag-Drop Event** will be suppressed for the dialog or dialog element. If it is suppressed, the dialog or dialog element will not get control when a drop occurs over the dialog or dialog element during a drag-drop operation.

For OLE drag-drop operations, the status of this attribute also determines whether a drop is currently possible when the dialog or dialog element is defined as a drop target (see the **DROP-MODE** attribute) and the drag cursor enters or traverses the area occupied by the dialog or dialog element during a drag-drop operation. If it is suppressed, a "no drop" drag cursor is displayed and no drop is allowed, even if the **DROP-MODE** attribute value would otherwise allow it. Therefore, by dynamically updating the status of this attribute during a drag-enter and/or drag-leave event, you can temporarily inhibit or allow a drop depending on the current context (e.g., position within the dialog or dialog element).

Applies to	ActiveX control, bitmap control, control box control, dialog (all types), edit area control, list box control, list view control, tree view control.
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

212 SUPPRESS-DRAG-ENTER-EVENT

Determines whether or not the **Drag-Enter Event** will be suppressed for the dialog or dialog element. If it is suppressed, the dialog or dialog element will not get control when it is defined as a drop target (see the **DROP-MODE** attribute) and the drag cursor enters the area occupied by the dialog or dialog element during a drag-drop operation.

Applies to	ActiveX control, bitmap control, control box control, dialog (all types), edit area control, list box control, list view control, tree view control.
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

213 SUPPRESS-DRAG-LEAVE-EVENT

Determines whether or not the **Drag-Leave Event** will be suppressed for the dialog or dialog element. If it is suppressed, the dialog or dialog element will not get control when it is defined as a drop target (see the **DROP-MODE** attribute) and the drag cursor leaves the area occupied by the dialog or dialog element during a drag-drop operation.

Applies to	ActiveX control, bitmap control, control box control, dialog (all types), edit area control, list box control, list view control, tree view control.
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

214 SUPPRESS-DRAG-OVER-EVENT

Determines whether or not the **Drag-Over Event** will be suppressed for the dialog or dialog element. If it is suppressed, the dialog or dialog element will not get control when it is defined as a drop target (see the **DROP-MODE** attribute) and the drag cursor traverses the area occupied by the dialog or dialog element during a drag-drop operation.

Applies to	ActiveX control, bitmap control, control box control, dialog (all types), edit area control, list box control, list view control, tree view control.
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

215 SUPPRESS-END-DRAG-EVENT

Determines whether or not the **End-Drag Event** will be suppressed for the dialog element. If it is suppressed, the dialog element will not get control when a drag-drop operation is completed and the dialog element is used as the drag source.

Applies to	ActiveX control, bitmap control, list box control, list view control, tree view control.
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

216 SUPPRESS-ENTER-CELL-EVENT

Determines whethe or not the **Enter-Cell Event** will be suppressed for the dialog element. If it is suppressed, the event handler will not get control when the end user enters a cell.

Applies to	Column specification control, table control.
Data Type	I4
Default Value	0 (NOT-SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

217 SUPPRESS-EXPAND-EVENT

Determines whether the **Expand Event** will be suppressed for the dialog element or not. If it is suppressed, the dialog element will not get control when an attempt is made to expand one of its items.

Applies to	Tree view control.
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

218 SUPPRESS-ENTER-EVENT

Determines whether or not the **Enter Event** will be suppressed for the dialog or dialog element. If it is suppressed, the event handler will not get control when the dialog element gets the focus or when the dialog is activated.

Applies to	ActiveX control, column specification control, dialog (all types), edit area control, input field control, list view control, selection box control, table control, tab control tab, tree view control.
Data Type	I4
Default Value	0 (NOT-SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

219 SUPPRESS-FILL-EVENT

Determines whether or not the **Fill Event** will be suppressed for the dialog element. If it is suppressed, the event handler will not get control when the end user scrolls to the beginning or the end of the control (for example, the list box control).

Applies to	List box control, table control.
Data Type	I4
Default Value	0 (NOT-SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

220 SUPPRESS-IDLE-EVENT

Determines whether or not the **Idle Event** will be suppressed for the dialog. If it is suppressed, the event handler will not get control when an action occurs which could affect the state of the dialog's user interface (for example, when a key or mouse button is pressed or released).

Applies to	Dialog (window, MDI frame, MDI child).
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

221 SUPPRESS-LEAVE-CELL-EVENT

Determines whether or not the **Leave-Cell Event** will be suppressed for the dialog element. If it is suppressed, the event handler will not get control when the end user leaves a cell.

Applies to	Column specification control, table control.
Data Type	I4
Default Value	0 (NOT-SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

222 SUPPRESS-LEAVE-EVENT

Determines whether or not the **Leave Event** will be suppressed for the dialog or dialog element. If it is suppressed, the event handler will not get control when the dialog element loses the focus or when the dialog is deactivated.

Applies to	ActiveX control, column specification control, date/time picker control, dialog (all types), edit area control, input field control, selection box control, table control, tab control tab.
Data Type	I4
Default Value	0 (NOT-SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

223 SUPPRESS-PASTE-EVENT

Determines whether or not the **Paste Event** will be suppressed for the dialog element. If it is suppressed, the event handler will not get control when the dialog element has the keyboard focus and a menu or toolbar item with a **MENU-ITEM-TYPE** of MT-EDITPASTE is clicked.

Applies to	ActiveX control, list box control, list view control, tree view control.
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

224 SUPPRESS-SIZE-EVENT

Determines whether or not the **Size Event** will be suppressed for the dialog or dialog element. If it is suppressed, the event handler will not get control when the size of the dialog or dialog element is changed.

Applies to	Control box control, dialog (window, MDI frame, MDI child), dialog bar control.
Data Type	I4
Default Value	0 (NOT-SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

225 SUPPRESS-UNDO-EVENT

Determines whether or not the **Undo Event** will be suppressed for the dialog element. If it is suppressed, the event handler will not get control when the dialog element has the keyboard focus and a menu or toolbar item with a **MENU-ITEM-TYPE** of MT-EDITUNDO is clicked.

Applies to	ActiveX control, list box control, list view control, tree view control.
Data Type	I4
Default Value	1 (SUPPRESSED)
Possible Values	0 (NOT-SUPPRESSED) / 1 (SUPPRESSED)

Note: You can use the symbols SUPPRESSED and NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

226 TIME

Specifies the current date and time for a date and time picker (DTP) control.

Although DTP controls can store both date and time information internally, it can only display either the time component or the date component, depending on whether the control's “Time (t)” **STYLE** is set or not (respectively).

Note that the allowed range of values may be changed via the **SET-TIME-RANGE** action.

Applies to	Date/time picker control.
Data Type	T
Default Value	0 (implying that control uses the current time)
Possible Values	Any in-range time value.

Note: The default value of "0" in the above table implies a resetted time value.

227 TIMER-INTERVAL

Determines the time interval (in milliseconds) at which the click event handler of the timer will be triggered, that is, the click event handler will be triggered every *n* milliseconds.

Applies to	Timer.
Data Type	I4
Default Value	0
Possible Values	0 or any other integer value

228 TOOLBAR-HANDLE

Associates a tool bar and a dialog. Several dialogs may share one tool bar. Please note that the handle value of the tool bar must be assigned to the TOOLBAR-HANDLE attribute before you can make the tool bar visible with the **HAS-TOOLBAR** attribute.

Applies to	Dialog (window, MDI frame, MDI child).
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	NULL-HANDLE / any handle of a tool bar

229 TOOLBAR-POS

Specifies the position of the tool bar relative to the associated dialog (top, bottom, left, right).

Applies to	Dialog (window, MDI Frame).
Data Type	I4
Default Value	TB-TOP
Possible Values	TB-TOP / TB-BOTTOM / TB-LEFT / TB-RIGHT

230 TOOLTIP

Specifies a dialog element's tool tip text, which is automatically displayed in a small pop-up window when the mouse pointer hovers over the dialog element. For menu items and signals, the tool tip text is not displayed directly, but is “inherited” by any tool bar items referencing this menu item via their **SAME-AS** attribute. Otherwise, the specified tool tip text is only displayed if it is not an empty string and if the corresponding **HAS-TOOLTIP** attribute is set to TRUE, as follows. Item tooltips (e.g., for status bar panes, tab control tabs, tool bar items, etc.) are only displayed if the **HAS-TOOLTIP** attribute of the parent control is set to TRUE. Control or dialog tooltips (e.g., for status bar controls, tab controls, tool bar controls, etc.) are only displayed if the **HAS-TOOLTIP** attribute of the dialog is set to TRUE.

Applies to	Date/time picker control, list view control, list view item, menu item, progress bar control, signal, slider control, spin control, status bar pane, tab control tab, tool bar item.tree view control, tree view item.
Data Type	A253
Default Value	Empty string
Possible Values	Empty string / any text string

231 TYPE

Determines the type of dialog element to be created next. You can also use this attribute to query the type of an existing dialog element. You must use the TYPE value as a parameter in the PROCESS GUI statement action ADD. This will determine which type of dialog element is to be created.

Applies to	Date/time picker control, dialogs (all types) and all dialog elements, image control, image list control, list view column, list view control, list view item, progress bar control, slider control, spin control, tree view control, tree view item.
Data Type	I4
Default Value	undefined

Possible Values

BITMAP (0)	CANVAS (24)
COLUMNSPECIFICATION (29)	EDITAREA (1)
CONTEXTMENU (32)	CONTROLBOX (33)
DATETIMEPICKER (50	DIALOGBAR (38)
FONT (2)	GRAPHICTEXT (27)
GROUPFRAME (3)	IMAGE (44)
IMAGELIST (43)	INPUTFIELD (4)
LINE (25)	LISTBOX (5)
LISTBOXITEM (6)	LISTVIEW (45)
MDICHILD (19)	MDIFRAME (18)
MDIPLUGIN (42)	MENUBAR (7)
MENUITEM (8)	OCXCONTAINER (31)
OLECONTAINER (30)	PROGRESSBAR (53)

PUSHBUTTON (9)	RADIOBUTTON (10)
RECTANGLE (26)	SCROLLBAR (23)
SELECTIONBOX (11)	SELECTIONBOXITEM (12)
SIGNAL (36)	SLIDER (52)
SPINCTRL (51)	STATUSBARCTRL (35)
STATUSBARPANE (37)	SUBMENU (20)
TABCTRL (40)	TABCTRLTAB (41)
TABLE (28)	TEXTCONSTANT (13)
TIMER (22)	TOGGLEBUTTON (14)
TOOLBAR (15)	TOOLBARCTRL (34)
TOOLBARITEM (16)	TREVIEW (48)
TREEVIEWITEM (49)	WALLPAPER (39)
WINDOW (17)	

Examples of usage of the PROCESS GUI statement action ADD:

```
 PROCESS GUI ACTION ADD WITH #LISTBOX LISTBOXITEM #ITEM PROCESS GUI ACTION ADD WITH ↩
PARAMETERS
 PARENT = #LISTBOX STRING = 'Test' TYPE = LISTBOXITEM HANDLE-VARIABLE = #ITEM ↩
END-PARAMETERS
```

Note: The text representation can be substituted for the integer representation because the local data area NGULKEY1 (supplied in library SYSTEM) is automatically included in the event handler code.

232 VARIABLE

Links a Natural variable such as EMP.PERSONNEL-ID to the dialog element. This attribute must always be used with the **LINKED** attribute. As an alternative to using the LINKED and the VARIABLE attributes, you can also specify “Linked Variable” in a “Source” dialog box of an attributes window.

Applies to	Input field control, selection box control.
Data Type	Any Natural data type
Default Value	No variable associated
Possible Values	Any Natural variable

233 VERSION

Indicates the Natural version with which the dialog was saved last; you can query it.

Applies to	Dialog (all types).
Data Type	A253
Default Value	Empty string.
Possible Values	Empty string / Natural version information in encoded string format MMNNSSPP, where MM = major version, NN = minor version, SS = SM level and PP = patch level.

234 VERT-SCROLLABLE

Determines whether the dialog or dialog element has a vertical scroll bar.

Note: This attribute can be dynamically enabled, but not dynamically disabled.

Applies to	Dialog (all types), edit area control, table control.
Data Type	BOOLEAN
Default Value	TRUE
Possible Values	TRUE / FALSE

235 VIEW-MODE

Specifies the type of visual representation used by a list view control (e.g., icon view or column-based report view).

Note that the item positions are not automatically retained on switching between view modes.

For more information, please refer to the article Working with List View Controls.

Applies to	List view control.
Data Type	I4
Default Value	VM-ICON (0)
Possible Values	VM-ICON (0) = Large icon view VM-SMALLICON (1) = Small icon view VM-LIST (2) = List view VM-REPORT (3) = Report view

Note: The symbolic view mode constants listed above are defined in the local data area NGULKEY1, which is automatically used by all dialogs created within the dialog editor.

236 VISIBLE

If this attribute is set, the dialog element is visible. You can use this attribute to hide dialog elements by setting it to FALSE, if, for example, a logical condition is fulfilled and displaying the dialog or dialog element does not make sense under this condition.

Applies to	ActiveX control, bitmap control, canvas control, column specification control, control box control, date/time picker control, dialog (window, MDI frame, MDI child), dialog bar control, edit area control, graphic text control, group frame control, input field control, list box control, list view column, list view control, line control, menu bar, OLE container control, progress bar control, push button control, radio button control, rectangle control, scroll bar control, selection box control, signal, slider control, spin control, submenu control, status bar control, status bar pane, tab control, tab control tab, table control, text constant control, tool bar control, toggle button control, tree view control, wallpaper control.
Data Type	BOOLEAN
Default Value	TRUE
Possible Values	TRUE / FALSE

237 WALLPAPER

Specifies the handle of the wallpaper control (if any) that is used to paint the background of the dialog or dialog element. The background is automatically redrawn if the wallpaper control used is changed or modified. If the wallpaper control is deleted, the WALLPAPER attribute of the dialog or dialog element is automatically reset to NULL-HANDLE.

Any parts of the dialog or dialog element's background that are not covered by the wallpaper (if any) are painted using the color specified by the **BACKGROUND-COLOUR-NAME** or **BACKGROUND-COLOUR-VALUE** attributes. A value of NULL-HANDLE for the wallpaper attribute indicates that no wallpaper should be used.

Note: Wallpapers are not available for list view controls on Windows 2000.

Applies to	Control box control, dialog (all types), dialog bar control, list view control, tab control.
Data Type	HANDLE
Default Value	NULL-HANDLE
Possible Values	Handle of existing wallpaper control or NULL-HANDLE.

238 ZOOM-FACTOR

Magnifies or reduces the default representation of an OLE server object that has become visible in an OLE container control.

If you set this attribute to the value "0", the server application's object is scaled so that it fits into the OLE container control's rectangle. To avoid a distorted display, the OLE container control's rectangle should have a width/height ratio similar to that of the server object.

Applies to	OLE container control.
Data Type	I4
Default Value	100
Possible Values	0 to 500

III Events

This part covers the following topics:

Activate Event
After-Any Event
After-Edit Event
After-Open Event
Before-Any Event
Before-Edit Event
Before-Open Event
Begin-Drag Event
Change Event
Check Event
Click Event
Client-Size Event
Clipboard-Status Event
Close Event
Collapse Event
Command-Status Event
Context-Menu Event
Copy Event
Cut Event
DDE-Client Event
DDE-Server Event
Default Event
Delete Event
Delete-Row Event
Double-Click Event

Drag-Drop Event
Drag-Enter Event
Drag-Leave Event
Drag-Over Event
End-Drag Event
Enter-Cell Event
Enter Event
Error Event
Expand Event
Fill Event
Idle Event
Insert-Row Event
Leave-Cell Event
Leave Event
Paste Event
Size Event
Top Event
Undo Event
User-Defined Events

239 Activate Event

- Applies To 610
- Description 610

Applies To

List View Control, **OLE Container Control**, Tree View Control.

Description

This event is raised for OLE container controls when the corresponding OLE server application is about to be activated in place.

For list view controls and tree view controls, this event is raised when one or more items are activated (e.g., by double-clicking them, or by pressing the <Enter> key when a selection is active).

240 After-Any Event

- Applies To 612
- Description 612

Applies To

Dialog.

Description

This event handler section is performed after each event in an application and is therefore specified only once. You use it, for example, to modify variables uniformly across the application, or to perform checks. When using this type of event, make sure this does not interrupt the processing of the events that are performed afterwards.

You can use the system variables *CONTROL and *EVENT to determine which dialog element has received which event.

Note: The after-any event will not occur after a close event.

241 After-Edit Event

- Applies To 614
- Description 614

Applies To

List View Control, Tree View Control.

Description

If not suppressed, this event is received by a list view or tree view control when the user has finished editing the item's label. However, the event is not received if the user aborted the editing process via pressing the <Esc> key.

The program can query the control's **ITEM** attribute to determine which list view item or tree view item caused the event to occur.

The item's **STRING** attribute is updated by Natural before this event is raised. However, the program is free to modify the **STRING** attribute within this event if it wishes to reject or alter the new label. For example, if the program wishes to reject the new label and restore the old one (for example, if the new label is invalid), it can restore the old value of the **STRING** attribute that it previously saved in the **Before-Edit Event**.

For more information, please refer to the article Label Editing in Tree View and List View Controls.

242 After-Open Event

- Applies To 616
- Description 616

Applies To

Dialog.

Description

This event handler section is performed after the dialog with all child dialog elements has been created and displayed. You can use it, for example, to create dialog elements dynamically by using the ADD action of the PROCESS GUI statement in its event handler code. You can also use it to add items to list box controls or selection-box controls with the corresponding PROCESS GUI statement action.

243 Before-Any Event

- Applies To 618
- Description 618

Applies To

Dialog.

Description

This event handler section is performed before each event in an application and is therefore specified only once. You use it, for example, to modify variables uniformly across the application, or to perform checks. When using this type of event, make sure this does not interrupt the processing of the event handlers that are performed afterwards.

244 Before-Edit Event

- Applies To 620
- Description 620

Applies To

List View Control, Tree View Control.

Description

If not suppressed, this event is received by a **MODIFIABLE** list view or tree view control whenever an attempt is made to edit an item's label, either by clicking its label whilst the item is selected, by pressing the F2 key whilst an item has the focus, or (programatically) via the **EDIT-LABEL** action.

The program can query the control's **ITEM** attribute to determine which list view item or tree view item caused the event to occur.

If the item's **MODIFIABLE** attribute is set to FALSE on returning from this event, the editing process is aborted without label editing mode being entered.

For more information, please refer to the article Label Editing in Tree View and List View Controls.

245 Before-Open Event

- Applies To 622
- Description 622

Applies To

Dialog, **Context Menu**, **Selection Box Control**, **Submenu Control**.

Description

For dialogs, this is a code section that is executed before the dialog is created and displayed. You can use it, for example, to load profile information from a database. This code section cannot be suppressed. Note that, as the dialog elements are created after this dialog event, all handle variables except #DLG$PARENT are still set to NULL-HANDLE at this time.

For context menus and submenus, the event occurs immediately before the menu is displayed, allowing menu items to be checked, disabled, etc., according to the current context. The event may be suppressed in this case.

For selection box controls, the event occurs immediately before the selection box list is opened, allowing dynamic (re-)initialisation of the list *on demand*. The event may be suppressed in this case.

246 Begin-Drag Event

- Applies To 624
- Description 624

Applies To

ActiveX Controls, **Bitmap Control**, **List Box Control**, List View Control, Tree View Control.

Description

If not suppressed, this event is received by the drag source (see the **DRAG-MODE** attribute) when a drag-drop operation is initiated (either automatically by Natural, or manually by the application via the **PERFORM-DRAG-DROP** action).

Applications must respond to this event to by calling the **SET-CLIPBOARD-DATA** action to place some data on the drag-drop clipboard, otherwise Natural implicitly cancels the drag-drop operation.

Note that the application does *not* need to call either of the **OPEN-CLIPBOARD** or **CLOSE-CLIPBOARD** actions in this case.

247 Change Event

- Applies To 626
- Description 626

Applies To

Date and Time Picker (DTP) Control, **Edit Area Control**, **Input Field Control**, **OLE Container Control**, **Scrollbar Control**, **Selection Box Control**, Slider Control, **Spin Control**, **Table Control**.

Description

This event is raised to indicate a change in the control's value.

For a **Scrollbar Control**, this event occurs when the control's thumb position (**SLIDER** attribute value) is changed by the user.

For a Slider Control, this event occurs when the control's slider position (**SLIDER** attribute value) is changed by the user.

For a **Spin Control**, this event occurs when the control's spin position (**POSITION** attribute value) is changed by the user.

For a Date and Time Picker (DTP) Control, this event occurs when the control's date or time (**TIME** attribute value) is changed by the user.

For an **OLE Container Control**, this event occurs when the object's data have been changed.

For all other control types listed above, this event occurs when the control's text **STRING** is changed by the user or (in the case of input fields and table controls) by the program.

If this event occurs in a cell of a **Table Control**, the **ROW** and **COLUMN** attributes are modified so that they contain the position of the changed cell.

If this event occurs for an **Input Field Control** or a **Selection Box Control** with a linked variable, the linked variable must be updated in the change event.

248 Check Event

- Applies To 628
- Description 628

Applies To

List View Control, Tree View Control.

Description

If not suppressed, this event is received by an list view or tree view control whenever an item is checked or unchecked.

For this event to be received, the control must have the "Check boxes (c)" **STYLE**.

The program can query the control's **ITEM** attribute to determine which list view item or tree view item caused the event to occur.

For more information, please refer to the articles Working with List View Controls and Working with Tree View Controls.

249 Click Event

- Applies To 630
- Description 630

Applies To

Bitmap Control, **List Box Control**, **List View Column**, List View Control, **Menu Item**, **OLE Container Control**, **Push Button Control**, **Radio Button Control**, **Table Control**, **Toggle Button Control**, **Tool Bar Item**, Tree View Control.

Description

Occurs whenever the end user selects the dialog element by a keyboard function or by a mouse click: a push button control may be pressed, a list box item may be selected, or a toggle button control may be checked. It is the most important event because the end user may trigger business logic with it (press a push button control to trigger a calculation or to validate data that have been entered, and so on).

Clicking a dialog element always selects the dialog element, but it does not necessarily cause an action to occur. A click event handler section is therefore only relevant if you want an action to occur.

For list view controls, a click event is raised every time the selected state of a list view item changes, regardless of whether this represents a selection or a de-selection. If necessary, these two cases can be distinguished in the click event handler by examining the **SELECTED** state of the corresponding **ITEM**.

If this event occurs in a cell of a **Table Control**, the **ROW** and **COLUMN** attributes are modified so that they contain the position of the cell that has been clicked on.

250 Client-Size Event

- Applies To 632
- Description 632

Applies To

Dialog.

Description

Occurs whenever the interior size of the dialog window changes, for example, if a tool bar control is re-docked at a different location. The dialog interior is the part of the dialog which does not include the menu bar (if any), tool bars (if any) or status-bar (if any). You can use the client-size event handler to size controls in the dialog window relative to the dialog window's size.

251 Clipboard-Status Event

- Applies To 634
- Description 634

Applies To

ActiveX Controls, **List Box Control**, List View Control, Tree View Control.

Description

If not suppressed, this event is received by an ActiveX or list box control that has the keyboard focus when the enabled/disabled status of one or more tool bar items, menu items or signals of type "Cut", "Copy", "Paste", "Delete" or "Undo" need to be updated. The event code should respond to this event by setting the status of the **SUPPRESS-CUT-EVENT**, **SUPPRESS-COPY-EVENT**, **SUPPRESS-PASTE-EVENT**, **SUPPRESS-DELETE-EVENT**, and **SUPPRESS-UNDO-EVENT** attributes. If any of these attributes are set to SUPPRESSED after returning from this event, the corresponding clipboard commands will be disabled. Otherwise, they will be enabled. This event only needs to consider those clipboard-related events for which event handlers are available. If none exist, the Clipboard-Status event can be left in its default, suppressed state.

Typically, the Cut, Copy and Delete commands should be disabled if there is no active selection. The Paste command should be made available if clipboard data is available in a recognized format, as determined by the **INQ-FORMAT-AVAILABLE** action.

Note: Natural cannot provide any default handling for this event because it is ambiguous as to what action should be taken.

252 Close Event

- Applies To 636
- Description 636

Applies To

Dialog, **Dialog Bar Control**, **OLE Container Control**, **Tool Bar Control**.

Description

Occurs for dialogs whenever the end user selects the Close option on the window's system menu to close a dialog window or whenever the dialog is closed using the CLOSE DIALOG statement. For OLE containers, the event occurs whenever an OLE container application is closed. It can be used to make the associated event handler check if all work has been saved and ask the user to save and close, exit and close, or cancel. When a dialog is closed, all child dialogs are also closed. In this case, the close event handlers of the child dialogs are not triggered. If you want to close the parent dialog and trigger the close event handlers in the child dialogs as well, you use the subprogram NGU-DIALOG-CLOSE-ALL.

In the case of dialog bar and tool bar controls, this event occurs when the user hides the bar via the close button (or via the system menu) when floated. For dialog bar controls, the event also occurs when its close button is clicked whilst the bar is docked.

Note: In either case, the Close event is *not* raised if the bar is explicitly closed or hidden by the program, or implicitly with the parent dialog.

253 Collapse Event

- Applies To 638
- Description 638

Applies To

Tree View Control.

Description

If not suppressed, this event is received by a tree view control when the user attempts to collaps a tree view item to hide its child items (if any).

The program can query the control's **ITEM** attribute to determine which tree view item is being collapsed. Before this event is raised, this item's **EXPANDED** attribute will have already been set to FALSE. However, if the program sets this attribute to TRUE within this event, the collapse does not occur.

For more information, please refer to the article Working with Tree View Controls.

254 Command-Status Event

- Applies To 640
- Description 640

Applies To

Dialog.

Description

This event occurs whenever one or more commands (signals, menu items or tool bar items without a **SAME-AS** attribute, or status bar panes) need to be updated (i.e., disabled, checked, etc.) by a dialog during idle processing or in response to an explicit **UPDATE-COMMAND-STATUS** action. The commands do not necessarily belong to the dialog receiving the command-status event if their **SHARED** attribute is set to TRUE, which causes MDI frame commands to be automatically redirected to the active MDI child dialog (if any). This is precisely the case where the command-status event is most useful, because it allows multiple instances of an MDI child dialog (or even completely different MDI child dialogs) to share the same tool bar and status bar controls without interfering with each other, as would be the case if each MDI child would attempt to update the commands directly.

The intended usage of the command-status event is to only update state variables (and not the commands themselves) when the program state changes, and then to do a bulk update of the commands according to the current values of the state variables within the command-status event handler.

Note: The command-status event handler is called frequently, and therefore should return as quickly as possible. For this reason, database access (for example) should be avoided in this event.

If there are no commands which need to be serviced by a particular dialog, no command-status event is raised, even if there is code available for it and the event is not suppressed. This is the case for an active MDI child if no commands have been marked as shared, for example.

For performance reasons, it is not possible for the application to find out which commands caused a particular command-status event to be triggered. Instead, a dialog should update all commands for which it is the current target, and which require a non-default state to be set. The default state is disabled and unchecked for signals, menu items and tool bar items and invisible for status bar panes. Any commands not explicitly enabled or disabled (or any status bar panes not explicitly shown or hidden) by the program in the command-status event will be automatically reset to the default state by the system. This allows a particular MDI child dialog, for example, to only set the status of commands it "knows" about, and to let the system implicitly reset the commands intended only for different child dialogs. Thus, if a command is introduced for a specific MDI child dialog only, the command-status events of the all other MDI child dialogs do not need to be modified. This automatic command resetting is not performed if the corresponding dialog's command-status event is suppressed.

The command-status event for a particular dialog is raised before the idle event (if any) for that dialog, in order that the effects of the idle event code can be taken into account by the command status updating process.

255 Context-Menu Event

- Applies To 644
- Description 644

Applies To

Dialog, **ActiveX Controls**, **Bitmap Control**, **Canvas Control**, Date and Time Picker (DTP) Control, **Dialog Bar Control**, **Edit Area Control**, **Input Field Control**, **List Box Control**, List View Control, **Progress Bar Control**, **Push Button Control**, **Radio Button Control**, **Scrollbar Control**, **Selection Box Control**, Slider Control, **Status Bar Control**, **Tab Control**, **Table Control**, **Toggle Button Control**, **Tool Bar Control**, Tree View Control.

Description

If not suppressed, this event is a dialog element whenever the user attempt to open a context menu for it, either via the keyboard (using a dedicated context menu key, or <Shift>+F10) or via the mouse (using the secondary mouse button).

The primary purpose of this event is to give the program the chance to set the **CONTEXT-MENU** attribute dynamically to the handle of the correct context menu to be displayed in the case where multiple candidate context menus exist. If the choice is dependent on the relative position within the control (as is typically the case), this information may be obtained via the **INQ-CLICKPOSITION** action.

For more information, please refer to the article Defining and Using Context Menus.

256 Copy Event

- Applies To 646
- Description 646

Applies To

ActiveX Controls, **List Box Control**, List View Control, Tree View Control.

Description

If not suppressed, this event is received by an ActiveX or list box control that has the keyboard focus when the user triggers a tool bar item, menu item or signal of type "Copy". The event code should respond to this event by copying the active selection (if any) to the clipboard using the **OPEN-CLIPBOARD**, **SET-CLIPBOARD-DATA** and **CLOSE-CLIPBOARD** actions.

Note: Natural cannot provide any default handling for this event because it is ambiguous as to what action should be taken.

257 Cut Event

- Applies To 648
- Description 648

Applies To

ActiveX Controls, **List Box Control**, List View Control, Tree View Control.

Description

If not suppressed, this event is received by an ActiveX or list box control that has the keyboard focus when the user triggers a tool bar item, menu item or signal of type "Cut". The event code should respond to this event by cutting the active selection (if any) to the clipboard using the **OPEN-CLIPBOARD**, **SET-CLIPBOARD-DATA** and **CLOSE-CLIPBOARD** actions.

Note: Natural cannot provide any default handling for this event because it is ambiguous as to what action should be taken.

258 DDE-Client Event

- Applies To 650
- Description 650

Applies To

Dialog.

Description

Occurs when your dialog is acting as a client of a DDE conversation and receives a DDE message from the DDE server application, for example Microsoft Excel. The field DDE-VIEW.MESSAGE (or its equivalent) contains one of the following values:

- DISCONNECT
- DATA
- NOTIFY
- TIMEOUT

259 DDE-Server Event

- Applies To 652
- Description 652

Applies To

Dialog.

Description

Occurs when your dialog is acting as a client of a DDE conversation and receives a DDE message from a DDE client application, for example Microsoft Excel. The field DDE-VIEW.MESSAGE (or its equivalent) contains one of the following values:

- CONNECT
- DISCONNECT
- REQUEST
- ADVISE
- UNADVISE
- POKE
- EXECUTE
- TIMEOUT

260 Default Event

- Applies To 654
- Description 654

Applies To

Any dynamically created dialog element.

Description

Occurs whenever a non-suppressed event occurs for which no event handler section is specified. Also occurs whenever the end user triggers an event in a dynamically created dialog element. The default event handler section can be used to DECIDE which event (value of *EVENT) occurred for which dialog element (value of *CONTROL). For each particular dialog element and event, you can then specify the individual event handler code section.

A default event occurs, for example, if you issue SEND EVENT 'HUGO', but no such 'HUGO' event has been defined. Another example is that the event is not suppressed, but the event handler section is empty.

261 Delete Event

- Applies To 656
- Description 656

Applies To

ActiveX Controls, **List Box Control**, List View Control, Tree View Control.

Description

If not suppressed, this event is received by an ActiveX or list box control that has the keyboard focus when the user triggers a tool bar item, menu item or signal of type "Delete". The event code should respond to this event by deleting the control's active selection (if any), together with any associated data.

Note: Natural cannot provide any default handling for this event because it is ambiguous as to what action should be taken.

262 Delete-Row Event

- Applies To 658
- Description 658

Applies To

Table Control.

Description

Occurs after the end user has deleted a row by selecting it and by pressing DEL or by calling "PROCESS GUI ACTION TABLE-DELETE-ROW...". The table control's **STYLE** attribute must have had the value "e" (extendable, allows end users to delete rows).

If this event occurs in a cell, the **ROW** and **COLUMN** attributes are modified so that they contain the positon of the cell.

263 Double-Click Event

- Applies To 660
- Description 660

Applies To

Bitmap Control, **Canvas Control**, **List Box Control OLE Container Control**, **Table Control**.

Description

Occurs whenever the end user double-clicks on the dialog element. For list box controls and radio button controls, the double-click event normally triggers the event handler with the business logic, whereas the click event would normally trigger an event handler with a simple display function.

The first click of a double-click triggers a click event, except for when the click event is suppressed. In the context of list box controls and radio button controls, the first click of the double-click selects a list box item or a **Radio Button Control**. The second click should be equivalent to pressing the default button in the corresponding window.

If this event occurs in a cell of a **Table Control**, the **ROW** and **COLUMN** attributes are modified so that they contain the position of the cell.

If this event occurs inside the rectangle of an **OLE Container Control**, the OLE server application is started. Double-clicking inside an **OLE Container Control** is equivalent to clicking with the right mouse button. (Clicking the right mouse button inside the **OLE Container Control** 's rectangle invokes a pop-up menu. By default, releasing the mouse button without further selection executes the first command verb of the OLE server application.)

264 Drag-Drop Event

- Applies To 662
- Description 662

Applies To

Dialog, **ActiveX Controls**, **Bitmap Control**, **Control Box Control**, **Edit Area Control**, **List Box Control**, List View Control, Tree View Control.

Description

If not suppressed, this event is received by the drop target (if any) under the drag-cursor when a drop occurs during a drag-drop operation.

Natural supports two types of drag-drop operation. The older mechanism is used exclusively for drag-drop operations between two bitmap controls within the same dialog. The newer, OLE-based, mechanism is more general and supports any of the above control types as drop targets (see the **DROP-MODE** attribute). The latter mechanism is much more powerful than the former, which has been retained only in order to maintain compatibility with earlier Natural versions. In both cases, the application typically responds to this event by first calling the **INQ-DRAG-DROP** action to find out more information about the drop (e.g., where the drop occurred within the dialog or dialog element, the type of drop operation performed, etc.).

In the case of OLE-based drag-drop, the application then calls the **GET-CLIPBOARD-DATA** action to retrive the dragged data from the drag-drop clipboard.

In the case of non-OLE drag-drop, the application should delete the source data (obtained directly via the source control handle returned by the **INQ-DRAG-DROP** action, since the drag-drop clipboard is not used in this case).

Note: For OLE drag-drop, the deletion of the source data is done in the **END-DRAG** event by the drag source instead.

265 Drag-Enter Event

- Applies To 664
- Description 664

Applies To

Dialog, **ActiveX Controls**, **Bitmap Control**, **Control Box Control**, **Edit Area Control**, **List Box Control**, List View Control, Tree View Control.

Description

If not suppressed, this event is received by the drop target (see the **DROP-MODE** attribute) under the drag cursor when the drag cursor enters the area occupied by the drop target during a drag-drop operation.

Applications typically use this event to determine whether a compatible Natural data format exists on the drag-drop clipboard via the **INQ-FORMAT-AVAILABLE** action, and to enable or disable a drop by setting the **SUPPRESS-DRAG-DROP-EVENT** attribute accordingly.

266 Drag-Leave Event

- Applies To 666
- Description 666

Applies To

Dialog, **ActiveX Controls**, **Bitmap Control**, **Control Box Control**, **Edit Area Control**, **List Box Control**, List View Control, Tree View Control.

Description

If not suppressed, this event is received by the drop target (see the **DROP-MODE** attribute) under the drag cursor when the drag cursor leaves the area occupied by the drop target during a drag-drop operation without a drop having occurred, or if the drag-drop operation was cancelled (e.g. via pressing the Escape key).

Applications should use this event to dynamically remove any drop highlighting set in the drag-over event. For example, if the drop target is an ActiveX tree view control, the selection can be restored to the originally selected node if the selection was temporarily moved in the drag-over event to highlight the current drop position.

If drop highlighting is not used (as is usually the case if the effects of the drop are not position-dependent), this event is can be left suppressed.

Note: For each drag-enter event it receives, a drop target receives either a drag-drop or drag-leave event (depending on whether a drop occurred or not), but not both. Therefore, some (or all) of the tasks performed in the drag-leave event may also need to be performed in the drag-drop event.

267 Drag-Over Event

- Applies To 668
- Description 668

Applies To

Dialog, **ActiveX Controls**, **Bitmap Control**, **Control Box Control**, **Edit Area Control**, **List Box Control**, List View Control, Tree View Control.

Description

If not suppressed, this event is received by the drop target (see the **DROP-MODE** attribute) under the drag cursor when the drag cursor traverses the area occupied by the drop target during a drag-drop operation.

Applications typically use this event to dynamically enable or disable a drop according to the position within the drop target (as returned by the **INQ-DRAG-DROP** action) by setting the **SUPPRESS-DRAG-DROP-EVENT** attribute accordingly. Alternatively, or additionally, this event can be used to highlight this drop position. For example, if the drop target is an ActiveX tree view control, the tree view node where the drop would currently occur could be marked as selected. If the effect of the drop is not position-dependent, this event can be left suppressed.

Note: This event is raised frequently during a drag-drop operation. Therefore, the application's drag-over event handler (if any) should be made as performant as possible (e.g., by performing position-independent tasks in the drag-enter event instead).

268 End-Drag Event

- Applies To 670
- Description 670

Applies To

Dialog, **ActiveX Controls**, **Bitmap Control**, **List Box Control**, List View Control, Tree View Control.

Description

If not suppressed, this event is received by the drag source (see the **DRAG-MODE** attribute) when a drag-drop operation is terminated (either successfully or unsuccessfully). The event is raised if (and only if) a **BEGIN-DRAG** event was previously raised.

Applications may determine the type of drop (if any) by inspecting the "Drop Effect" parameter returned by the **INQ-DRAG-DROP** action. If no drop occurred, this parameter is set to DM-NONE, otherwise it is set to one of the other simple drag mode constants (DM-MOVE, DM-COPY or DM-LINK) to indicate the type of drag operation that was performed. If this parameter is set to DM-MOVE, the application should delete the source data.

269 Enter-Cell Event

- Applies To 672
- Description 672

Applies To

Table Control.

Description

Occurs whenever the end user changes the focus by clicking on a cell in a **Table Control**, by using TAB to enter a cell, or by using the arrow keys to enter a cell.

If this event occurs, the **ROW** and **COLUMN** attributes are modified so that they contain the positon of the cell.

270 Enter Event

- Applies To 674
- Description 674

Applies To

Dialog, **ActiveX Controls**, **Column Specification Control**, Date and Time Picker (DTP) Control, **Edit Area Control**, **Input Field Control**, **Selection Box Control**, **Tab Control Tab**, **Table Control**.

Description

Occurs whenever the end user changes the focus by clicking on the dialog element or by using TAB.

For dialogs, this event occurs when a dialog becomes active. This happens when

- the end user activates it by a mouse click or by a keyboard operation; or
- the PROCESS GUI statement action **SET-FOCUS** is applied to this dialog.

For dialog elements, this event occurs when the dialog element receives the focus by end user action or if the **SET-FOCUS** Action is applied.

For table controls, this event occurs if the table receives the focus or if the end user enters a new *row*. If the end user enters a new *cell*, the enter-cell event occurs.

If this event occurs in a cell of a **Table Control**, the **ROW** and **COLUMN** attributes are modified so that they contain the position of the cell.

When an end user leaves an MDI application, for example, by clicking into another application outside Natural, and clicks into the MDI application again, an enter event only occurs for the MDI Frame dialog, not for the MDI Child dialogs.

271 Error Event

- Applies To 676
- Description 676

Applies To

Dialog.

Description

This event handler section is performed whenever a runtime error occurs while a dialog is active. You can specify event handler code to be executed whenever this error occurs. If no error event handler code is specified, Natural will terminate with an error message and all dialogs will be closed. If an error event handler code section was specified, the dialog will remain open and handle the error.

You can use the system variables *CONTROL and *EVENT to determine the event handler in which the error occurred. If you want the dialog to continue after processing the error, code an ESCAPE ROUTINE statement at the end of the error event handler.

272 Expand Event

- Applies To 678
- Description 678

Applies To

Tree View Control.

Description

If not suppressed, this event is received by a tree view control when the user attempts to expand a tree view item to show its child items (if any).

The program can query the control's **ITEM** attribute to determine which tree view item is being expanded. Before this event is raised, this item's **EXPANDED** attribute will have already been set to TRUE. However, if the program resets this attribute to FALSE within this event, the expansion does not occur.

At the time this event is raised, the child items have not yet been shown, allowing the child items to be created or destroyed dynamically without any transition effects being visible to the user.

For more information, please refer to the article Working with Tree View Controls.

273 Fill Event

- Applies To 680
- Description 680

Applies To

List Box Control, **Table Control**.

Description

Occurs for list box controls whenever the end user scrolls to the end of a list box control that has scroll bars. Occurs for table controls whenever the end user navigates to the last row in the **Table Control** or scrolls to the end of the table control with the scroll bar. If all items in the **List Box Control** are visible without scrolling, there is no scroll bar and therefore no fill event will occur. In this event, you would usually append additional items to the end of the list box control with the PROCESS GUI statement action ADD-ITEMS.

If this event occurs in a cell of a **Table Control**, the **ROW** and **COLUMN** attributes are modified so that they contain the position of the cell.

274 Idle Event

- Applies To 682
- Description 682

Applies To

Dialog.

Description

Occurs during idle-time processing in situations where the user interface may need to be updated, which is after each time a key or mouse button is pressed or released.

Note: Mere mouse movement does not normally cause idle events to be raised, because this would be too slow. An exception is when the mouse pointer is moved away from a tool bar, in order to allow the program to update the status-bar text.

In addition to allowing the program to update the status-bar with an idle message (e.g., "Ready"), the idle event may also be used for monitoring user interface changes. For example, the idle event of an MDI frame dialog can query the active MDI child dialog and show or hide individual tool bars appropriately if necessary.

Note: The idle event handler is called frequently, and therefore should return as quickly as possible. For this reason, database access (for example) should be avoided in this event.

275 Insert-Row Event

- Applies To 684
- Description 684

Applies To

Table Control.

Description

Occurs after the end user has inserted a row by pressing INS or by calling "PROCESS GUI ACTION TABLE-INSERT-ROW...". The table control's **STYLE** attribute must have had the value "e" (extendable, allows end users to insert rows).

If this event occurs in a cell, the **ROW** and **COLUMN** attributes are modified so that they contain the position of the cell.

276 Leave-Cell Event

- Applies To 686
- Description 686

Applies To

Table Control.

Description

Occurs whenever the end user moves the focus away from a cell (for example, by clicking on another cell in the table control or by using TAB to enter another cell).

If this event occurs in a cell, the **ROW** and **COLUMN** attributes are modified so that they contain the position of the cell.

277 Leave Event

- Applies To 688
- Description 688

Applies To

Dialog, **ActiveX Controls**, **Column Specification Control**, Date and Time Picker (DTP) Control, **Edit Area Control**, **Input Field Control**, **Selection Box Control**, **Tab Control Tab**, **Table Control**.

Description

Occurs whenever the dialog or dialog element loses the focus. You can use the leave event handler to validate field entries or to dynamically modify other dialog elements whose status depends on that of the dialog element just left. Does not occur on closing a dialog.

For table controls, this event occurs if the end user leaves a *row* or if the table loses the focus. If the end user leaves a *cell*, the leave-cell event occurs.

If this event occurs in a cell of a **Table Control**, the **ROW** and **COLUMN** attributes are modified so that they contain the position of the cell.

Before this event handler code is processed, end user input in (linked variable) input-field controls and selection-box controls is checked against any **EDIT-MASK** value or applicable Natural data type. However, in the following situations the leave event is *not* triggered after end user input was checked and a linked variable was updated:

- An **Input Field Control** or a **Selection Box Control** has the focus and the end user clicks a menu item; or
- the end user presses ENTER, triggering the default button; or
- the end user presses ENTER, triggering the OK button; or
- the end user presses an accelerator key.

Note: It is not recommended to execute the PROCESS GUI statement action SET-FOCUS from the leave event because the leave event implies that the focus is to be set to yet another part of the application.

278 Paste Event

- Applies To 690
- Description 690

Applies To

ActiveX Controls, **List Box Control**, List View Control, Tree View Control.

Description

If not suppressed, this event is received by an ActiveX or list box control that has the keyboard focus when the user triggers a tool bar item, menu item or signal of type "Paste". The event code should respond to this event by pasting the contents of the clipboard into the control at the current position (if any) using the **GET-CLIPBOARD-DATA** action.

Note: Natural cannot provide any default handling for this event because it is ambiguous as to what action should be taken.

279 Size Event

- Applies To 692
- Description 692

Applies To

Dialog (except MDI Plug-in), **Control Box Control**, **Dialog Bar Control** .

Description

Occurs whenever the exterior size of the dialog or dialog element changes (for example, if the window is minimized or maximized). You can use the size event handler to resize and or re-position any child dialog elements accordingly.

Note: To be notified of a change in the dialog's interior size (if different), use the **Client-Size Event** instead.

280 Top Event

- Applies To 694
- Description 694

Applies To

Table Control.

Description

Occurs whenever the end user navigates to the top of the table control (either by using the scroll bar or by entering a cell of the top row).

If this event occurs in a cell, the **ROW** and **COLUMN** attributes are modified so that they contain the positon of the cell.

281 Undo Event

- Applies To 696
- Description 696

Applies To

ActiveX Controls, **List Box Control**, List View Control, Tree View Control.

Description

If not suppressed, this event is received by an ActiveX or list box control that has the keyboard focus when the user triggers a tool bar item, menu item or signal of type "Undo". The event code should respond to this event by undoing the effect of the last action applied to the control, if possible.

Note: Natural cannot provide any default handling for this event because it is ambiguous as to what action should be taken.

282 User-Defined Events

- Apply To 698
- Description 698

Apply To

Dialog.

Description

User-defined events occur whenever you have specified a SEND EVENT statement with the name of a previously undefined event in an event handler code of a dialog.

Note: The user event must be defined in another dialog than the one where it occurs. For more details and examples of how to cause an action to occur in another dialog, see the section Triggering User-Defined Events in Event-driven Programming Techniques documentation.

You can define user events for dialogs by pressing the "New" button in the "Events..." dialog box of the dialog's attributes window. You can then enter any name of your newly-defined user event and specify the corresponding event section. It is recommend that you use "#" as the first letter to distinguish user-defined events from previously defined events.

If no event handler is defined for an event generated by the SEND EVENT statement, the default event handler is executed with the value of the system variable *EVENT as the user event name.

IV PROCESS GUI Statement Actions

This part covers the following topics:

General Information
ADD Action
ADD-ITEMS Action
ADD-ITEMS-EX Action
ARRANGE Action
BEEP Action
CALL-DIALOG Action
CLEAR Action
CLEAR-TICKS Action
CLOSE-CLIPBOARD Action
DELETE-CHILDREN Action
DELETE-WINDOW Action
DELETE Action
DELETE-SUBITEM-DATA Action
EDIT-GET-LINE-NUMBER Action
EDIT-LABEL Action
EDIT-LINE-DELETE Action
EDIT-LINE-GET-SELECTION Action
EDIT-LINE-GET-TEXT Action
EDIT-LINE-INSERT Action
EDIT-LINE-SET-SELECTION Action
EDIT-LINE-SET-TEXT Action
ENUM-CHILDREN Action
ENUM-CLIENT-KEYS Action
ENSURE-VISIBLE Action

GET-CLIENT-VALUE Action
GET-CLIPBOARD-DATA Action
GET-FIRST-WINDOW Action
GET-FOCUS Action
GET-MESSAGE-TEXT Action
GET-NEXT-WINDOW Action
GET-TEXT Action
GET-SUBITEM-DATA Action
HELP Action
HOURGLASS-REMOVE Action
HOURGLASS-STACK Action
HOURGLASS-UNSTACK Action
INPUT-COPY-SELECTION Action
INPUT-CUT-SELECTION Action
INPUT-DELETE-SELECTION Action
INPUT-GET-LINE-LENGTH Action
INPUT-GET-SELECTION Action
INPUT-GET-TEXT Action
INPUT-PASTE Action
INPUT-SET-SELECTION Action
INPUT-SET-TEXT Action
INPUT-UNDO Action
INQ-CLICKPOSITION Action
INQ-DRAG-DROP Action
INQ-FORMAT-AVAILABLE Action
INQ-INNER-RECT Action
INQ-ITEM-BY-POSITION Action
INQ-NON-CLIENT-METRICS Action
LOAD-LAYOUT Action
MOVE-NAVIGATION-ITEMS Action
MESSAGE-BOX Action
OLE-ACTIVATE
OLE-DEACTIVATE
OLE-GET-DATA
OLE-INSERT-OBJECT
OLE-READ-FROM-FILE
OLE-SAVE-TO-FILE
OLE-SET-DATA

OPEN-CLIPBOARD Action
PERFORM-DRAG-DROP Action
PICK-FILENAME Action
PLAY-SOUND Action
PROCESS-EVENTS Action
RECALC-LAYOUT Action
REFRESH-LINKS Action
RESET-ATTRIBUTES Action
SAVE-LAYOUT Action
SET-ACCELERATION Action
SET-AUX-COLOR Action
SET-AUX-FONT Action
SET-CLIENT-VALUE Action
SET-CLIPBOARD-DATA Action
SET-FOCUS Action
SET-SUBITEM-DATA Action
SET-TABS Action
SET-TEXT Action
SET-TICKS Action
SET-TIME-RANGE Action
SHOW-CONTEXT-MENU Action
SORT-ITEMS Action
SYSTEM-GET-NATIVE-HANDLE Action
SYSTEM-PRINTER-SETUP Action
TABLE-DELETE-ROW Action
TABLE-FIND-FIELD Action
TABLE-GET-SELECTION Action
TABLE-INQUIRE-CELL Action
TABLE-INQUIRE-ROW Action
TABLE-INSERT-ROW Action
TABLE-REFRESH Action
TABLE-SET-SELECTION Action
TEXT-GET-EXTENT Action
UPDATE-COMMAND-STATUS Action
VALIDATE Action

283 General Information

The PROCESS GUI statement actions execute procedures from within the PROCESS GUI statement. They are flexible in parameter handling: it is sufficient to supply type-compatible parameters with them (for example, if A5 is the exact parameter type, supplying an I1 parameter will be accepted).

For more information about the PROCESS GUI statement, refer to *Executing Standardized Procedures* in the section *Event-Driven Programming Techniques*.

To use the PROCESS GUI statement actions more comfortably, the local data area NGULKEY1 is automatically included in the list of local data areas used by any new dialog.

NGULKEY1 contains reserved symbols to be used in any event handler code. This enables you to use meaningful names as parameters in a PROCESS GUI statement. It also enables you to refer to certain attribute values by the more meaningful texts rather than by the integer values.

Notes:

1. The "Response" parameters in the PROCESS GUI statement actions are optional.
2. ActiveX controls use the PROCESS GUI statement in a slightly different way. You use the statement to execute the ActiveX control's own methods and to access parameterized properties. For a description on how to use the PROCESS GUI statement for ActiveX controls, see Working with ActiveX Controls.

284 ADD Action

- Description 706
- Parameters for the ADD WITH option 706

Description

Creates a single specified dialog element dynamically. See also *Creating/Deleting Dialog Elements Dynamically* in the section *Event-Driven Programming Techniques* documentation. This action is most frequently used to add items to a **List Box Control** or to a **Selection Box Control** or to add column specifications to a table. It can also be used to create all kinds of dialog elements dynamically.

There are two syntax options of this action:

PROCESS GUI ACTION ADD WITH...

This option has the parameters as listed below. Other attributes of the newly created dialog element have to be set in the global attributes list before the PROCESS GUI statement.

PROCESS GUI ACTION ADD WITH PARAMETERS... END-PARAMETERS

This option accepts a list of attribute assignments, one for each attribute that is to be specified for the newly created dialog element. If you use this option, the global attribute list is not used or affected. For all attributes that are not explicitly specified, the default value is taken.

Parameters for the ADD WITH option

Name/Data Type	Explanation
HANDLE OF GUI	Input The handle of the parent dialog element.
Type (I4)	Input The type of dialog element to be created.
HANDLE OF GUI	Output The handle of the newly created dialog element.
Response (I4)	Output Natural error (if applicable).

Example 1 (option 1):

```
DEFINE DATA LOCAL
  1 #NEW1 HANDLE OF INPUTFIELD
  END-DEFINE
  ...
  #NEW1.STRING:= 'NEW1'
  #NEW1.RECTANGLE-X:= 24
  #NEW1.RECTANGLE-Y:= 30
  #NEW1.RECTANGLE-W:= 176
  #NEW1.RECTANGLE-H:= 28
  #NEW1.ENABLED:= TRUE
  #NEW1.VISIBLE:= TRUE
  PROCESS GUI ACTION ADD WITH #DLG$WINDOW INPUTFIELD #NEW1
```

Example 2 (option 2):

```
DEFINE DATA LOCAL
  1 #NEW2 HANDLE OF INPUTFIELD
  END-DEFINE
  ...
  PROCESS GUI ACTION ADD WITH PARAMETERS
     HANDLE-VARIABLE = #NEW2
     TYPE = INPUTFIELD
     STRING = 'NEW2'
     RECTANGLE-X = 24
     RECTANGLE-Y = 30
     RECTANGLE-W = 176
     RECTANGLE-H = 28
     ENABLED = TRUE
     VISIBLE = TRUE
     PARENT = #DLG$WINDOW
  END-PARAMETERS ↩
```

If you insert a new dialog element dynamically by using the ADD action, you determine its position in the navigation sequence by creating the dialog element and setting the **SUCCESSOR** attribute to the handle value of its successor.

Example:

```
  /* Insert Input Field Control #NEW1 before push button control #PB-1
  /* Be careful not to trigger the PROCESS GUI statement action from a push
  /* button control named #PB-1 because you are already defining it
  DEFINE DATA LOCAL
  1 #NEW1 HANDLE OF INPUTFIELD
  1 #PB-1 HANDLE OF PUSHBUTTON
  END-DEFINE
  ...
  PROCESS GUI ACTION ADD WITH PARAMETERS
     PARENT = #DLG$WINDOW
```

```
      HANDLE-VARIABLE = #NEW1
      TYPE = INPUTFIELD
      SUCCESSOR = #PB-1
      ...
   END-PARAMETERS
↩
```

285 ADD-ITEMS Action

- Description 710
- Parameters 710

Description

Adds several list box items in a **List Box Control** at once. Does the same for selection box items in a **Selection Box Control** and column specifications in a **Table Control**.

Parameters

Name/Data Type	Explanation
HANDLE OF dialog element	Input Specifies a dialog element.
Number of Items (I4)	Input You can add any number of items.
Item (list of A253-compatible values or one-dimensional array)	Input Item string(s).
Response (I4)	Output Natural error (if applicable).

Example:

```
DEFINE DATA LOCAL
   1 #AMOUNT (I4)
   1 #ITEM (A20/1:5)
   1 #RESPONSE (I4)
   END-DEFINE
   ...
   #AMOUNT:= 5
   #ITEM(1):= 'Berlin'
   #ITEM(2):= 'Paris'
   #ITEM(3):= 'London'
   #ITEM(4):= 'Milan'
   #ITEM(5):= 'Madrid'
   PROCESS GUI ACTION ADD-ITEMS WITH #LB-1 #AMOUNT #ITEM(*) GIVING #RESPONSE ↩
```

286 ADD-ITEMS-EX Action

- Description 712
- Parameters 712

Description

Adds list box items in a **List Box Control** and selection box items in a **Selection Box Control**. Associates data to the items either as CLIENT-KEY/CLIENT-VALUE pairs or as CLIENT-DATA values. The item strings and the corresponding data can be specified either as single values or as one-dimensional arrays. Both strings and data must be provided in the same way. If the parameter "Client key" is blank, each data value is entered as the **CLIENT-DATA** of the corresponding item. If it is not blank, it is entered as the **CLIENT-VALUE** with the corresponding **CLIENT-KEY**.

Parameters

Name/Data Type	Explanation
HANDLE OF dialog element	Input Specifies a dialog element.
Number of Items (I4)	Input
Client key (A253)	Input CLIENT-KEY attribute to be used for each value in the list of values.
String list (A253)	Input A number of alphanumeric variables or constants or an array specification.These strings are entered in the STRING attribute of each item. You may only use a one-dimensional array in the array specification. Using part of a higher-level array causes an error.
Value list (A253 or I4)	Input A number of alphanumeric variables or constants or an array specification.These are interpreted as the **CLIENT-VALUE** of the given **CLIENT-KEY**. If the CLIENT-KEY value is blank, the value list is entered in the **CLIENT-DATA**.
Response (I4)	Output Natural error (if applicable).

Example:

```
/* Definitions in the dialog's local data area:
1 #NUMBER (N4)
1 #CITY (A20/1:2)
1 #CODE (I4/1:2)
1 #KEY (A20)
...
/* Event handler code:
#NUMBER:= 2
#CITY(1):= 'Berlin'
#CODE(1):= 1015
#CITY(2):= 'Munich'
#CODE(2):= 8053
#KEY:= 'Code'
PROCESS GUI ACTION ADD-ITEMS-EX
WITH #LB-1 #NUMBER #KEY #CITY(1) #CITY(2) #CODE(1) #CODE(2) GIVING #RESPONSE
/*  Another possible WITH clause (same result)
WITH #LB-1 #NUMBER #KEY #CITY(1:2) #CODE(1:2) GIVING #RESPONSE
```

287 ARRANGE Action

- Description 716
- Parameters 716

Description

Re-arranges the items belonging to the specified list view control according to a logical grid, if the list view is currently in one of the icon view modes.

The size of the logical grid can be modified from the default via updating the list view control's **SPACING-X** and **SPACING-Y** attribute values.

Parameters

Name/Data Type	Explanation
Control (HANDLE OF LISTVIEW)	Input Handle of list view control whose items are to be re-arranged.
Snap (L)	Input (optional parameter) Snap to grid. FALSE=re-arrange to contiguous logical grid positions according to orientation determined by the presence or absence of the list view control's "Align vertically (v)" STYLE attribute flag (default); TRUE=move items to nearest aligned logical grid position.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION ARRANGE WITH #LV-1 GIVING *ERROR ↩
```

288 BEEP Action

- Description 718
- Parameters 718

Description

Issues a "beep" sound.

Parameters

Name/Data Type	Explanation
Number (1 - 3)	Input Type of the beep to be issued. "1" denotes a warning beep, "2" denotes a note beep, "3" denotes an error beep.

Note: The "Number" parameter is optional. If you omit this parameter, the value "1" (warning beep) applies by default.

Example:

```
PROCESS GUI ACTION BEEP ↩
```

289 CALL-DIALOG Action

- Description 720
- Parameters 720

Description

Triggers an event for the specified dialog or dialog element. The event may optionally be processed synchronously or asynchronously. In the latter case, the event will not be processed until the currently executing event and any already queued events (arising from previous calls to this action) have completed, or until a call to the **PROCESS-EVENTS** action is made. Note that, in contrast to the SEND EVENT statement, it is not possible to specify event parameters.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of the dialog to be invoked to process the event.
HANDLE OF GUI	Input Handle of the target dialog element, or NULL-HANDLE for dialog events. The system variable *CONTROL will be set to this value.
Event (A253)	Input The event name. The system variable *EVENT will be set to this value.
Synchronous mode (L)	Input (optional parameter) TRUE (=default) if the event should be processed immediately, or FALSE if the event should be queued.
Response (I4)	Output Natural error (if applicable).

Example:

```
/* trigger a click event for push button #pb-1
 PROCESS GUI ACTION CALL-DIALOG WITH #DLG$WINDOW #PB-1 'CLICK'
   GIVING #RESPONSE

/* trigger an asynchronous close event for parent dialog
PROCESS GUI ACTION CALL-DIALOG WITH #DLG$PARENT NULL-HANDLE
  'CLOSE' FALSE GIVING #RESPONSE
↩
```

290 CLEAR Action

- Description 722
- Parameters 722

Description

Clears the contents of a dialog element. Any items (e.g., **List View Items**, an **List Box Items** or a **Selection Box Items**) belonging to the dialog element are deleted.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an **Edit Area Control**, **Input Field Control**, **List Box Control**,**List View Control**, **OLE Container Control**, **Selection Box Control**,or a **Table Control**.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION CLEAR WITH #EA-1 GIVING #RESPONSE ↩
```

291 CLEAR-TICKS Action

- Description 724
- Parameters 724

Description

Clears all manual tick marks (if any) for a slider control. Manual tick marks are those added via the **SET-TICKS** action

This action cannot be used if the control has the "Auto ticks (a)" STYLE.

Note that this action does not clear the tick marks at the beginning or end of the slider's range, which are automatically created by the control.

Parameters

Name/Data Type	Explanation
HANDLE OF SLIDER	Input Handle of slider for which the tick marks are to be cleared.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION CLEAR-TICKS WITH #SLIDER-1 GIVING *ERROR ↩
```

292 CLOSE-CLIPBOARD Action

- Description 726
- Parameters 726

Description

Closes the local clipboard and places the contents of the local clipboard on the Windows clipboard, thus making the data available for pasting into other applications.

After calling this action, no further calls to **SET-CLIPBOARD-DATA** may be made until the local clipboard is re-opened via the **OPEN-CLIPBOARD** action.

Note that if data for a drag-drop operation is being prepared in the **BEGIN-DRAG** event, this call is not necessary, because you usually don't need or want to also make the transferred data available for pasting. Drag and drop can (and normally does) work directly with the local clipboard. See the **GET-CLIPBOARD-DATA** action for more information on this topic.

Parameters

Name/Data Type	Explanation
Response (I4)	Output Natural error (if applicable).

Example:

```
 /* Empty the Windows clipboard
PROCESS GUI ACTION OPEN-CLIPBOARD GIVING #RESPONSE
PROCESS GUI ACTION CLOSE-CLIPBOARD GIVING #RESPONSE

 ↩
```

293 DELETE-CHILDREN Action

- Description 728
- Parameters 728

Description

Deletes all children of a given dialog or dialog element dynamically. You use this action, for example, to delete all items in a **List Box Control** before filling the list box again (using the **ADD-ITEMS** Action).

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of the dialog or dialog element whose children are to be deleted.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION DELETE-CHILDREN WITH #LB-1 GIVING #RESPONSE ↩
```

294 DELETE-WINDOW Action

- Description 730
- Parameters 730

Description

For the specified dialog, and its child dialogs (if any), this action destroys the dialog's window (if it exists) and causes the dialog object to be unloaded as soon as all currently executing events for that dialog have been completed. Note that the dialog does not receive a **CLOSE** event.

Note: You do not normally need to call this action explicitly, as the dialog implicitly generates a call to this action that is executed during CLOSE event handling. Therefore, it is recommended that the CLOSE DIALOG statement statement be used in preference to this action.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of the dialog's window, or NULL-HANDLE. In the latter case, the dialog specified by the system variable *DIALOG-ID is unloaded.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION DELETE-WINDOW WITH #DLG$WINDOW GIVING #RESPONSE ↩
```

295 DELETE Action

- Description 732
- Parameters 732

Description

Deletes a specified dialog element. You can use this action, for example, to delete items from a **List Box Control** or to remove dialog elements from a dialog.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Any dialog element.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION DELETE WITH #PB-1 GIVING #RESPONSE ↩
```

296 DELETE-SUBITEM-DATA Action

- Description 734
- Parameters 734

Description

Deletes the ("subitem") data for the specified list view item for the specified list view column(s).

The list view item handle may be specified as NULL-HANDLE, in which case the default data for the specified column(s) will be deleted. The default data is the data returned by the **GET-SUBITEM-DATA** action in the case where no subitem data is present.

The column handle parameter may be repeated, in order to allow the data for multiple columns to be deleted in a single action.

Note that this action cannot be used to delete the data for the primary column (i.e., the column used to display the list view item labels).

For more information, please refer to the article Working with List View Controls.

Parameters

Name/Data Type	Explanation
Item (HANDLE OF GUI)	Input Handle of list view item whose data is to be deleted, or NULL-HANDLE if the default data for the specified list view column(s) should be deleted.
Column (HANDLE OF GUI)	Input Handle of list view column whose data is to be deleted. Cannot be specified as NULL-HANDLE.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION DELETE-SUBITEM-DATA WITH #LVITEM-1 #LVCOL-2 #LVCOL-3 #LVCOL-4 ↩
GIVING #RESPONSE ↩
```

297 EDIT-GET-LINE-NUMBER Action

- Description 736
- Parameters 736

Description

Retrieves the number of lines in the **Edit Area Control**. This is also true if the **STYLE** attribute value has been set to value "w" (wordwrap).

Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA	Input Specifies an **Edit Area Control**.
Line number (I4)	Output The number of lines.
Response (I4)	Output Natural error (if applicable).

Example:

```
  PROCESS GUI ACTION
   EDIT-GET-LINE-NUMBER WITH #EA-1
   #LINE-NUMBER GIVING
   #RESPONSE
   #IF-1.STRING:= #LINE-NUMBER
↩
```

298 EDIT-LABEL Action

- Description 738
- Parameters 738

Description

Initiates label editing for a list view item or tree view item.

The list view or tree view control is automatically scrolled (and/or expanded, in the case of tree view controls), if necessary, in order to bring the specified item into view.

Note that the standard label editing process, as detailed in the article Label Editing in Tree View and List View Controls, is initiated by this action.

Parameters

Name/Data Type	Explanation
Item (HANDLE OF LISTVIEWITEM or HANDLE OF TREEVIEWITEM)	Input Handle of list view item or tree view item to be edited.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION EDIT-LABEL WITH #LVITEM-1 GIVING *ERROR ↩
```

299 EDIT-LINE-DELETE Action

- Description 740
- Parameters 740

Description

Deletes a line in the **Edit Area Control**.

Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA	Input Specifies an **Edit Area Control**.
Line number (I4)	Output The line to be deleted. If you specify "0", the last line will be deleted.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION EDIT-LINE-DELETE WITH #EA-1 #LINE-NUMBER GIVING #RESPONSE
```

300 EDIT-LINE-GET-SELECTION Action

- Description 742
- Parameters 742

Description

Retrieves the extent of the selected area (lines and columns) in an **Edit Area Control**.

Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA	Input Specifies an **Edit Area Control**.
Line from (I4)	Output Selection starts from this line onwards.
Column from (I4)	Output Selection starts from this column onwards.
Line to (I4)	Output Last selected line.
Column to (I4)	Output The column position immediately following the last selected character.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION
EDIT-LINE-GET-SELECTION WITH #EA-1
#LINE-FROM #COLUMN-FROM
#LINE-TO #COLUMN-TO GIVING #RESPONSE ↩
```

301 EDIT-LINE-GET-TEXT Action

- Description 744
- Parameters 744

Description

Retrieves text in a line of the **Edit Area Control**.

Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA	Input Specifies an **Edit Area Control**.
Line number (I4)	Input The text is retrieved out of this line.
Column from (I4)	Input The text is retrieved from this column onwards.
Column to (I4)	Input/Output The text is retrieved up to this column. If you specify "0", the complete line is retrieved. The position immediately following the last character in the line will be returned (or 1, if the line is empty).
Line text (A253)	Output Returns the retrieved text string.
Split (L)	Output Indicates whether the total length of the requested text exceeds the actually retrieved line text string.
Response (I4)	Output Natural error (if applicable).

Example:

```
#LINE-NUMBER := 1
#COLUMN-FROM := 1
#COLUMN-TO := 0
PROCESS GUI ACTION EDIT-LINE-GET-TEXT WITH #EA-1 #LINE-NUMBER #COLUMN-FROM
#COLUMN-TO #LINE-TEXT #SPLIT GIVING #RESPONSE ↩
```

302 EDIT-LINE-INSERT Action

- Description 746
- Parameters 746

Description

Inserts a new line into an **Edit Area Control**.

Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA	Input Specifies an **Edit Area Control**.
Line number (I4)	Input The new line will be inserted before this line. If you specify "0", the new line will be appended to the last line of the **Edit Area Control** .
Line length (I4)	Input/Output The number of characters to be inserted into the new line starting from the first character of "Line text". If you specify "0", an empty line will be inserted. If you specify "-1", the "Line text" string will be copied into the new line, trailing blanks will be removed and the number of copied characters will be returned.
Line text (A)	Input The text string of the line to be inserted. **Note:** If this parameter is a dynamic or large alpha variable, it is possible to insert a line longer than 253 characters.
Response (I4)	Output Natural error (if applicable).

Example:

```
#LINE-NUMBER := 0
#LINE-LENGTH := 10
#LINE-TEXT := 'Hello!'
PROCESS GUI ACTION EDIT-LINE-INSERT WITH #EA-1 #LINE-NUMBER #LINE-LENGTH
#LINE-TEXT GIVING #RESPONSE ↩
```

program

303 EDIT-LINE-SET-SELECTION Action

- Description 748
- Parameters 748

Description

Selects a range of text (lines, columns) in an **Edit Area Control**. To set the text caret to a certain position, "Line from" and "Line to" must have the same value; "Column from" and "Column to" must also have the same value.

Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA	Input Specifies an **Edit Area Control**.
Line from (I4)	Input/Output Selection starts from this line onwards. If you specify "0", the last line is selected.
Column from (I4)	Input/Output Selection starts from this column onwards. If you specify "0", the end of the line is selected as the starting column.
Line to (I4)	Input/Output Last selected line. If you specify "0", the last line is selected. The number of this last line will be returned.
Column to (I4)	Input/Output Last selected column. If you specify "0", the complete line is selected. The last selected position in the line will be returned.
Response (I4)	Output Natural error (if applicable).

Example:

```
#LINE-FROM := 1          /* Select the first three lines
#LINE-TO := 3
#COLUMN-FROM := 1        /* Select from the first to the last column
#COLUMN-TO := 0
PROCESS GUI ACTION EDIT-LINE-SET-SELECTION WITH #EA-1 #LINE-FROM #COLUMN-FROM
#LINE-TO #COLUMN-TO GIVING #RESPONSE ↩
```

304 EDIT-LINE-SET-TEXT Action

- Description 750
- Parameters 750

Description

Replaces text in a line of the **Edit Area Control**. Note that you can also use this action to

- insert text by setting parameters "Column from" and "Column to" to the same value (this will be the insertion position); or to
- delete text by setting the parameter `Line length` to the value "0".

Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA	Input Specifies an **Edit Area Control**.
Line number (I4)	Input/Output The text is replaced out of this line.
Column from (I4)	Input/Output The text is replaced from this column onwards.
Column to (I4)	Input/Output The text is replaced up to this column. If you specify "0", the rest of the line is replaced. The last position in the line will be returned.
Line text (A253)	Input/Output Contains the string replacing the string specified with `Line number`, `Column from`, `Column to`.
Line length (I4)	Input/Output The number of "Line text" characters (out of 253) that replace the old string starting from the first character. If you specify "-1", trailing blanks will be removed and the number of characters used will be returned.
Response (I4)	Output Natural error (if applicable).

Example:

```
#LINE-NUMBER := 1       /*Replaces line 1
#COLUMN-FROM := 1
#COLUMN-TO := 0
#LINE-TEXT := 'New text'
PROCESS GUI ACTION EDIT-LINE-SET-TEXT WITH #EA-1 #LINE-NUMBER #COLUMN-FROM
#COLUMN-TO #LINE-TEXT #LINE-LENGTH GIVING #RESPONSE
MOVE #LINE-TEXT TO #EA-1.STRING ↩
```

305 ENUM-CHILDREN Action

- Description 754
- Parameters 754

Description

Enumerates the first or next direct or indirect child object belonging to a specified root object. The enumeration sequence id depth first, with parents being enumerated before their children.

The action may be called repeatedly, in order to enumerate all child objects belonging to a given root (ancestor) object. For the first call, both parameters should be set to the handle of the root object. The value NULL-HANDLE is returned by the output parameter if enumeration is complete.

Parameters

Name/Data Type	Explanation
Root (HANDLE OF GUI)	Input Handle of object whose direct and indirect children are being enumerated.
Current (HANDLE OF GUI)	Input/Output On input, the handle of the root object if this is the first call to this action (whereupon the handle of the first child object will be returned), otherwise the object handle returned by the previous call to this action (whereupon the handle of the next child object will be returned). On output, the handle of the first or next child object (as described above), or NULL-HANDLE if enumeration is complete.
Response (I4)	Output Natural error (if applicable).

Example:

```
DEFINE DATA LOCAL 1
 #TV-1 HANDLE OF TREEVIEW 1
 #ITEM HANDLE OF GUI
END-DEFINE * /*
 Output labels of all tree view items:
  #ITEM :=
  #TV-1 REPEAT PROCESS GUI ACTION
  ENUM-CHILDREN WITH #TV-1 #ITEM GIVING *ERROR IF #ITEM <> NULL-HANDLE
  WRITE #ITEM.STRING (AL=72)
  ELSE ESCAPE BOTTOM
 END-IF
END-REPEAT ↩
```

306 ENUM-CLIENT-KEYS Action

- Description 756
- Parameters 756

Description

Enumerates the client keys currently in use by a dialog or dialog element.

The action only returns a single key at a time, and must therefore be called repeatedly in order to retrieve the entire key list. The key that is returned is the "next" key, based on an internal enumeration cursor, which is originally reset (implying that the "first" key is returned) and which is implicitly advanced to the "next" key on each successive call to this action. By omitting the optional client key parameter, the internal enumeration cursor may be explicitly reset at any time.

If there are no further keys left to enumerate, an empty (i.e., all blank) client key value is returned.

If the client key/client value pair to which the internal enumeration cursor currently refers is deleted, the enumeration cursor is implicitly set to the "previous" key, if any, or is reset otherwise, such that the result of a subsequent call to this action is unaffected. It is thus possible to delete the keys whilst enumerating them (see example below).

Note that the words "first", "next" and "previous" relate to the internal client key sequence, which is implementation-defined, and thus may change in future Natural versions. A Natural application must not depend on this enumeration sequence!

For more information, please refer to the article Storing and Retrieving Client Data for a Dialog Element.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of dialog or dialog element, for which the keys are to be enumerated.
Client key (A253)	Output (optional parameter) If specified, this parameter retrieves the next key (if any) in use by the specified dialog or dialog element. If this parameter is omitted, the internal enumeration cursor is reset, implying that the next key returned by this action will be the "first" key.
Response (I4)	Output Natural error (if applicable).

Example:

```
DEFINE DATA LOCAL01
#CLIENT-KEY (A253)
...
END-DEFINE
*
/*reset enumeration cursor
PROCESS GUI ACTION ENUM-CLIENT-KEYS WITH #DLG$WINDOW GIVING *ERROR
/* Enumerate and delete the keys in use by the dialog
REPEAT
  PROCESS GUI ACTION ENUM-CLIENT-KEYS WITH #DLG$WINDOW #CLIENT-KEY
    GIVING *ERROR
  IF #CLIENT-KEY <> ' '
    /* Delete the key and associated value
    PROCESS GUI ACTION SET-CLIENT-VALUE WITH #DLG$WINDOW #CLIENT-KEY
      GIVING *ERROR
  END-IF
WHILE #CLIENT-KEY <> ' '
END-REPEAT ↩
```

307 ENSURE-VISIBLE Action

- Description 760
- Parameters 760

Description

Ensures that the specified list view item or tree view item is visible.

The list view or tree view control is automatically scrolled (and/or expanded, in the case of tree view controls), if necessary, in order to bring the specified item into view.

Parameters

Name/Data Type	Explanation
Item (HANDLE OF LISTVIEWITEM or HANDLE OF TREEVIEWITEM)	Input Handle of list view item or tree view item to be brought into view, if necessary.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION ENSURE-VISIBLE WITH #LVITEM-1 GIVING *ERROR ↩
```

308 GET-CLIENT-VALUE Action

- Description 762
- Parameters 762

Description

Retrieves user-defined information for a dialog or dialog element that has been set by the use of either the **CLIENT-KEY** and **CLIENT-VALUE** attributes, or the **SET-CLIENT-VALUE** action.

Unlike the use of the **CLIENT-VALUE** attribute, the retrieved data is moved (with MOVE -compatible conversion, if necessary) directly into the output field specified to retrieve the value, with no intermediate conversion to alpha being performed. If no value is found, the output field is implicitly RESET. In order to be able to differentiate between this case from the retrieval of an explicitly stored resetted value, an optional parameter may be supplied to return a flag indicating whether the specified key was found.

For more information, please refer to the article Storing and Retrieving Client Data for a Dialog Element.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of dialog or dialog element for which the client information is to be retrieved.
Client value (any data type)	Output Specifies the field, into which the value that is stored under the specified key is to be retrieved (with conversion, if necessary). If the specified key was not found for the specified dialog or dialog element, the field is reset.
Client key (A253)	Input (optional parameter) If specified, this is the key, for which the specified value is to be retrieved.If this parameter is omitted, the key (if any) specified by the **CLIENT-KEY** attribute of the specified dialog or dialog element is implicitly used.If the key is empty (i.e., all blank), the request is ignored
Found (L)	Output (optional parameter) If specified, this parameter is set to TRUE if the specified key was found for the specified dialog or dialog element, or FALSE otherwise.
Response (I4)	Output Natural error (if applicable).

Example:

```
DEFINE DATA LOCAL
01 #TIME (T)
01 #DATE (D)
...
END-DEFINE
*
/* Get value for (explicit) 'TIME' key for dialog
PROCESS GUI ACTION GET-CLIENT-VALUE WITH #DLG$WINDOW #TIME 'TIME'
  GIVING *ERROR

/* Get value for (implicit) 'DATE' key for dialog
#DLG$WINDOW.CLIENT-KEY := 'DATE'
PROCESS GUI ACTION GET-CLIENT-VALUE WITH #DLG$WINDOW #DATE
  GIVING *ERROR

 ↩
```

309 GET-CLIPBOARD-DATA Action

- Description 766
- Parameters 766

Description

Retrieves data in the specified Natural data format from the Windows or drag-drop clipboard. If a drag-drop operation is in progress, the data is preferentially read from the drag-drop clipboard (the local clipboard belonging to the source process), otherwise the data is read from the Windows clipboard.

Both pre-defined and private data formats may be used (see the **SET-CLIPBOARD-DATA** action). As for the **SET-CLIPBOARD-DATA** action, multiple data operands may be supplied, which can be any mixture of scalar and or array operands (including array index ranges) of potentially mixed types. In general, each data operand receives one data item (i.e., one text line in the case of the CF-TEXT format or one file name for the CF-FILELIST format) from the clipboard, and the delimiters and trailer (if any) corresponding to the pre-defined formats are skipped. However, if the last operand supplied is a dynamic alpha variable, all remaining unread data items are returned into the variable, including any embedded delimiters (carriage return/line feeds or null-terminators).

The incoming data is automatically converted to the format of the receiving operand in manner compatible with the MOVE statement if the data is not already in this format.

You can use the nX notation to skip one or more data items (see example below).

If more receiving fields are available than the number of items available on the clipboard plus the number of items skipped (if any), the remaining data fields are reset in a manner compatible with the RESET statement (see example below).

Parameters

Name/Data Type	Explanation
Format (A253)	Input Clipboard Natural data format. Can be standard numerc string for predefined formats (e.g., CF-TEXT) or user-defined string for private formats.
Data (List of any type except handles)	Input Data fields to receive the data. Can consist of any number of non-handle scalar and / or array operands, including array index ranges and multidimensional arrays. Dynamic alpha variables are also supported.
Response (I4)	Output Natural error (if applicable).

Example:

```
DEFINE DATA LOCAL
1 #FMT (A9) CONST<'MYPRIVFMT'>
1 #A (A32)
1 #I (I4) INIT<1880>
1 #D (D)
1 #RESPONSE (I4)
END-DEFINE
*
* The following example demonstrates the use of private formats
*
PROCESS GUI ACTION SET-CLIPBOARD-DATA WITH #FMT
  'A man a plan a canal - Panama' 42 *DATX GIVING #RESPONSE
*
* Read data back in (note use of nX notation to skip second item,
* and that #I is reset as all available data items have been exhausted)
*
PROCESS GUI ACTION GET-CLIPBOARD-DATA WITH #FMT
  #A 1x #D #I GIVING #RESPONSE
*
WRITE 'First item:' #A 'Third item:' #D '=' #I ↩
```

310 GET-FIRST-WINDOW Action

- Description 770
- Parameters 770

Description

Gets the first or last window (dialog or dialog element) in Z-order (depth) sequence that has the same parent as the specified reference window. The window returned can be the reference window itself if the reference window is at the top of the Z-order sequence.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Reference window. This can be a dialog or dialog element.
HANDLE OF GUI	Output First or last window in Z-order sequence containing reference window.
Reverse (L)	Input (optional parameter) If set to FALSE (= default), the first window in the Z-order sequence will be returned. If set to TRUE, the last window in the Z-order sequence will be returned.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION GET-FIRST-WINDOW WITH #CHILD #FIRST GIVING #RESPONSE ↩
```

311 GET-FOCUS Action

- Description 772
- Parameters 772

Description

Gets the handle of the dialog element which currently has the focus, that is, which responds to keyboard input.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Output The dialog element that has the focus.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION GET-FOCUS WITH #FOCUS GIVING #RESPONSE ↩
```

312 GET-MESSAGE-TEXT Action

- Description 774
- Parameters 774

Description

Gets the text for a given application message file number.

Parameters

Name/Data Type	Explanation
Message number (I4)	Input The message number for which the text is to be read.
Destination (A253)	Output The message text.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION GET-MESSAGE-TEXT WITH #MESSAGENR #DEST GIVING #RESPONSE ↩
```

313 GET-NEXT-WINDOW Action

- Description 776
- Parameters 776

Description

Gets the next or previous window (dialog or dialog element), if any, in Z-order (depth) sequence that has the same parent as the specified reference window.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Reference window. This can be a dialog or dialog element.
HANDLE OF GUI	Output Next or previous window in Z-order sequence containing reference window, or NULL-HANDLE if no such window is available (i.e., end of list)
Reverse (L)	Input (optional parameter) If set to FALSE (= default), the next window in the Z-order sequence will be returned. If set to TRUE, the previous window in the Z-order sequence will be returned.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION GET-NEXT-WINDOW WITH #CHILD #NEXT GIVING #RESPONSE  ↩
```

314 GET-TEXT Action

- Description 778
- Parameters 778

Description

Retrieves the text (if any) associated with the specified dialog or dialog element. For example, the contents of an edit area or the caption of a window or button.

This action returns the same information as a query of the dialog or dialog element's **STRING** attribute, if available. However, unlike the use of the attribute, this action allows text in excess of 253 characters to be retrieved.

Note that this action can only be used to retrieve text belonging to dialog elements that are windows or controls. For example, it can be used on a dialog or pushbutton, but not to retrieve the text of a list box item, even though the latter has a **STRING** attribute.

The receiving field may be a dynamic variable, allowing any significant trailing blanks in the text to be recognized and preserved.

Parameters

Name/Data Type	Explanation
Object (HANDLE OF GUI)	Input Handle of dialog or dialog element whose text is to be retrieved.
Text (A)	Output Text associated with specified dialog or dialog element.
Response (I4)	Output Natural error (if applicable).

Example:

```
DEFINE DATA LOCAL
1 #CONTROL HANDLE OF GUI
1 #TEXT (A) DYNAMIC
END-DEFINE
*
PROCESS GUI ACTION GET-TEXT WITH #CONTROL
  #TEXT GIVING *ERROR
WRITE #TEXT (AL=72) *LENGTH(#TEXT) ↩
```

315 GET-SUBITEM-DATA Action

- Description 780
- Parameters 780

Description

Retrieves the ("subitem") data for the specified list view item for the specified list view column(s).

The list view item handle may be specified as NULL-HANDLE, in which case the default data for the specified column(s) will be retrieved. The default data is also returned by this action in the case where no subitem data is present.

The column handle and data parameters may be repeated (in pairs), in order to allow the data for multiple columns to be retrieved in a single action.

The data will be converted (if necessary) from the column's data **FORMAT** to the specified field, with which it must therefore be MOVE -compatible.

For more information, please refer to the article Working with List View Controls.

Parameters

Name/Data Type	Explanation
Item (HANDLE OF GUI)	Input Handle of list view item whose data is to be retrieved, or NULL-HANDLE if the default data for the specified list view column(s) should be retrieved.
Column (HANDLE OF GUI)	Input Handle of list view column whose data is to be retrieved. Cannot be specified as NULL-HANDLE.
Data (any data type)	Output Field into which the column's data will be retrieved.
Response (I4)	Output Natural error (if applicable).

Example:

```
DEFINE DATA LOCAL
 1 #ID (I4)
 1 #CITY (A32)
 1 #RESPONSE (I4)
END-DEFINE
*
PROCESS GUI ACTION GET-SUBITEM-DATA WITH #LVITEM-1
  #LVCOL-1 #ID #LVCOL-2 #CITY GIVING #RESPONSE ↩
```

316 HELP Action

- Description 784
- Parameters 784

Description

Invokes the help system and points to the given help topic in the help file HELP-FILENAME determined by the attribute for the specified dialog, or *libraryname.hlp* if this has not been set. Natural expects the help file to be located in one of the directories listed in the desription of the **HELP-FILENAME** attribute.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input The dialog or dialog element acting as parent window for the help window. Must not be an item. This parameter is also used to retrieve any non-default help file name via dialogs **HELP-FILENAME** attribute.
Help ID (I4)	Input The help topic ID.
Response (I4)	Output Natural error (if applicable).

Example:

```
#EA-1.HELP-ID := 1234
  /*Set 1234 as help topic ID for the edit area
  PROCESS GUI ACTION HELP WITH #EA-1 #EA-1.HELP-ID GIVING #RESPONSE ↩
```

317 HOURGLASS-REMOVE Action

- Description 786
- Parameters 786

Description

Sets the pointer shape from "hourglass" to "arrow" and clears the pointer stack. You can also use this action to suspend the hourglass pointer and use the arrow shape while displaying a message box during long periods of processing. You resume with the hourglass pointer on the same nesting level as when it was suspended by passing the value returned in the level parameter to the **HOURGLASS-STACK** Action.

Parameters

Name/Data Type	Explanation
Level (I4)	Output The pointer stack level before the stack is cleared.

Example:

```
PROCESS GUI ACTION HOURGLASS-REMOVE WITH #LEVEL GIVING #RESPONSE ↩
```

318 HOURGLASS-STACK Action

- Description 788
- Parameters 788

Description

Sets the pointer shape to "hourglass" and keeps the previous pointer shape in the stack of pointers. This indicates that end-user input is disabled during a long period of processing. To restore the previous shape (hourglass or arrow), you use the **HOURGLASS-UNSTACK** Action. This stack/unstack logic can be used in nested program structures to determine the pointer shape.

The level parameter can be used to resume the hourglass on the same stack level as when it was suspended by the **HOURGLASS-REMOVE** Action.

Note: While a Natural dialog is performing a lengthy operation during which it is not able to process events, it should display the hourglass to the user to indicate that it is currently not available for operation. While the hourglass is active, no events will be delivered to the dialog.

Parameters

Name/Data Type	Explanation
Level (I4)	Input The pointer stack level (max. 10) up to which the stack is filled with hourglass pointers.

Note: The `Level` parameter is optional.

Example:

```
PROCESS GUI ACTION HOURGLASS-STACK WITH #LEVEL GIVING #RESPONSE
```

319 HOURGLASS-UNSTACK Action

- Description 790
- Parameters 790

Description

Restores the previous pointer shape (hourglass or arrow) by decreasing the stack of pointers by one and by setting the pointer to the new stack top. If the stack is empty, the arrow shape is activated. This action corresponds to action **HOURGLASS-STACK**.

Parameters

Name/Data Type	Explanation
Level (I4)	Output The current pointer stack level.

Note: The `Level` parameter is optional.

Example:

```
PROCESS GUI ACTION HOURGLASS-UNSTACK WITH #LEVEL GIVING #RESPONSE
```

320 INPUT-COPY-SELECTION Action

- Description 792
- Parameters 792

Description

The text currently selected in the input dialog element is copied to the clipboard. The text can be selected in an **Input Field Control** , a **Selection Box Control** or an **Edit Area Control**.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an **Input Field Control** , a **Selection Box Control** or an **Edit Area Control**.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION INPUT-COPY-SELECTION WITH #IF-1 GIVING #RESPONSE ↩
```

Note: For selection-boxes, the action must be called whilst the selection-box still has the focus.

321 INPUT-CUT-SELECTION Action

- Description 794
- Parameters 794

Description

The text currently selected in the input dialog element is copied to the clipboard and is deleted in the original place. The text can be selected in an **Input Field Control**, a **Selection Box Control** or an **Edit Area Control**.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an **Input Field Control**, a **Selection Box Control** or an **Edit Area Control**.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION INPUT-CUT-SELECTION WITH #IF-1 GIVING #RESPONSE ↩
```

Note: For selection-boxes, the action must be called whilst the selection-box still has the focus.

322 INPUT-DELETE-SELECTION Action

- Description 796
- Parameters 796

Description

The text currently selected in the input dialog element is deleted. The text can be deleted from an **Input Field Control** , a **Selection Box Control** or an **Edit Area Control**.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an **Input Field Control** , a **Selection Box Control** or an **Edit Area Control**.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION INPUT-DELETE-SELECTION WITH #IF-1 GIVING #RESPONSE ↩
```

Note: For selection-boxes, the action must be called whilst the selection-box still has the focus.

323 INPUT-GET-LINE-LENGTH Action

- Description 798
- Parameters 798

Description

Gets the length of a line in an **Edit Area Control** or in an **Input Field Control** .

Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA or INPUTFIELD	Input Specifies an **Edit Area Control** or an **Input Field Control**.
Line number (I4)	Input Length is retrieved from this line.
Line length (I4)	Output Returns the line length.
Response (I4)	Output Natural error (if applicable).

Example:

```
#LINE-NUMBER := 1   /* Examine the first line
PROCESS GUI ACTION INPUT-GET-LINE-LENGTH WITH #EA-1 #LINE-NUMBER #LINE-LENGTH
GIVING #RESPONSE
#IF-1.STRING := #LINE-LENGTH /* Display the result in this Input Field Control
```

324 INPUT-GET-SELECTION Action

- Description 800
- Parameters 800

Description

Retrieves the position of the selected text in an **Edit Area Control** or in an **Input Field Control** from the beginning (character number) to the end (character number).

Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA or INPUTFIELD	Input Specifies an **Edit Area Control** or an **Input Field Control**.
Position from (I4)	Output Selection starts from this position onwards.
Position to (I4)	Output Last selected position.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION INPUT-GET-SELECTION WITH #EA-1 #POSITION-FROM #POSITION-TO
GIVING #RESPONSE
#IF-1.STRING := #POSITION-FROM /* Display the result in these two
#IF-2.STRING := #POSITION-TO   /* Input Field Controls ↩
```

325 INPUT-GET-TEXT Action

- Description 802
- Parameters 802

Description

Retrieves text in a line of the **Edit Area Control** or in an **Input Field Control** (from a position onwards).

Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA or INPUTFIELD	Input Specifies an **Edit Area Control** or an **Input Field Control**.
Position-from (I4)	Input The text is retrieved from this position onwards.
Line length (I4)	Input/Output The number of the characters to be retrieved (must be 1 to 253).
Line text (A253)	Output The retrieved text will be copied into this text string.
End (L)	Output Becomes TRUE if the last retrieved character was the last character in the dialog element. If `End` is TRUE, the number of retrieved characters is returned in the `Line length` parameter.
Response (I4)	Output Natural error (if applicable).

Example:

```
#POSITION := 8     /* Set input
#LINE-LENGTH := 22 /* values
PROCESS GUI ACTION INPUT-GET-TEXT WITH #EA-1 #POSITION-FROM #LINE-LENGTH
   #LINE-TEXT #END GIVING #RESPONSE
#IF-1.STRING := #POSITION    /* Display output values
#IF-2.STRING := #LINE-LENGTH /* in Input Field Controls
#IF-3.STRING := #TEXT ↩
```

326 INPUT-PASTE Action

- Description 804
- Parameters 804

Description

Copies the content of the clipboard into an **Input Field Control** , a **Selection Box Control** , or an **Edit Area Control**. If nothing is selected inside the dialog element, the clipboard text is inserted at the cursor position. If an area of text is selected, this selection is deleted and the content of the clipboard is inserted.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies a **Selection Box Control** or an **Edit Area Control**.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION INPUT-PASTE WITH #IF-1 GIVING #RESPONSE ↩
```

327 INPUT-SET-SELECTION Action

- Description 806
- Parameters 806

Description

Selects an area (position to position) in an **Edit Area Control** or in an **Input Field Control** .

Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA or INPUTFIELD	Input Specifies an **Edit Area Control** or an **Input Field Control**.
Position from (I4)	Input Selection starts from this position onwards.
Position to (I4)	Input Last selected position. If you specify "0", the selection will extend to the last character.
Response (I4)	Output Natural error (if applicable).

Example:

```
#POSITION-FROM := 10  /* Select from the 10th to the 22nd character
#POSITION-TO := 22
PROCESS GUI ACTION INPUT-SET-SELECTION WITH #EA-1 #POSITION-FROM #POSITION-TO
GIVING #RESPONSE ↩
```

328 INPUT-SET-TEXT Action

- Description 808
- Parameters 808

Description

Replaces text in an **Edit Area Control** or in an **Input Field Control** (from position to position).

Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA or INPUTFIELD	Input Specifies an **Edit Area Control** or an **Input Field Control**.
Position-from (I4)	Input Position of the first character to be replaced. If you specify "0" the text is appended to the text already present in the dialog element.
Position-to (I4)	Input/Output Position of the last character to be replaced. If you specify "0", the index of the last character in the dialog element will be returned.
Text length (I4)	Input/Output The number of characters to be inserted before `Position-to` starting from the first character of "Line text". If you specify "-1", the "Line text" string will be inserted into the new line, trailing blanks will be removed and the number of copied characters will be returned.
Text (A253)	Input This text string replaces the old one.
Response (I4)	Output Natural error (if applicable).

Example:

```
#POSITION-FROM := 8
#POSITION-TO := 24
#TEXT-LENGTH := 16
#TEXT := 'Insert this text'
PROCESS GUI ACTION INPUT-SET-TEXT WITH #EA-1 #POSITION-FROM #POSITION-TO
#TEXT-LENGTH #TEXT GIVING #RESPONSE ↩
```

329 INPUT-UNDO Action

- Description 810
- Parameters 810

Description

Undoes the last editing action in an **Input Field Control**, a **Selection Box Control** or an **Edit Area Control**. Editing actions that can be undone are entering, copying, pasting, cutting and deleting text in a dialog element.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an **Input Field Control** , a **Selection Box Control** or an **Edit Area Control**.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION INPUT-UNDO WITH #EA-1 GIVING #RESPONSE ↩
```

330 INQ-CLICKPOSITION Action

- Description 812
- Parameters 812

Description

Returns the x-axis and the y-axis position where the end user has clicked (relative to the dialog element's rectangle). These coordinates are also updated immediately before a context-menu's before-open event is called. The application can then combine this call with a call to either **INQ-ITEM-BY-POSITION** or **TABLE-INQUIRE-CELL** to determine which list box item or **Table Control** cell was *right-clicked*.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies a dialog element.
X-Position (I4)	Output (optional parameter) The x-axis position in pixels.
Y-Position (I4)	Output (optional parameter) The y-axis position in pixels.
Valid (L)	Output (optional parameter) The click position is valid. If this parameter is set to FALSE, there is no click position (e.g., when a context menu is accessed via the keyboard), and the coordinate (0, 0) is returned (if requested).
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION INQ-CLICKPOSITION WITH #BM-1 #X-POSITION #Y-POSITION
GIVING #RESPONSE
#IF-1.STRING := #X-POSITION  /* Display the coordinates in these two
#IF-2.STRING := #Y-POSITION  /* Input Field Controls ↩
```

331 INQ-DRAG-DROP Action

- Description 814
- Parameters 814

Description

Retrieves miscellaneous information relating to the current drag-drop operation.

This action should only be called if a drag-drop operation is in progress. Note that not all parameters are relevant and / or available for all drag events.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Output (optional parameter) The handle of the dialog containing the drop target, if available.
HANDLE OF GUI	Output (optional parameter) The drag source control's handle, if available.
Mode (I4)	Output (optional parameter) Indicates which augmentation keys are pressed. Possible values: 0 No key or information not available. 1 SHIFT key. 2 CTRL key. 3 Both SHIFT and CTRL keys
X-Position (I4)	Output (optional parameter) X-axis position on the target.
Y-Position (I4)	Output (optional parameter) Y-axis position on the target.
Buttons (I4)	Output (optional parameter) Indicates which mouse buttons are pressed. Possible values: 0 = No mouse button or information not available 1 = Left mouse button 2 = Right mouse button 4 = Middle mouse button. If multiple mouse buttons are pressed, the value returned is the sum of the individual values.
Drop Effect (I4)	Output (optional parameter) Indicates the type of drag-drop operation that occurred. Possible values: DM-NONE (0) = No drop, or information not available. DM-COPY (1) = Copy operation DM-MOVE (2) = Move operation DM-LINK (4) = Link operation
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION INQ-DRAG-DROP WITH #DIA-HANDLE #DIA-ELEMENT #MODE
#X-POSITION #Y-POSITION GIVING #RESPONSE
#IF-1.STRING := #X-POSITION  /* Display the coordinates in these two
#IF-2.STRING := #Y-POSITION  /* Input Field Controls
```

332 INQ-FORMAT-AVAILABLE Action

- Description 818
- Parameters 818

Description

Queries whether data is available in the specified format on the Windows or drag-drop clipboard. If a drag-drop operation is in progress, the drag-drop clipboard (the local clipboard belonging to the source process) is preferentially queried, otherwise the Windows clipboard is queried.

The result of the query can be used to determine whether Paste commands should be enabled or disabled, or whether a drop is allowed during a drag-drop operation.

For more information on clipboard formats, see the **SET-CLIPBOARD-DATA** action.

Parameters

Name/Data Type	Explanation
Format (A253)	Input Clipboard Natural data format. Can be standard numerc string for predefined formats (e.g., CF-TEXT) or user-defined string for private formats.
Exists (L)	Output Returns TRUE if data is available in the specfied format, or FALSE otherwise.
Response (I4)	Output Natural error (if applicable).

Example:

```
DEFINE DATA LOCAL
1 #AVAIl (L)
1 #RESPONSE (I4)
1 #CONTROL HANDLE OF GUIEND-DEFINE
*
* The following example demonstrates a typical DRAG-ENTER event handler
*
PROCESS GUI ACTION INQ-FORMAT-AVAILABLE WITH CF-TEXT #AVAIL
  GIVING #RESPONSE
*
* Allow or disallow drop based on clipboard format availability
*
#CONTROL := *CONTROL
IF #AVAIL
  #CONTROL.SUPPRESS-DRAG-DROP-EVENT := NOT-SUPPRESSED
ELSE
```

```
  #CONTROL.SUPPRESS-DRAG-DROP-EVENT := SUPPRESSED
END-IF

 ↩
```

333 INQ-INNER-RECT Action

- Description 822
- Parameters 822

Description

Retrieves the dimensions of the client rectangle, or the dimensions and relative position of the interior rectangle, of the screen, a dialog element, or the client window of a dialog. For SDI (standard window) dialogs and MDI child dialogs, the client window is the interior window onto which the dialog elements are usually placed. For MDI frame dialogs, it is the MDI client window (the window that hosts the MDI child dialogs). For dialog elements, the client rectangle is the area excluding any frame components such as borders, margins and scroll bars. If the dialog element possesses an implicit child client window (as is the case with tab controls, for example), this action applies to this client window rather than to its parent.

The interior rectangle is, in most cases, identical to the client rectangle. The exception is the screen, where the client rectangle is the full screen size and the interior rectangle is the screen area that excludes any desktop toolbars (such as the system taskbar).

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of the dialog or dialog element. If NULL-HANDLE is specified, the screen is implied.
Width (I4)	Output Returns the width (in pixels) of the client rectangle, if neither the `Left` nor the `Top` parameter is supplied, or the width of the interior rectangle otherwise.
Height (I4)	Output Returns the height (in pixels) of the client rectangle, if neither the `Left` nor the `Top` parameter is supplied, or the height of the interior rectangle otherwise.
Left (I4)	Output (optional parameter) Returns the horizontal offset (in pixels) of the left edge of the interior rectangle relative to the left edge of the client rectangle.
Top (I4)	Output (optional parameter) Returns the vertical offset (in pixels) of the top edge of the interior rectangle relative to the top edge of the client rectangle.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION INQ-INNER-RECT WITH #DLG$WINDOW #WIDTH #HEIGHT GIVING
#RESPONSE
#IF-1.STRING := #HEIGHT /* Display the width and height
#IF-2.STRING := #WIDTH
```

334 INQ-ITEM-BY-POSITION Action

- Description 826
- Parameters 826

Description

Returns the item at a given position within a control. This is particularly useful for finding out which **List Box Item** or **Status Bar Pane** was clicked with the right mouse button before a context menu is displayed (see example below).

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies a list box or status bar control.
X-Position (I4)	Input Specifies the x-axis position in pixels relative to the top-left hand corner of the control.
Y-Position (I4)	Input Specifies the y-axis position in pixels relative to the top-left hand corner of the control.
Item (HANDLE OF GUI)	Output The handle of the **List Box Item** or **Status Bar Pane** at the specified position, or NULL-HANDLE if none.
Response (I4)	Output NATURAL error (if applicable).

Example:

```
/* Sample BEFORE-OPEN code for context menu:
PROCESS GUI ACTION INQ-CLICKPOSITION WITH
#LB-1 #X-POSITION #Y-POSITION GIVING #RESPONSE
PROCESS GUI ACTION INQ-ITEM-BY-POSITION WITH
#LB-1 #X-POSITION #Y-POSITION #LBITEM GIVING #RESPONSE ↩
```

335 INQ-NON-CLIENT-METRICS Action

- Description 828
- Parameters 828

Description

Retrieves miscellaneous non-client window dimensions and settings from the operating system. The "small caption" referred to below is the caption used for floating tool bar controls.

Note: All font descriptions are in the standard form used by the **FONT-STRING** attribute.

Parameters

Name/Data Type	Explanation
Border width (I4)	Output (optional parameter) Thickness (in pixels) of the sizing border.
Scroll width (I4)	Output (optional parameter) Width (in pixels) of a vertical scroll bar.
Scroll height (I4)	Output (optional parameter) Height (in pixels) of a horizontal scroll bar.
Caption width (I4)	Output (optional parameter) Width (in pixels) of caption buttons.
Caption height (I4)	Output (optional parameter) Height (in pixels) of caption buttons.
Caption font (A253)	Output (optional parameter) Caption font string.
Small caption width (I4)	Output (optional parameter) Width (in pixels) of small caption buttons.
Small caption height (I4)	Output (optional parameter) Height (in pixels) of small caption buttons.
Small caption font (A253)	Output (optional parameter) Small caption font string.
Menu width (I4)	Output (optional parameter) Width (in pixels) of menu bar buttons.

Name/Data Type	Explanation
Menu height (I4)	Output (optional parameter) Height (in pixels) of menu bar buttons.
Menu font (A253)	Output (optional parameter) Menu bar font string.
Status font (A253)	Output (optional parameter) Status bar font string. **Note:** This font is also used for tooltips.
Message font (A253)	Output (optional parameter) Message box font string.
Response (I4)	Output Natural error (if applicable).

Example:

```
DEFINE DATA LOCAL
1 NONCLIENTMETRICS
  2 BORDER-WIDTH (I2)
  2 SCROLL-WIDTH (I2)
  2 SCROLL-HEIGHT (I2)
  2 CAPTION-WIDTH (I2)
  2 CAPTION-HEIGHT (I2)
  2 CAPTION-FONT (A80)
  2 SMALL-CAPTION-WIDTH (I2)
  2 SMALL-CAPTION-HEIGHT (I2)
  2 SMALL-CAPTION-FONT (A80)
  2 MENU-WIDTH (I2)
  2 MENU-HEIGHT (I2)
  2 MENU-FONT (A80)
  2 STATUS-FONT (A80)
  2 MESSAGE-FONT (A80)
*
1 #FONT HANDLE OF FONT
1 #WIDTH (I4)
1 #HEIGHT (I4)
END-DEFINE
...
PROCESS GUI ACTION INQ-NON-CLIENT-METRICS WITH NONCLIENTMETRICS
     GIVING *ERROR
*
/* Create font based on status bar font description
PROCESS GUI ACTION ADD WITH PARAMETERS
     PARENT = #DLG$WINDOW
```

```
         TYPE = FONT
         HANDLE-VARIABLE = #FONT
         STRING = NONCLIENTMETRICS.STATUS-FONT
END-PARAMETERS GIVING *ERROR
*
/* Get width and height of text "Test" in status bar font
PROCESS GUI ACTION TEXT-GET-EXTENT WITH #FONT 'Test'
    #WIDTH #HEIGHT GIVING *ERROR
*
/* Delete font we created if no longer needed
PROCESS GUI ACTION DELETE WITH #FONT

 ↩
```

336 LOAD-LAYOUT Action

- Description 832
- Parameters 832

Description

Loads the specified dialog's control bar layout (if any) that was previously saved for the current user via the **SAVE-LAYOUT** Action. If no such saved layout exists, this action has no effect.

Note that the bar types (dockable or fixed) should match those at the time the layout was saved, otherwise the results are unpredictable.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of the dialog for which the control bar layout is to be loaded.
Profile name (A253)	Input (optional parameter) The name of the profile from which the information is to be loaded. If this parameter is not specified, the name of the library containing the dialog is used.
Section name (A253)	Input (optional parameter) The name of the profile section from which the information is to be loaded. If this parameter is not specified, the name of the dialog is used.

Example:

```
PROCESS GUI ACTION LOAD-LAYOUT WITH #DLG$WINDOW GIVING #RESPONSE ↩
```

337 MOVE-NAVIGATION-ITEMS Action

- Description 834
- Parameters 834

Description

Moves a range of controls to a new position in the control sequence. The control sequence within the range of controls being moved remains unchanged.

Parameters

Name/Data Type	Explanation
First control (HANDLE OF GUI)	Input Specifies the first control in the range of controls which is to be moved. If this parameter is specified as NULL-HANDLE, the first control in the control sequence is used.
Last control (HANDLE OF GUI	Input Specifies the last control in the range of controls which is to be moved. If this parameter is specified as NULL-HANDLE, the last control in the control sequence is used.
Position To (HANDLE OF GUI)	Input Specifies the control which is to immediately precede the moved control(s) in the new control sequence. This control must not be one of the controls being moved. If this parameter is specified as NULL-HANDLE, the controls are moved to the end of the control sequence. If this parameter is specified as the handle of the dialog itself, the controls are moved to the front of the control sequence.
Response (I4)	Output NATURAL error (if applicable).

Example:

```
  /* Move the controls in the current control sequence starting from #PB-1 and ↩
ending with the last
  /* control in the control sequence to follow #SB-1
  PROCESS GUI ACTION MOVE-NAVIGATION-ITEMS WITH
      #PB-1 NULL-HANDLE #SB-1 GIVING #RESPONSE
```

338 MESSAGE-BOX Action

- Description 836
- Parameters 836

Description

Displays a standard modal message box.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of the parent dialog (or dialog element within the parent dialog). If NULL-HANDLE is specified, the active window is implicitly used as the parent window. **Note:** A message box is always displayed relative to, and immediately in front of, its parent window.
Message (A)	Input The message text to be displayed. **Note:** If this parameter is a dynamic or large alpha variable, it is possible to set a message text longer than 253 characters.
Title (A253)	Input (optional parameter) The text to be displayed in the title bar of the message box. If this parameter is not supplied, the message box is displayed with no title.
Style (A32)	Input (optional parameter) The type of message box (for possible input values, see below). If this parameter is not supplied, the message box is displayed with an OK push button, and without an icon.
Button (A1)	Output (optional parameter) Returns the selected button (for possible output values, see below).
Response (I4)	Output Natural error (if applicable).

The `Style` parameter may consist of one or more of the following characters:

Style Value	Message Box Type
I	Informational icon (e.g., lower case "i" in blue circle) displayed.
!	Warning icon (e.g., exclamation mark) displayed.
S	Critical error icon (e.g., "Stop" sign) displayed.
?	Prompt icon (e.g. question mark) displayed. Indicates that the user should make a choice between two or more options. However, it is modern practice to use one of the above icons instead, to indicate the severity of the error.
may be combined with:	
O	OK push button (default).
OC	OK and Cancel push buttons.
YNC	Yes, No, and Cancel push buttons.
YN	Yes and No push buttons.
RC	Retry and Cancel push buttons until the end user responds to the message box.
may be combined with:	
1	Make the first push button the default (default).
2	Make the second push button (if any) the default.
3	Make the third push button (if any) the default.

Note: If the messagebox has the style "C", an OK button is generated because a messagebox with only a Cancel button is not supported.

The Button parameter may consist of one or more of the following characters:

Button Value	Selected Button
O	OK push button..
C	Cancel push button.
Y	Yes push button.
N	No push button.
R	Retry push button.

Example:

```
PROCESS GUI ACTION MESSAGE-BOX WITH #DLG$WINDOW
'Do you want to save the changes ?' 'Exit editor' '?YNC1'
#BUTTON
GIVING *ERROR
```

where #BUTTON is defined as:

```
01 #BUTTON (A1)
```

339 OLE-ACTIVATE

- Description 840
- Parameters 840

Description

Activates the default action of an OLE server programmatically. To enable the end user to activate the OLE server, you set the OLE container control's **MODIFIABLE** attribute to TRUE. The end user then activates the OLE server by using the right mouse button to click in the OLE container control's rectangle; this shows the context menu, from which the end user can choose the appropriate command. The OLE server can also be activated by using the left mouse button to double-click in the OLE container control's rectangle. If the menu item Show Object Verbs is provided, the end user can also use this to activate the server.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an **OLE Container Control**.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION OLE-ACTIVATE WITH #OCT-1 GIVING #RESPONSE ↩
```

340 OLE-DEACTIVATE

- Description 842
- Parameters 842

Description

Deactivates an OLE server programmatically, that is, in-place editing is finished. The end user can also deactivate the server by pressing ESC or by clicking into the Natural dialog outside the **OLE Container Control** .

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an **OLE Container Control**.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION OLE-DEACTIVATE WITH #OCT-1 GIVING #RESPONSE ↩
```

341 OLE-GET-DATA

- Description 844
- Parameters 844

Description

Reads an embedded object into a Natural variable. It is recommended to define an array of variables that is large enough to hold the object: some objects are at least 10 KB in size. You then get the object by executing this procedure; the second parameter contains the size of the array. The procedure returns the real size being used and a Natural error code other than "0" is returned. To avoid getting this error code, you first query the current size of the object with the **OBJECT-SIZE** attribute.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an **OLE Container Control**.
Variable	Input Variable or (usually) an array of variables.
Size (I4)	Input Size of the variable or the array of variables provided.
Real size (I4)	Output Real size of the variable or the array of variables.
Response (I4)	Output Natural error (if applicable).

Example:

```
#CURRSIZE := #OCT-1.OBJECT-SIZE  /* How large is the object?
IF..  /* If the object is too large, do the following...
...   /* If not, execute the procedure
END-IF
PROCESS GUI ACTION OLE-GET-DATA WITH #OCT-1 #MYVARI (1:5) 15000 #REALSIZE GIVING
#RESPONSE ↩
```

342 OLE-INSERT-OBJECT

- Description 846
- Parameters 846

Description

Provides the end user with a dialog box which allows him/her to select and start an OLE server application or an external object.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an **OLE Container Control**.
String (A253)	Input Dialog box caption.
Flag (I4)	Input For future use.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION OLE-INSERT-OBJECT WITH #MYCONTAINER 'My Caption' #FLAG
GIVING #RESPONSE  ↩
```

343 OLE-READ-FROM-FILE

- Description 848
- Parameters 848

Description

Reads an embedded object into the **OLE Container Control**.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an **OLE Container Control**.
File name (A253)	Input Specifies the name of an embedded object (in Natural, this is a ".neo" file)
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION OLE-READ-FROM-FILE WITH #MYCONTAINER 'c:\natgui\myobject.neo'
GIVING #RESPONSE ↩
```

344 OLE-SAVE-TO-FILE

- Description 850
- Parameters 850

Description

Saves the currently embedded object to a file with the default extension ".neo" (meaning "Natural Embedded Object").

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an **OLE Container Control**.
File name (A253)	Input Specifies the name of an embedded object (in Natural, this is a ".neo" file).
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION OLE-SAVE-TO-FILE WITH #MYCONTAINER 'MYOBJECT.NEO'
GIVING #RESPONSE
```

345 OLE-SET-DATA

- Description 852
- Parameters 852

Description

Puts the content of a Natural variable into an embedded object. Corresponds with the **OLE-GET-DATA** Action. You put the variable content into the object by executing this procedure; the second parameter contains the size of the object. It is recommended to define an array of variables that is large enough to hold the object: some objects are at least 10 KB in size.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an **OLE Container Control**.
Variable	Input Variable or (usually) an array of variables.
Size (I4)	Input Size of the object.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION OLE-SET-DATA WITH #OCT-1 #MYVARI (1:5) #REALSIZE
GIVING #RESPONSE
```

346 OPEN-CLIPBOARD Action

- Description 854
- Parameters 854

Description

Empties the current contents (if any) of the local clipboard, and opens it for writing (see the **SET-CLIPBOARD-DATA** action). The contents of the global (Windows) clipboard are not modified by this action.

Each Natural process has its own local clipboard that is used to build up data destined either for the Windows clipboard, or for the target of a drag-drop operation. In the latter case, the local clipboard of the source process is also referred to as the "drag-drop clipboard".

This action allows a clipboard owner to optionally be set. If an owner is specified, the data is stored with the owner, and is retrieved from the owner by the Windows clipboard whenever the clipboard data is requested. If the owner is destroyed, Natural implicitly flushes the data to the Windows clipboard to ensure that it remains available. Although there is usually little or no impact from these internal activites on the application itself, it is recommended that, if the handle of the control (if any) sourcing the data be specified as the owner. For example, if the data corresponding to some selected list box items is to be placed on the clipboard, it is recommended that the clipboard owner be specified as the handle of the list box to which these list box items belong.

Note that if data for a drag-drop operation is being prepared in the Begin-Drag event, an explicit call to this action is not necessary, since Natural implicitly opens the drag-drop clipboard in this case.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input (optional parameter) The handle of the dialog or dialog element owning the clipboard (see above). If this parameter is omitted, or is specified as NULL-HANDLE (default), the clipboard has no owner.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION OPEN-CLIPBOARD WITH #LB-1
GIVING #RESPONSE ↩
```

347 PERFORM-DRAG-DROP Action

- Description 858
- Parameters 858

Description

Manually initiates a drag-drop operation.

Calling this action is not necessary for dialog elements supporting automatic drag-drop (e.g., list box controls). In this case, Natural initiates a drag operation automatically in response to the relevant user actions. Regardless of whether the drag-drop operation is initiated automatically or manually (via this action), the drag source control must have a **DRAG-MODE** attribute value other than the default value of DM-NONE in order for the drag-drop operation to be successfully started.

The drag source receives a **BEGIN-DRAG** event when the drag operation begins, and an **END-DRAG** event when the drag operation ends (either with or without a drop having occurred). The drag operation occurs optionally either immediately or after the cursor is moved a system-defined minimum threshold distance of typically a few pixels. In the latter case, no **BEGIN-DRAG** or **END-DRAG** event is received by the drag source if the operation is cancelled (e.g. by releasing the mouse button that was depressed on calling this action) before the cursor movement threshold has been reached, and no drag cursor is displayed.

The drag source must respond to the **BEGIN-DRAG** event by placing some data on the drag-drop clipboard using the **SET-CLIPBOARD-DATA** action, otherwise the drag-drop operation is implicitly cancelled by Natural.

Note that this action is, in practice, currently limited to ActiveX controls, as only these controls (sometimes) support the primitive mouse button events that are appropriate for calling this action.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of dialog element which is to act as the drag source (and thus receive the **BEGIN-DRAG** and **END-DRAG** events, if any).
Immediate (L)	Input (optional parameter) Indicates whether the drag-drop operation should occur immediately or after the cursor moves a system-defined minimum threshold distance (=default).
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION PERFORM-DRAG-DROP
GIVING *ERROR ↩
```

348 PICK-FILENAME Action

- Description 862
- Parameters 862

Description

Invokes a standard dialog box that allows the end user to pick the name of a file which is *optionally* assigned to a Natural work file number.

Parameters

Name/Data Type	Explanation
Title (A253)	Input The text displayed in the title bar of the dialog box.
Init (A253)	Input A string pattern (*[directoryname\ ][filename[.extension]]*) that determines the initial directory (if specified), together with either the files to choose from (if the *filter list* parameter is not specified) or the pre-selected file (if a filename and extension containing no wildcard characters (*,?) is used). If the pattern is an empty string, all files of the current directory are listed.
Work file number (I4)	Input The work file number to which the selected file name is to be assigned. If this parameter is "0", the file name is not assigned to a work file number.
File name (A253)	Output Returns the selected file name with the entire path name.
Dialog Type (I4)	Input (optional parameter) Determines type of file selection dialog displayed: 0 (default) = 'Open' dialog (new files allowed); 1 = 'Open' dialog (selected file must exist); 2 = 'Save' dialog.
Filter list (A253/*)	Input (optional parameter) A one-dimensional array containing pre-specified display filter definitions. The array does not need to be filled entirely. Each filter definition is represented by a *pair* of array elements. The first element of each pair is a textual description of the filter used for display in the file selection dialog. The second element of each pair is the associated display filter pattern. The display filter pattern may be complex, consisting of multiple components separated by a semi-colon.
Filter index (I4)	Input (optional parameter) The index of the filter to be initially used when the file selection dialog is first opened. 0 = first filter, 1 = second filter , and so on.

Name/Data Type	Explanation
Response (I4)	Output Natural error (if applicable).

Example:

```
DEFINE DATA LOCAL
   1 #RESPONSE (I4)
   1 #MYTITLE (A253)
   1 #SELECTNAME (A253)
   1 #FILTER (A32/6) CONST
     <
       'Private resource files', '*.nr*',
       'Shared resource files',  '*.bmp;*.ico;*.hlp;*.neo;*.rpt',
       'All Files (*.*)',        '*.*'
     >
   END-DEFINE
   ...
   #MYTITLE := 'Example using implicit display filter'
   PROCESS GUI ACTION PICK-FILENAME WITH #MYTITLE 'c:\*.*' 0
   #SELECTNAME GIVING #RESPONSE
   #IF-1.STRING := #SELECTNAME
   *
   #MYTITLE := 'Example using explicit display filter'
   PROCESS GUI ACTION PICK-FILENAME WITH #MYTITLE 'd:\fuser\mylib\res\'
     0 #SELECTNAME 0 #FILTER(*) 1 GIVING #RESPONSE
   #IF-2.STRING := #SELECTNAME
```

349 PLAY-SOUND Action

- Description 866
- Parameters 866

Description

Plays a tune from a ".WAV" audio file.

Note: This action is not available under Windows NT.

Parameters

Name/Data Type	Explanation
Audio file name (.wav)	Input Either the full path name of the audio file or just the file name. If you specify just the file name, this file must be located in the directory referred to by the environment variable NATGUI_BMP. If you specify a path name, this must be the fully expanded path name including the drive and the path specifications.

Examples:

```
PROCESS GUI ACTION PLAY-SOUND WITH 'CHIMES.WAV'
PROCESS GUI ACTION PLAY-SOUND WITH #FILE
GIVING #RESPONSE ↩
```

350 PROCESS-EVENTS Action

- Description 868
- Parameters 868

Description

Processes all outstanding messages for the Natural application (raising any associated events) before returning. This is particularly useful for allowing the user to interrupt a long operation. Typically, you create a simple abort dialog of style 'modal' with a 'Cancel' push button, and call this action periodically whilst processing the long operation. If the user has clicked the 'Cancel' push button in the meantime, the **CLICK Event** for this button will be received during the call, which typically sets an abort flag that can be tested on return and handled accordingly.

Note: This action must be used with care, because ALL user interactions with the application will be processed during the call. This includes clicks on tool bar or menu items, for instance, unless these are explicitly disabled, or if a modal dialog is currently on display (as in the above example).

Parameters

Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION PROCESS-EVENTS
GIVING #RESPONSE ↩
```

351 RECALC-LAYOUT Action

- Description 870
- Parameters 870

Description

Causes the control bar layout for the specified dialog to be recalculated, which may result in the bar positions and/or sizes being adjusted. The control bars include the newer tool-bar and status bar controls, but not the traditional tool bar and status bar available in earlier Natural versions.

Normally, Natural implicity causes a layout recalculation where necessary (e.g. immediately before the **After-Open Event** , or if a bar is created or moved). However, in order to reduce the number of screen refreshes to a minimum, it may be that only a delayed implicit layout recalculation is performed. Since some operations, such as getting the dialog's client area via the **INQ-INNER-RECT** Action, require that the bars are in their final positions, a preceding manually-triggered layout recalculation may be necessary in some cases. Note that this is not the case for all bars defined statically with the dialog editor, because Natural automatically does an immediate layout recalculation before calling the user's AFTER-OPEN event handler.

Note that an immediate layout recalculation will clear any outstanding delayed layout recalculation request. Also, multiple delayed layout recalculation requests are consolidated (i.e., do not cause multiple refreshes).

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of the dialog whose bar layout is to be recalculated.
Mode (L)	Input (optional parameter) If true (default), an immediate layout recalculation is performed. If false, a delayed layout recalculation is performed.

Example:

```
PROCESS GUI ACTION RECALC-LAYOUT WITH #DLG$WINDOW
GIVING #RESPONSE ↩
```

352 REFRESH-LINKS Action

- Description 872
- Parameters 872

Description

Refreshes the content of those input-field controls and selection-box controls for which the linked variable option was chosen for the content variables. This becomes necessary after these variables have been changed in code to display new values, for example, after a new record has been read from a database.

Note: If the content of such a dialog element is modified by the end user, the corresponding linked variable is updated automatically when the dialog element loses the focus.

Parameters

Name/Data Type	Explanation
HANDLE OF dialog	Input The dialog in which all linked variables are to be refreshed.
Response (I4)	Output Natural error (if applicable).

Example:

```
#MYVARI:= 'Refreshed'
PROCESS GUI ACTION REFRESH-LINKS WITH #DLG$WINDOW
GIVING #RESPONSE ↩
```

353 RESET-ATTRIBUTES Action

- Description 874
- Parameters 874

Description

Resets all attributes in the global attribute list to their default values. For the default values, see the section **Attributes**. It is recommended to use this action before creating dialog elements dynamically with the **ADD** Action, for example, in an after open event handler.

Parameters

Name/Data Type	Explanation
Response (I4)	Output Natural error (if applicable).

Example:

```
/* After open event handler code
PROCESS GUI ACTION RESET-ATTRIBUTES GIVING #RESPONSE
PROCESS GUI ACTION ADD...  ↩
```

354 SAVE-LAYOUT Action

- Description 876
- Parameters 876

Description

Saves the specified dialog's current control bar layout for the current user. The control bars include the newer tool-bar and status bar controls, but not the traditional tool bar and status bar available in earlier Natural versions. The layout includes the bar type (dockable or fixed), location, position, and the bar's visible and enabled state.

The information is currently stored in the registry under:

```
\\HKEY_CURRENT_USER\Software\Software AG\Natural Applications\<profile name>\<section ↩
name>
```

where <profile name> and <section name> are described in the table below. However, the base location (or whether the information is stored in the registry at all) may change for future Natural versions.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of the dialog for which the control bar layout is to be saved.
Profile name (A253)	Input (optional parameter) The name of the profile to which the information is to be saved. If this parameter is not specified, the name of the library containing the dialog is used.
Section name (A253)	Input (optional parameter) The name of the profile section to which the information is to be saved. If this parameter is not specified, the name of the dialog is used.

Example:

```
PROCESS GUI ACTION SAVE-LAYOUT WITH #DLG$WINDOW
GIVING #RESPONSE ↩
```

355 SET-ACCELERATION Action

- Description 878
- Parameters 878

Description

Defines or clears the acceleration for a spin control. The acceleration is the change in spin rate over time, when either one of the spin control arrow buttons is held down, or when one of the up or down arrow keys is held down whilst the spin control has the focus.

Spin controls have a default acceleration, which can be either removed or changed via this action.

Parameters

Name/Data Type	Explanation
HANDLE OF SPINCTRL	Input Handle of spin control for which the acceleration is to be set.
Delay (I4)	Input (repeating parameter) Delay (in seconds) before the associated increment is set. The Delay and Increment parameters are specified in pairs. The first delay value specified must be zero. Subsequent delay values must be specified in ascending order.
Increment (I4)	Input (repeating parameter) Increment size to apply after associated delay time has elapsed.
Response (I4)	Output Natural error (if applicable).

Example:

```
/* Set increment to 1 initially, to 3 after 5 seconds, and
/* to 10 after 15 seconds:
PROCESS GUI ACTION SET-ACCELERATION WITH #SPIN-1
  0 1 5 3 15 10 GIVING *ERROR
*
/* Remove any existing acceleration (increment always 1):
PROCESS GUI ACTION SET-ACCELERATION WITH #SPIN-1
  0 1 GIVING *ERROR
 ↩
```

356 SET-AUX-COLOR Action

- Description 880
- Parameters 880

Description

Sets an auxilliary color for a dialog element.

An auxilliary color is an additional color to those that can be selected via the standard **BACKGROUND-COLOUR-NAME**, **BACKGROUND-COLOUR-VALUE**, **FOREGROUND-COLOUR-NAME** and **FOREGROUND-COLOUR-VALUE** attributes (if available).

Parameters

Name/Data Type	Explanation
Control (HANDLE OF GUI)	Input Handle of dialog element to which the color is to be applied.
Part (I4)	Input Defines the part of the dialog element to be colored. The interpretation is specific to the type of dialog element (see below).
Color (I4)	Input Color name (one of the color constants defined in the local data area NGULKEY1).
Value (B3)	Input (optional parameter) Color value. Only used if color name is set to CUSTOM (=50, as defined in the local data area NGULKEY1).
Response (I4)	Output Natural error (if applicable).

The following auxilliary colors are currently available:

Dialog Element Type	Part	Interpretation
Date/Time Picker	1	Background color displayed between months in the drop-down month calendar (if present).
	2	Color used to display text within a month in the drop-down month calendar.
	3	Background color displayed in the drop-down month calendar's title.
	4	Color used to display text within the drop-down month calendar's title.
	5	Background color displayed within the month in the drop-down month calendar.

Dialog Element Type	Part	Interpretation
	6	Color used to display header day and trailing day text in the drop-down month calendar (if present). Header and trailing days are the days displayed that belong to the previous and following months, respectively.

Example:

```
/* Set date/time picker's title background to red
PROCESS GUI ACTION SET-AUX-COLOR WITH
  #DTP-1 3 RED GIVING *ERROR
 ↩
```

357 SET-AUX-FONT Action

- Description 884
- Parameters 884

Description

Sets an auxilliary font for a dialog element.

An auxilliary font is an additional font to those that can be selected via the **FONT-HANDLE** attribute (if available).

Parameters

Name/Data Type	Explanation
Control (HANDLE OF GUI)	Input Handle of dialog element to which the font is to be applied.
Part (I4)	Input Defines the font of the dialog element to be set. The interpretation is specific to the type of dialog element (see below).
Font (HANDLE OF FONT)	Input Handle of font to be set.
Response (I4)	Output Natural error (if applicable).

The following auxilliary fonts are currently available:

Dialog Element Type	Part	Interpretation
Date/Time Picker	1	Font used by the drop-down month calendar (if present).

Example:

```
/* Set date/time picker's calendar font
PROCESS GUI ACTION SET-AUX-FONT WITH
  #DTP-1 1 #DLG$WINDOW.FONT-HANDLE GIVING *ERROR
 ↩
```

358 SET-CLIENT-VALUE Action

- Description 886
- Parameters 886

Description

Sets or deletes user-defined ("client") information for a (dialog or) dialog element. Multiple items of information may be stored under different retrieval keys, although each call to this action can only create, update or delete a single value. The information is logically stored in the form of key/value pairs. If a value is deleted, the accompanying key is also removed. I f the dialog element is deleted, all associated client information (if any) is automatically deleted along with it.

The value is stored in the supplied format, without conversion. However, trailing blanks belonging to non-dynamic alphanumeric values are not stored, in order to save space.

Note that this is an alternative to setting the client information via the use of the **CLIENT-KEY** and **CLIENT-VALUE** attributes. Both mechanisms manipulate the same information internally, and their use may be intermixed. For example, information set by this action may be retrieved either via the **GET-CLIENT-VALUE** action or (in general) via the **CLIENT-VALUE** attribute. However, there are significant benefits to using this action in preference to the attribute-based approach. For more information, please refer to the article Storing and Retrieving Client Data for a Dialog Element.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of dialog or dialog element for which the client information is to be set or deleted.
Client value (any data type)	Input (optional parameter) If specified, this is the value to be set. If omitted, any existing value is deleted.
Client key (A253)	Input (optional parameter) If specified, this is the key, under which the specified value is to be stored. Any value currently stored under that key for the specified dialog or dialog element is replaced.If this parameter is omitted, the key (if any) specified by the **CLIENT-KEY** attribute of the specified dialog or dialog element is implicitly used.If the key is empty (i.e., all blank), the request is ignored.
Response (I4)	Output Natural error (if applicable).

Example:

```
/* Set value for (explicit) 'TIME' key for dialog to current time
PROCESS GUI ACTION SET-CLIENT-VALUE WITH #DLG$WINDOW *TIMX 'TIME'
  GIVING *ERROR

/* Set value for (implicit) 'DATE' key for dialog to current date
#DLG$WINDOW.CLIENT-KEY := 'DATE'
PROCESS GUI ACTION SET-CLIENT-VALUE WITH #DLG$WINDOW *DATX
  GIVING *ERROR

/* Delete value for (implicit) 'TIME' key for dialog
#DLG$WINDOW.CLIENT-KEY := 'TIME'
PROCESS GUI ACTION SET-CLIENT-VALUE WITH #DLG$WINDOW
  GIVING *ERROR


/* Delete value for (explicit) 'DATE' key for dialog
PROCESS GUI ACTION SET-CLIENT-VALUE WITH #DLG$WINDOW 1X 'DATE'
  GIVING *ERROR ↩
```

359 SET-CLIPBOARD-DATA Action

- Description 890
- Parameters 891

Description

Places data on the local clipboard in the specified Natural data format. If the local clipboard already contains some data in this format, the data is replaced by this call.

Natural supports both pre-defined and private data formats (see below). The advantage of using predefined formats is that they are recognized and understood by many other Windows applications. Pre-defined formats are represented by numeric strings, for which symbolic constants have been defined in the local data area NGULKEY1. For example, the format CF-TEXT indicates standard text format, where the data consists of lines of text separated by carriage return/line feed character pairs. Likewise, the CF-FILELIST format indicates that the data consists of file and/or directory path names, separated by null-terminators (characters with value 0), with an extra trailing null-terminator used to designate the end of the list. Such a file list corresponds to the standard clipboard format used by the Windows Explorer, and many other applications supporting file drag and drop, such as Natural itself. Note that Natural automatically inserts the required delimiters (carriage return/line feed or null-terminator) between the data items between each field (scalar or array element) it writes, as well as the trailer (if any) after the last data item, depending on the format. However, if you use a single large field (e.g. a dynamic alpha variable) to write multiple items (lines of text or file names), you must specify the delimiters between the data items explicitly. Note that if array ranges are used, the array range is implicitly expanded internally and handled as if each array element within the range had been explicitly specified in turn (see example below). Note that, for CF-TEXT and CF-FILELIST formats, the incoming data is automatically converted to alpha format in a manner compatible with the MOVE statement if the data is not already in alpha format.

Private formats are more flexible, but are only understood by Natural. In this case, a copy of the data is stored, without any data conversion, except that any trailing blanks belonging to any alpha operands are removed to save space. Private formats are represented by any string that does not begin with a digit. Both the source and target simply need to know the name of the format to use it in the SET-CLIPBOARD-DATA and **GET-CLIPBOARD-DATA** actions, respectively. Note that, although the same internal data structure is used for all private formats, the ability to use multiple private formats prevents the data from being interpreted incorrectly. For example, you may wish to transfer both ActiveX tree view node data and list box item data in private format. By specifying distinct formats in each case (e.g. 'TVNODEDATA' and 'LBITEMDATA'), you can conveniently prevent the list box from attempting to interpret the tree view data, and vice versa. To decide whether or not a paste or drop operation may occur, each control needs to simply pass its respective format string to the **INQ-FORMAT-AVAILABLE** action, and does not need to inspect the data itsef.

Note that although the local clipboard can contain information in multiple formats at any one time, the data for each format must be written via separate calls to SET-CLIPBOARD-DATA. Furthermore, the data for each format must be specified in a single call to SET-CLIPBOARD-DATA. It is not possible to build up the data for each format bit-by-bit by using separate calls to this action.

The clipboard must be open in order to use this action. If you are using this action to prepare drag-drop data in the **BEGIN-DRAG** event, you do not need to explicitly open the clipboard, as this has already been done implicitly by Natural. Otherwise (e.g., if preparing data for the Windows clipboard) you need to precede this action with an explicit call to **OPEN-CLIPBOARD**.

Note that this action does not write data to the Windows clipboard. The data will only be available on the Windows clipboard (and hence visible to other applications) after you issue a subsequent call to the **CLOSE-CLIPBOARD** action. This is, however, not necessary for drag-drop operations, because the **GET-CLIPBOARD-DATA** action used by the drop target works preferentially directly on the drag-drop clipboard (the local clipboard in the source process) in this case.

Parameters

Name/Data Type	Explanation
Format (A253)	Input Clipboard Natural data format. Can be standard numerc string for predefined formats (e.g., CF-TEXT) or user-defined string for private formats.
Data (List of any type except handles)	Input Data to be written. Can consist of any number of non-handle scalar and/or array operands, including array index ranges and multidimensional arrays. Dynamic alpha variables are also supported.
Response (I4)	Output Natural error (if applicable).

Example:

```
DEFINE DATA LOCAL
1 #DYN (A) DYNAMIC
1 #ARR (A20/3) INIT<'Line 1', 'Line 2', 'Line 3'>
1 #RESPONSE (I4)
END-DEFINE
*
* NOTE: All three calls to SET-CLIPBOARD-DATA below have the same effect!
*
/* Example 1 - Items specified as individual fields
PROCESS GUI ACTION SET-CLIPBOARD-DATA WITH CF-TEXT #ARR(1) #ARR(2) #ARR(3)
GIVING #RESPONSE
*
/* Example 2 - Items specified as array index range
PROCESS GUI ACTION SET-CLIPBOARD-DATA WITH CF-TEXT #ARR(*)
  GIVING #RESPONSE
```

```
*
/* Example 3 - Items specified using dynamic variable/* (note the use of the explicit ↩
delimiter)
COMPRESS #ARR(1) END-OF-LINE #ARR(2) END-OF-LINE #ARR(3) INTO #DYN
  LEAVING NO SPACE
PROCESS GUI ACTION SET-CLIPBOARD-DATA WITH CF-TEXT #DYN #ARR(3)
  GIVING #RESPONSE

 ↩
```

360 SET-FOCUS Action

- Description 894
- Parameters 894

Description

Selects a dialog element so that the end user can enter data with the keyboard. You can use this action with handles of the following dialog elements: **Bitmap Control**, **Edit Area Control**, **Input Field Control**, **List Box Control**, **Push Button Control**, **Radio Button Control**, **Scrollbar Control**, **Selection Box Control**, **Toggle Button Control**. You can also use this action with a dialog's window handle, for example, with "#DLG$WINDOW".

Note: It is not recommended to execute the SET-FOCUS action from the leave event, because the leave event implies that the focus is to be set to yet another part of the application.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Any dialog element of the above-mentioned type that is supposed to get the focus.
Response (I4)	Output Natural error (if applicable).

Example:

```
#INPUT := #IF-1
PROCESS GUI ACTION SET-FOCUS WITH #INPUT
GIVING #RESPONSE ↩
```

361 SET-SUBITEM-DATA Action

- Description 896
- Parameters 896

Description

Updates the ("subitem") data for the specified list view item for the specified list view column(s).

The list view item handle may be specified as NULL-HANDLE, in which case the default data for the specified column(s) will be updated. The default data is the data returned by the **GET-SUBITEM-DATA** action in the case where no subitem data is present.

The column handle and data parameters may be repeated (in pairs), in order to allow the data for multiple columns to be updated in a single action.

The supplied data will be converted (if necessary) to the column's data **FORMAT** , with which it must therefore be MOVE -compatible.

For more information, please refer to the article Working with List View Controls.

Parameters

Name/Data Type	Explanation
Item (HANDLE OF GUI)	Input Handle of list view item to be updated, or NULL-HANDLE if the default data for the specified list view column(s) should be updated.
Column (HANDLE OF GUI)	Input Handle of list view column to be updated. Cannot be specified as NULL-HANDLE.
Data (any data type)	Input Data to be written.
Response (I4)	Output Natural error (if applicable).

Example:

```
/* Example 1 - Updating of data for two columns in one action
PROCESS GUI ACTION SET-SUBITEM-DATA WITH #LVITEM-1
  #LVCOL-1 123 #LVCOL-2 'London' GIVING *ERROR
*
/* Example 2 - Set default data for column 3 (to -1)
PROCESS GUI ACTION SET-SUBITEM-DATA WITH NULL-HANDLE
  #LVCOL-3 -1 GIVING *ERROR
 ↩
```

362 SET-TABS Action

- Description 898
- Parameters 898

Description

Sets tabulator stops in a **List Box Control** even if the font in the **List Box Control** is proportional. The tabulator stops apply to the **STRING** attribute in any item of the **List Box Control**.

Parameters

Name/Data Type	Explanation
HANDLE OF LISTBOX	Input Handle of the **List Box Control**.
Tabstop (I4)	Input The tabstop is measured in units of one quarter of the system font's average width. You can enter this parameter any number of times (in ascending order). In the list box items' STRING, you determine the place of the tabstop by inserting the HORIZONTAL-TAB character (defined in the local data area NGULKEY1). Inserting this character will move the following STRING text to the next tabstop.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION SET-TABS WITH #LB-1 50 100 150 GIVING #RESPONSE
COMPRESS 'HELLO' HORIZONTAL-TAB 'HUGO' TO #STRING
PROCESS GUI ACTION ADD-ITEMS WITH #LB-1 1 #STRING  ↩
```

363 SET-TEXT Action

- Description 900
- Parameters 900

Description

Sets the text to be associated with the specified dialog or dialog element. For example, the contents of an edit area or the caption of a window or button.

This action sets the same information as a update of the dialog or dialog element's **STRING** attribute, if available. However, unlike the use of the attribute, this action allows text in excess of 253 characters to be set.

Note that this action can only be used to set text belonging to dialog elements that are windows or controls. For example, it can be used on a dialog or pushbutton, but not to set the text of a list box item, even though the latter has a **STRING** attribute.

The receiving field may be a dynamic variable, allowing any trailing blanks in the text to be preserved.

Parameters

Name/Data Type	Explanation
Object (HANDLE OF GUI)	Input Handle of dialog or dialog element whose text is to be set.
Text (A)	Input Text to be associated with specified dialog or dialog element. **Note:** If this parameter is a dynamic or large alpha variable, it is possible to set text longer than 253 characters.
Response (I4)	Output Natural error (if applicable).

Example:

```
DEFINE DATA LOCAL
1 #CONTROL HANDLE OF GUI
END-DEFINE
*
PROCESS GUI ACTION SET-TEXT WITH #CONTROL
  'Caption' GIVING *ERROR ↩
```

364 SET-TICKS Action

- Description 902
- Parameters 902

Description

Sets manual tick marks for a slider control.

This action cannot be used if the control has the "Auto ticks (a)" **STYLE**, since the tick marks are added automatically in this case.

Note that it does not make sense to add tick marks at the beginning or end of the slider's range, since these tick marks are automatically created by the control.

Tick marks are only shown if the control has either (or both) of the "Side 1 ticks (1)" or "Side 2 ticks (2)" **STYLE** flags set.

To clear the manually-added tick marks, use the **CLEAR-TICKS** action.

Parameters

Name/Data Type	Explanation
HANDLE OF SLIDER	Input Handle of slider to which the tick marks are to be applied.
Ticks (List of I4)	Input Tick position(s) to set. Can be a scalar or an array index range (see example below). This parameter may be repeated, allowing any combination of scalars and array index ranges.
Response (I4)	Output Natural error (if applicable).

Example:

```
DEFINE DATA LOCAL
1 #TICKS (I4/3) CONST <3, 5, 7>
END-DEFINE
*
/* The following three calls have the same effect:
PROCESS GUI ACTION SET-TICKS WITH #SLIDER-1
  #TICKS(1) #TICKS(2) #TICKS(3) GIVING *ERROR
*
PROCESS GUI ACTION SET-TICKS WITH #SLIDER-1
  #TICKS(*) GIVING *ERROR
*
```

```
PROCESS GUI ACTION SET-TICKS WITH #SLIDER-1
  3 #TICKS(2:3) GIVING *ERROR
 ↩
```

365 SET-TIME-RANGE Action

- Description 906
- Parameters 906

Description

Sets the upper and/or lower date/time limit for a date and time picker control.

Parameters

Name/Data Type	Explanation
HANDLE OF DATETIMEPICKER	Input Handle of date and time picker control to which the date and/or time range is to be applied.
Lower (T)	Lower (T) Input Lower date/time limit. If omitted, no lower limit will be explicitly set.
Upper (T)	Input Upper date/time limit. If omitted, no upper limit will be explicitly set.
Response (I4)	Output Natural error (if applicable).

Example:

```
/* Set upper limit to today
PROCESS GUI ACTION SET-TIME-RANGE WITH
  #DTP-1 1X *DATX GIVING *ERROR
 ↩
```

366 SHOW-CONTEXT-MENU Action

- Description 908
- Parameters 908

Description

Manually invokes a context menu.

Note: This is an alternative to the preferred automatic context menu invocation based on the use of the **CONTEXT-MENU** attribute.

For more information, please refer to the article Defining and Using Context Menus.

Parameters

Name/Data Type	Explanation
HANDLE OF CONTEXTMENU	Input Handle of the **CONTEXT-MENU**.
HANDLE OF GUI	Input (optional parameter) The handle of the dialog or dialog element hosting the context menu, to which the (supplied or implicit) coordinates are relative. The context menu's **CONTROL** attribute will be set to this value. If not specified, the value of NULL-HANDLE is assumed, implying that the coordinates are screen-relative.
X-Position (I4)	Input (optional parameter) X-axis position at which the context menu is to be displayed, relative to the interior (client) area of the host dialog or dialog element, if any, or the screen position otherwise. If not specified, the value 0 is assumed.
Y-Position (I4)	Input (optional parameter) Y-axis position at which the context menu is to be displayed, relative to the interior (client) area of the host dialog or dialog element, if any, or the screen position otherwise. If not specified, the value 0 is assumed.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION SHOW-CONTEXT-MENU WITH #CTXMENU-1 GIVING *ERROR ↩
```

367 SORT-ITEMS Action

- Description 912
- Parameters 912

Description

Sorts the specified list view control or column, or the child items of the specified tree view control or item. For list views in either of the icon modes, the items are re-arranged.

Note that after this action has been called, new items will no longer be automatically inserted in sorted sequence, even if the appropriate **SORTED** attribute value is set to TRUE.

Parameters

Name/Data Type	Explanation
Object (HANDLE OF LISTVIEW, HANDLE OF LISTVIEWCOLUMN, HANDLE OF TREEVIEW or HANDLE OF TREEVIEWITEM)	Input Handle of list view control or item, or tree view control or item, that is to be sorted. **Note:** If a list view control handle is specified, and one or more list view columns have been defined, the sort is performed as if the handle of the primary column had been specified.
Descending (L)	Input (optional parameter) Sort direction. FALSE=ascending sequence (default); TRUE=descending sequence.
Recurse (L)	Input (optional parameter) Only applicable in case of tree view control or item. FALSE=only direct child items are to be sorted (default); TRUE=sort extended to also cover indirect child items.
Flags (I4)	Input (optional parameter) Sort flags. May be any combination of the following values (as specified by passing the sum of the individual flag values): 1=Case-insensitive sort 2=Use word sort instead of string sort (e.g., "co-ordinate" = "coordinate").

Name/Data Type	Explanation
	For list view controls with columns, the default value (if this parameter is omitted) is based on the corresponding STYLE attribute flags of the column being sorted. Otherwise the default is 0 (case-sensitive string sort).
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION SORT-ITEMS WITH #LV-1 GIVING *ERROR ↩
```

368 SYSTEM-GET-NATIVE-HANDLE Action

- Description 916
- Parameters 916

Description

Queries the native handle of a dialog element in terms of the windowing system. You can use this native handle for external function calls.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input The handle of the dialog element in Natural terms.
Native handle (I4)	Output The window handle of the dialog element in terms of the windowing system. For font controls, the native font handle is returned. For menu bars, menu items and submenu controls, the native menu handle is returned. For bitmap controls, the native bitmap handle is returned. For all other dialogs and dialog elements, the native window handle is returned.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION SYSTEM-GET-NATIVE-HANDLE WITH #PB-1 #NATIVEHANDLE GIVING
#RESPONSE
#IF-1.STRING := #NATIVEHANDLE  /* Display the handle value  ↩
```

369 SYSTEM-PRINTER-SETUP Action

- Description 918
- Parameters 918

Description

Invokes a dialog box that allows the end user to assign a physical printer to a logical device name. You can assign this logical device name to a report using the DEFINE PRINTER statement.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input The parent dialog element of the printer assignment dialog box.
Logical device (A8)	Input/Output The logical device name to which the printer is assigned. Possible values: "LPT1" to "LPT31", "lpt1" to "lpt31".
Commit (L)	Output (optional parameter) Returns TRUE if the user closed the dialog box via the 'OK' push button, or FALSE if the dialog was cancelled or if an error occurred.
Response (I4)	Output Natural error (if applicable).

Note: The first two parameters are mandatory.

Example:

```
PROCESS GUI ACTION SYSTEM-PRINTER-SETUP WITH #PB-1 'LPT1'
GIVING #RESPONSE ↩
```

370 TABLE-DELETE-ROW Action

- Description 920
- Parameters 920

Description

Deletes a row in a table. To do so, one of the table control's **STYLE** attribute values must be "e". Specifying the value "0" in the Row index parameter is useful in combination with the delete-row event because the **ROW** attribute then contains the index of the row in which the end user has caused the most recent event.

Parameters

Name/Data Type	Explanation
HANDLE OF TABLE	Input Specifies a **Table Control**.
Row Index (I4)	Input Specifies the number of the row to be deleted. If you specify "0", the value of the table's **ROW** attribute is taken as the number of the row to be deleted. If you specify "-1", the last row in the table will be deleted. If you specify "1", the first row in the table will be deleted.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION TABLE-DELETE-ROW WITH #TBL-1 2 GIVING #RESPONSE
...

PROCESS GUI ACTION TABLE-REFRESH WITH #TBL-1 GIVING #RESPONSE /* Recommended  ↩
```

371 TABLE-FIND-FIELD Action

- Description 922
- Parameters 922

Description

Finds the column with the specified title.

Parameters

Name/Data Type	Explanation
HANDLE OF TABLE	Input Specifies a **Table Control**.
Field name (A253)	Input The column title to be found.
Index (I4)	Output The index of the column where the title was found.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION TABLE-FIND-FIELD WITH #TBL-1 FLD1 #IDX
GIVING #RESPONSE ↩
```

372 TABLE-GET-SELECTION Action

- Description 924
- Parameters 924

Description

Retrieves the column and row indices of the selected cells in a **Table Control** from the beginning of the first selected cell to the end of the last selected cell.

Parameters

Name/Data Type	Explanation
HANDLE OF TABLE	Input Specifies a **Table Control**.
Column from (I4)	Output First selected...
Row from (I4)	Output ...cell.
Column to (I4)	Output Last selected...
Row to (I4)	Output ...cell.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION TABLE-GET-SELECTION WITH #TBL-1 #COLFIRST #ROWFIRST #COLLAST
#ROWLAST GIVING #RESPONSE  ↩
```

373 TABLE-INQUIRE-CELL Action

- Description 926
- Parameters 926

Description

enables you to inquire which cell is at a particular pixel offset within the table. This is particularly useful for determining the cell for which a context menu was invoked, in which case the position specified should be that returned by the **INQ-CLICKPOSITION** Action used within the context menu's **BEFORE-OPEN** Event.

Parameters

Name/Data Type	Explanation
HANDLE OF TABLE	Input Specifies a **Table Control**.
X-Position (I4)	Input Specifies the x-axis position in pixels relative to the top-left hand corner of the table.
Y-Position (I4)	Input Specifies the y-axis position in pixels relative to the top-left hand corner of the table.
Row Index (I4)	Output The row corresponding to the specified y-axis position, or 0 if outside the range of any row.
Column Index (I4)	Output The column corresponding to the specified x-axis position, or 0 if outside the range of any column.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION TABLE-INQUIRE-CELL WITH #TBL-1 #X-POSITION #Y-POSITION #ROW #COLUMN
GIVING #RESPONSE
 ↩
```

374 TABLE-INQUIRE-ROW Action

- Description 928
- Parameters 928

Description

enables you to inquire the value in each cell of a single row. The values will update the input parameters you specify. Later at runtime, you can retrieve the updated parameter values. Specifying the value "0" in the `Row Index` parameter is useful in combination with the **Insert-Row** Event because the **ROW** attribute then contains the index of the row in which the end user has caused the most recent event.

Parameters

Name/Data Type	Explanation
HANDLE OF TABLE	Input Specifies a **Table Control**.
Row Index (I4)	Input Specifies the number of the row to be inquired. If you specify "0", the value of the table's **ROW** attribute is taken as the number of the row to be inquired. If you specify "-1", the last row in the table will be inquired. If you specify "1", the first row in the table will be inquired.
List of parameters	Input You can either specify one parameter for each column of the row, or you can specify less parameters than there are columns. If you specify more parameters than there are columns, an error message is returned.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION TABLE-INQUIRE-ROW WITH #TBL-1 3 #P1 #P2 #P3 #P4
GIVING #RESPONSE
```

375 TABLE-INSERT-ROW Action

- Description 930
- Parameters 930

Description

Inserts a row into a table. To do so, one of the table control's **STYLE** attribute values must be "e". Specifying the value "0" in the `Row Index` parameter is useful in combination with the insert-row event because the **ROW** attribute then contains the index of the row in which the end user has caused the most recent event.

Parameters

Name/Data Type	Explanation
HANDLE OF TABLE	Input Specifies a **Table Control**.
Row Index (I4)	Input Specifies the number of the row before which the new row is to be inserted. If you specify "0", the value of the table's **ROW** attribute is taken as the number of the row before which the new row is to be inserted. If you specify "-1", the new row will be appended to the lat row in the table. If you specify "1", the row to be inserted will become the first row in the table.
List of parameters	Input You can either specify one parameter for each column of the row, or you can specify less parameters than there are columns. If you specify more parameters than there are columns, an error message is returned.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION TABLE-INSERT-ROW WITH #TBL-1 3 #P1 #P2 #P3 #P4
GIVING #RESPONSE
PROCESS GUI ACTION TABLE-REFRESH WITH #TBL-1 GIVING #RESPONSE /* Recommended
```

376 TABLE-REFRESH Action

- Description 932
- Parameters 932

Description

Refreshes the contents of a **Table Control**.

Parameters

Name/Data Type	Explanation
HANDLE OF TABLE	Input Specifies a **Table Control**.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION TABLE-REFRESH WITH #TBL-1
GIVING #RESPONSE ↩
```

377 TABLE-SET-SELECTION Action

- Description 934
- Parameters 934

Description

Selects a rectangular range of cells in a **Table Control**.

Parameters

Name/Data Type	Explanation
HANDLE OF TABLE	Input Specifies a **Table Control**.
Column from (I4)	Input First selected...
Row from (I4)	Input ...cell.
Column to (I4)	Input Last selected...
Row to (I4)	Input ...cell. If you specify "0", the selection will extend to the last cell in the table.
Response (I4)	Output Natural error (if applicable).

Example:

```
PROCESS GUI ACTION TABLE-SET-SELECTION WITH #TBL-1 #COLFIRST #ROWFIRST #COLLAST ↩
#ROWLAST
GIVING #RESPONSE ↩
```

378 TEXT-GET-EXTENT Action

- Description 936
- Parameters 936

Description

Returns the size of a text if a certain font is chosen. You can use this action, for example, to find out the best font that still has an acceptable width and height.

Parameters

Name/Data Type	Explanation
HANDLE OF FONT	Input Specifies a **Font Control**.
Text (A)	Input The text to be measured. **Note:** If this parameter is a dynamic or large alpha variable, it is possible to measure text longer than 253 characters.
Width (I4)	Output Width in pixels.
Height (I4)	Output Height in pixels.
Response (I4)	Output Natural error (if applicable).

Example:

```
/*Check the size a font will require
#TEXT := 'Check the size'
PROCESS GUI ACTION TEXT-GET-EXTENT WITH FONT-HANDLE #TEXT #WIDTH #HEIGHT
GIVING #RESPONSE
```

379 UPDATE-COMMAND-STATUS Action

- Description 938
- Parameters 938

Description

A dialog automatically receives command-status events (unless suppressed) during idle processing when the status of one or more commands (signals, or menu items or tool bar items without a **SAME-AS** attribute) need to be updated. This action, however, allows the same updating logic to be triggered manually at any time, in addition to the automatic idle-time update.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of the dialog or dialog element whose status is to be updated, or NULL-HANDLE. If a dialog handle is specified, all commands belonging to the specified dialog (excluding submenus and context menus) are included in the updating process. If a dialog element handle is specified, all commands belonging to the specified dialog element are included in the updating process. If NULL-HANDLE is specified, all commands in all dialogs (excluding submenus and context menus) are included in the updating process, as is the case for the implicit idle-time update.

Example:

```
PROCESS GUI ACTION UPDATE-COMMAND-STATUS WITH NULL-HANDLE
GIVING #RESPONSE ↩
```

380 VALIDATE Action

- Description 940
- Parameters 940

Description

Validates a handle variable. A handle variable is considered valid if it contains either the value NULL-HANDLE or a handle to an existing dialog or dialog element.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Any dialog or dialog element.
Response (I4)	Output Natural error (if applicable), or 0 if handle variable is valid.

Example:

```
PROCESS GUI ACTION VALIDATE WITH #PB-1
GIVING #RESPONSE ↩
```

V NGU Subprograms and Dialogs

General Information
NGU-CLIENT-ADVISE-HOT Subprogram
NGU-CLIENT-ADVISE-TERM Subprogram
NGU-CLIENT-ADVISE-WARM Subprogram
NGU-CLIENT-CONNECT Subprogram
NGU-CLIENT-DISCONNECT Subprogram
NGU-CLIENT-EXECUTE Subprogram
NGU-CLIENT-GET-DATA Subprogram
NGU-CLIENT-POKE Subprogram
NGU-CLIENT-REQUEST Subprogram
NGU-CLIENT-STOP Subprogram
NGU-COLOUR-SELECT Dialog
NGU-DIALOG-CLOSE-ALL Subprogram and Subroutine
NGU-FONT-SELECT Dialog
NGU-MESSAGEBOX Dialog
NGU-SERVER-DATA Subprogram
NGU-SERVER-GET-DATA Subprogram
NGU-SERVER-REGISTER Subprogram
NGU-SERVER-STOP Subprogram
NGU-SERVER-UNREGISTER Subprogram
NGU-SERVER-WAIT Subprogram
NGULKEY1 Reserved Symbols

381 General Information

The NGU-prefixed subprograms and dialogs in library SYSTEM provide you with frequently needed functionality. You then only specify a CALLNAT or OPEN DIALOG statement in your event handler code, instead of having to program everything manually.

For your convenience, the local data areas NGULKEY1 and NGULFCT1 are automatically included in the list of local data areas used by any new dialog.

- NGULFCT1 lists the names of the subprograms and dialogs as such.
- NGULKEY1 lists reserved symbols to be used in any event handler code. This enables you to refer to certain attribute values by the more meaningful texts rather than by the integer values. It also enables you to use meaningful dialog element names as parameters in a CALLNAT to an NGU-prefixed subprogram or in an OPEN DIALOG to an NGU-prefixed dialog.

There are two types of names for the NGULFCT1 subprograms or dialogs:

- Long Name (explanatory) is the basis of the alphabetical list.
- Natural Object Name is the shorter name.

To call the subprograms, you can use either name in event handler code: you specify CALLNAT *subprogram-long-name/Natural-object-name parameter-name...*

To call the dialogs, you can also use both types of names inside the OPEN DIALOG statement.

Note: The NGU subprograms and dialogs provided with Natural Version 2.1.3 have partly been replaced by corresponding PROCESS GUI statement actions. The old NGU subprograms and dialogs are compatible with this version. It is recommended that you replace the calls to the dialogs and subprograms with the corresponding PROCESS GUI statement actions. This will increase the performance of your applications.

382 NGU-CLIENT-ADVISE-HOT Subprogram

- Natural Object Name 946
- Parameters 946

Natural Object Name

NGUCLADH

Description

Requests the server identified by DDE-VIEW.CONV-ID to advise the client as soon as new data are available. If so, the client will be advised of the new data. This will be specified in DDE-VIEW.ITEM. The data will then be sent automatically.

Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

383 NGU-CLIENT-ADVISE-TERM Subprogram

- Natural Object Name 948
- Description 948
- Parameters 948

Natural Object Name

NGUCLADT

Description

Requests the server to no longer advise the client of new data.

Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

384 NGU-CLIENT-ADVISE-WARM Subprogram

- Natural Object Name 950
- Description 950
- Parameters 950

Natural Object Name

NGUCLADW

Description

Requests the server identified by DDE-VIEW.CONV-ID to advise the client as soon as new data are available. If so, the client will only be advised of the new data. This will be specified in DDE-VIEW.ITEM. If the application wants the data, it will have to request the data.

Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

385 NGU-CLIENT-CONNECT Subprogram

- Natural Object Name 952
- Description 952
- Parameters 952

Natural Object Name

NGUCLCON

Description

This subprogram requests a conversation with a server providing the service DDE-VIEW.SERVICE on the topic DDE-VIEW.TOPIC. If an appropriate server can be found, this starts a conversation identified by a unique conversation ID, which is returned to DDE-VIEW.CONV-ID. If not, this field is empty on return of the request. If you plan on having more than one conversation, store the value of DDE-VIEW.CONV-ID as you will need it to identify conversations.

Before you execute this subprogram, you must assign the *DIALOG-ID system variable of the client dialog to DDE-VIEW.CONV-ID. You must also assign the string "DLGID" to DDE-VIEW.MESSAGE.

Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

386 NGU-CLIENT-DISCONNECT Subprogram

- Natural Object Name 954
- Description 954
- Parameters 954

Natural Object Name

NGUCLDIS

Description

Closes the conversation identified by DDE-VIEW.CONV-ID. The server will be advised of this.

Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

387 NGU-CLIENT-EXECUTE Subprogram

- Natural Object Name 956
- Description 956
- Parameters 956

Natural Object Name

NGUCLEXE

Description

Requests the server identified by DDE-VIEW.CONV-ID to execute a command. The field DDE-VIEW.DATA describes the command to be executed.

Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

388 NGU-CLIENT-GET-DATA Subprogram

- Natural Object Name 958
- Description 958
- Parameters 958

Natural Object Name

NGUCLGDA

Description

You must use this subprogram in the event handler of a DDE-client event to fill the DDE-VIEW parameter data area. You must then ensure that your event handler code acts on the value of DDE-VIEW.MESSAGE. The possible values of DDE-VIEW.MESSAGE are:

DISCONNECT	The server has terminated the conversation identified by DDE-VIEW.CONV-ID by unregistering the topic, or by exiting.
DATA	The server has sent data for DDE-VIEW.CONV-ID and DDE-VIEW.ITEM. The data are contained in DDE-VIEW.DATA. The length of the data will be contained in DDE-VIEW.DATALEN.
NOTIFY	The server has new data available for the conversation identified by DDE-VIEW.CONV-ID and DDE-VIEW.ITEM. The data are not sent at this point. You must request these data in your event handler. Only then the data will actually be received.
TIMEOUT	No server message has been received during the timeout period.

When DDE-VIEW.MESSAGE has the value DISCONNECT, DATA, or NOTIFY, the fields DDE-VIEW.SERVICE and DDE-VIEW.TOPIC are also set appropriately.

Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.

Name/Data Type	Explanation
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

389 NGU-CLIENT-POKE Subprogram

- Natural Object Name 962
- Description 962
- Parameters 962

Natural Object Name

NGUCLPOK

Description

Sends data to the server identified by DDE-VIEW.CONV-ID. These data will be specified in DDE-VIEW.ITEM. When sending data to the server, you must set DDE-VIEW.DATALEN to a value:

- If set to "0", the entire data will be sent; the server will send on return a value that indicates how many data were actually sent.
- If set to "-1", trailing blanks will be removed and the server will send on return a value that indicates how many data were actually sent.
- If set to a value greater zero, this amount of data is actually sent.

The data to be sent are contained in DDE-VIEW.DATA.

Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

390 NGU-CLIENT-REQUEST Subprogram

- Natural Object Name 964
- Description 964
- Parameters 964

Natural Object Name

NGUCLREQ

Description

Requests data from the server identified by DDE-VIEW.CONV-ID. These data will be specified in DDE-VIEW.ITEM. If data are available, the server will send them to the client.

Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

391 NGU-CLIENT-STOP Subprogram

- Natural Object Name 966
- Description 966
- Parameters 966

Natural Object Name

NGUCLSTP

Description

Terminates all DDE activity for the client. Any open conversations are terminated. This subprogram may be useful to handle an error returned from a client request.

Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

392 NGU-COLOUR-SELECT Dialog

- Natural Object Name 968
- Description 968
- OPEN DIALOG Parameters 968

Natural Object Name

NGUCOSE1

Description

Provides a dialog box for selecting or creating a color.

OPEN DIALOG Parameters

Name/Data Type	Explanation
HANDLE OF parent	USING clause/Input The parent dialog.
Colour name (I4)	WITH clause/Input/Output Input: Name of color to be pre-selected in dialog. If this is set to DEFAULT(0), black will be pre-selected. Output: Name of selected color (if any). See **BACKGOUND-COLOUR-NAME** for a list of possible color names.
Colour value (B3)	WITH clause/Input /Output Input: RGB color value to be pre-selected in dialog (only used if color name is set to CUSTOM (50)). Output: RGB value of selected color.
Colour selected (L)	WITH clause/Output Indicates whether a color has been selected.
Style (A32)	WITH clause/Input (optional parameter) Reserved for future use.

Example:

```
OPEN DIALOG NGU-COLOUR-SELECT #DLG$WINDOW WITH
#DLG$WINDOW.BACKGROUND-COLOUR-NAME
#DLG$WINDOW.BACKGROUND-COLOUR-VALUE
#COLOUR-SEL
```

393 NGU-DIALOG-CLOSE-ALL Subprogram and Subroutine

- Natural Object Names 970
- Description 970
- Parameters 970

Natural Object Names

NGUDICL1 NGUDICLS

Description

Both the subprogram and the subroutine close all child dialogs of a given dialog. If you execute them, the close event occurs for all child dialogs. You can use them, for example, to close all MDI child dialogs in an MDI application. The difference between the two is that you use the NGUDICLS subroutine if you are accessing a global data area while closing all child dialogs. NGUDICLS is to be found in library SYSNGU. If the dialog calling this subprogram/subroutine is an MDI child dialog itself and all MDI child dialogs are to be closed, you must specify the parent of this dialog (the MDI frame dialog) as parameter. The MDI child dialog calling this subprogram/subroutine will then be the last to be closed.

Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of the dialog whose child dialogs are to be closed.
Response (I4)	Output Natural error (if applicable).

Examples:

```
CALLNAT NGU-DIALOG-CLOSE-ALL #MYDIA #RESPONSE /* NGUDICL1 subprogram
PERFORM NGU-DIALOG-CLOSE-ALL #MYDIA #RESPONSE /* NGUDICLS subroutine
```

394 NGU-FONT-SELECT Dialog

- Natural Object Name 972
- Description 972
- OPEN DIALOG Parameters 972

Natural Object Name

NGUFOSE1

Description

Provides the end user with a dialog box to select a font. To avoid problems, always ensure that you assign NULL-HANDLE to your font before you specify the OPEN DIALOG statement. Otherwise, an invalid font handle is assigned to the text(s) for which the end user has already selected a font.

OPEN DIALOG Parameters

Name/Data Type	Explanation
HANDLE OF GUI	USING clause/Input The parent dialog of the font selection dialog box.
HANDLE OF FONT	WITH clause/Output Returns a font.
Font selected (L)	WITH clause/Output Indicates whether a font has been selected.
Font string (A253)	WITH clause/Input/Output Input: preselected font, Output: font selected by end user.

Example:

```
#FONT-NEW:= NULL-HANDLE /*Recommended every time you use this dialog
OPEN DIALOG NGU-FONT-SELECT
     USING #DLG$WINDOW
     GIVING #DLG
     WITH #FONT-NEW #FONT-SEL #FONT-STRING
```

395 NGU-MESSAGEBOX Dialog

- Natural Object Name 974
- Description 974
- OPEN DIALOG Parameters 974
- Separator Keyword 975

Natural Object Name

NGUMBDI1

Description

Displays a message box, for example with the title: "Dialog editor" and the message "Information message box" with an "OK" style push button that is selected by default. Please note that such a message box is modal (the end user can only leave it with a choice).

OPEN DIALOG Parameters

Name/Data Type	Explanation
HANDLE OF GUI	USING clause/Input The parent dialog of the message box.
Button (A1)	WITH clause/Output Returns the selected button (for possible output values, see below).
Message (A253)	WITH clause/Input (BY VALUE) Here you specify the message to be issued.
Title (A50)	WITH clause/Input (BY VALUE) Here you specify the message box title.
Style (A32)	WITH clause/Input (BY VALUE) The type of message box (for possible input values, see below).

"Button" Value	Selected Button
O	OK push button.
C	Cancel push button.
Y	Yes push button.
N	No push button.
R	Retry push button.

"Style" Value	Message Box Type
I	blue lower-case "i" in a round icon, Information Provides information about the results of a command. Offers no user choices; the user acknowledges the message by clicking the OK button.
!	exclamation mark, Warning Alerts the user to a condition or situation that requires the user's decision and input before proceeding, such as an impending action with potentially destructive, irreversible consequences.

"Style" Value	Message Box Type
S	stop sign, Critical Informs the user of a serious problem that requires intervention or correction before work can continue.
?	This sign is no longer recommended as it does not clearly represent a type of message. The system continues to support its inclusion only for backward compatibility.
may be combined with:	
O	OK push button (default).
OC	OK and Cancel push buttons.
YNC	Yes, No, and Cancel push buttons.
YN	Yes and No push buttons.
RC	Retry and Cancel push buttons until the end user responds to the message box.
may be combined with:	
1	Make the first push button the default.
2	Make the second push button the default.
3	Make the third push button the default.

Note: If the messagebox has the style "C", an OK button is generated because a messagebox with only a Cancel button is not supported.

Example:

```
OPEN DIALOG NGU-MESSAGEBOX
    USING NULL-HANDLE
    WITH #BUTTON
    'Do you want to save the changes ?'  /* These parameters are
    'Exit editor' '?YNC1'                /* passed BY VALUE
```

Separator Keyword

END-OF-LINE

You use this keyword to force line breaks in message box texts when using the NGU-MESSAGEBOX dialog. To do so, first embed this keyword in the alphanumeric "Message" string using the COMPRESS statement, for example:

```
COMPRESS 'HELLO' END-OF-LINE 'THIS IS A MESSAGE' TO #MESSAGE
```

You should, however, only use this technique if the text of the message clearly breaks into more than one line, because this overrides system-provided formatting.

Usage in Enter or Leave Event Handlers

Before you open this dialog in an enter or leave event handler, you have to set the attributes SUPPRESS-ENTER-EVENT and SUPPRESS-LEAVE-EVENT to TRUE. This avoids an infinite loop on opening the dialog (when the messagebox appears, another enter and leave event occurs for the dialog or dialog element containing the enter or leave event handler). After the messagebox has been invoked, you modify the attributes to value FALSE again.

396 NGU-SERVER-DATA Subprogram

- Natural Object Name 978
- Description 978

Natural Object Name

NGUSVDAT

Description

Sends the data for service DDE-VIEW.SERVICE, topic DDE-VIEW.TOPIC, conversation-ID DDE-VIEW.CONV-ID, and item DDE-VIEW.ITEM to the client.

The data to be sent are contained in DDE-VIEW.DATA and must have been requested by the client in a "Request" or "Advise" message. DDE-VIEW.ITEM must be set to the item that was requested. Note that even if the "Advise" message resulted in a warm link, the data should still be specified. Natural will then decide whether the data will actually be sent to the client.

When sending data to the client, you must set DDE-VIEW.DATALEN to a value:

- If set to "0", the entire data will be sent; the client will send on return a value that indicates how many data were actually sent.
- If set to "-1", trailing blanks will be removed and the client will send on return a value that indicates how many data were actually sent.
- If set to a value greater zero, this amount of data is actually sent.

Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

397 NGU-SERVER-GET-DATA Subprogram

- Natural Object Name 980
- Parameters 981

Natural Object Name

NGUSVGDA

Description

Lets the server retrieve data supplied with a message from some client. You must use this subprogram in the event handler of a DDE-server event. DDE-VIEW.MESSAGE will hold one of the values described below. You must then ensure that your event handler code acts on the value of DDE-VIEW.MESSAGE.

The possible values of DDE-VIEW.MESSAGE are:

CONNECT	On request by a client, a conversation has been established with the topic DDE-VIEW.TOPIC and the service DDE-VIEW.SERVICE. Topic and service must have been registered by the server. The value in DDE-VIEW.CONV-ID uniquely identifies this conversation.
DISCONNECT	On request by a client, the conversation identified by DDE-VIEW.CONV-ID has been closed.
REQUEST	The client of the conversation specified in DDE-VIEW.CONV-ID has requested data as specified in DDE-VIEW.ITEM.
ADVISE	The client of the conversation specified in DDE-VIEW.CONV-ID has requested a data link for the data specified in DDE-VIEW.ITEM. The DDE-server event handler must then use the NGU-SERVER-DATA subprogram whenever new data become available during the conversation.
UNADVISE	The client of the conversation specified in DDE-VIEW.CONV-ID has requested to close a data link for the data specified in DDE-VIEW.ITEM. (This data link was previously opened using "Advise".)
POKE	The client of the conversation specified in DDE-VIEW.CONV-ID has sent data as specified in DDE-VIEW.ITEM and DDE-VIEW.DATA. These data have the length of DDE-VIEW.DATALEN. DDE-VIEW.DATA should be chosen as appropriate for the expected conversations.
EXECUTE	The client of the conversation specified in DDE-VIEW.CONV-ID has requested the execution of the command specified in DDE-VIEW.DATA and DDE-VIEW.ITEM. The length of this command is held in DDE-VIEW.DATALEN.
TIMEOUT	A timeout has occurred. No client message has been received during a specified time interval.

When DDE-VIEW.MESSAGE has the value DISCONNECT, DATA, or NOTIFY, the fields DDE-VIEW.SERVICE and DDE-VIEW.TOPIC are also set appropriately.

The value in DDE-VIEW.TIMEOUT specifies the maximum number of milliseconds the server wants to wait for a message:

- If set to "-1", the server will wait indefinitely.

- If set to "0", a message will be returned immediately, if there is one pending, or "Timeout" will be returned.
- If set to any other value, the exact time that elapses before a message is returned can only be guaranteed to be no less than DDE-VIEW.TIMEOUT.

Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

398 NGU-SERVER-REGISTER Subprogram

- Natural Object Name 984
- Parameters 984

Natural Object Name

NGUSVREG

Description

This subprogram makes it known that the server supports the service specified in DDE-VIEW.SERVICE and the conversations on the topic specified in DDE-VIEW.TOPIC. A client will not be able to establish a conversation with the server until this subprogram is executed. Your client should not try to register the name "Natural" because this is reserved.

Before you execute this subprogram, you must assign the *DIALOG-ID system variable of the server dialog to DDE-VIEW.CONV-ID. You must also assign the string "DLGID" to DDE-VIEW.MESSAGE.

Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

399 NGU-SERVER-STOP Subprogram

- Natural Object Name 986
- Description 986
- Parameters 986

Natural Object Name

NGUSVSTP

Description

Terminates all DDE activity for the server. Any topics registered for the service are unregistered and any open conversations are terminated. The server should not call any other DDE transaction after this subprogram has been executed.

Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

400 NGU-SERVER-UNREGISTER Subprogram

- Natural Object Name 988
- Description 988
- Parameters 988

Natural Object Name

NGUSVUNR

Description

Makes the topic DDE-VIEW.TOPIC unavailable. Any open conversations on that topic should be closed first.

Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

401 NGU-SERVER-WAIT Subprogram

- Natural Object Name 990
- Description 990
- Parameters 991

Natural Object Name

NGUSVWAT

Description

Lets the server wait for data supplied with a message from some client. DDE-VIEW.MESSAGE will hold one of the values described below. You must then ensure that your event handler code acts on the value of DDE-VIEW.MESSAGE

The possible values of DDE-VIEW.MESSAGE are:

CONNECT	On request by a client, a conversation has been established with the topic DDE-VIEW.TOPIC and the service DDE-VIEW.SERVICE. Topic and service must have been registered by the server. The value in DDE-VIEW.CONV-ID uniquely identifies this conversation.
DISCONNECT	On request by a client, the conversation identified by DDE-VIEW.CONV-ID has been closed.
REQUEST	The client of the conversation specified in DDE-VIEW.CONV-ID has requested data as specified in DDE-VIEW.ITEM.
ADVISE	The client of the conversation specified in DDE-VIEW.CONV-ID has requested a data link for the data specified in DDE-VIEW.ITEM. The DDE-server event handler must then use the NGU-SERVER-DATA subprogram whenever new data become available during the conversation.
UNADVISE	The client of the conversation specified in DDE-VIEW.CONV-ID has requested to close a data link for the data specified in DDE-VIEW.ITEM. (This data link was previously opened using "Advise".)
POKE	The client of the conversation specified in DDE-VIEW.CONV-ID has sent data as specified in DDE-VIEW.ITEM and DDE-VIEW.DATA. These data have the length of DDE-VIEW.DATALEN. DDE-VIEW.DATA should be chosen as appropriate for the expected conversations.
EXECUTE	The client of the conversation specified in DDE-VIEW.CONV-ID has requested the execution of the command specified in DDE-VIEW.DATA and DDE-VIEW.ITEM. The length of this command is held in DDE-VIEW.DATALEN.
TIMEOUT	A timeout has occurred. No client message has been received during a specified time interval.

When DDE-VIEW.MESSAGE has the value DISCONNECT, DATA, or NOTIFY, the fields DDE-VIEW.SERVICE and DDE-VIEW.TOPIC are also set appropriately.

The value in DDE-VIEW.TIMEOUT specifies the maximum number of milliseconds the server wants to wait for a message:

- If set to "-1", the server will wait indefinitely.

- If set to "0", a message will be returned immediately, if there is one pending, or "Timeout" will be returned.
- If set to any other value, the exact time that elapses before a message is returned can only be guaranteed to be no less than DDE-VIEW.TIMEOUT.

Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

402 NGULKEY1 Reserved Symbols

- Color Symbols 994
- Dialog Element Types 994
- Event-Suppressing Symbols 994
- Menu Item Style Symbols 994
- Menu Item Symbols 994
- Separator Symbols 995
- Tool Bar Symbols 995

Color Symbols

DEFAULT (0) WHITE (1) BLACK (2) LTGREY (3) GREY (4) DKGREY (5) RED (6) GREEN (7) BLUE (8) CYAN (9) MAGENTA (10) BROWN (11) YELLOW (12) LIGHTRED (13) LIGHTGREEN (14) LIGHTBLUE (15) LIGHTCYAN (16) LIGHTMAGENTA (17) BRIGHTWHITE (18) CUSTOM (50)

Dialog Element Types

BITMAP CANVAS COLUMNSPECIFICATION CONTEXTMENU CONTROLBOX EDITAREA FONT GRAPHICTEXT GROUPFRAME INPUTFIELD LINE LISTBOX LISTBOXITEM MDICHILD MDIFRAME MENUBAR MENUITEM PUSHBUTTON RADIOBUTTON RECTANGLE SCROLLBAR SELECTIONBOX SELECTIONBOXITEM SUBMENU TABLE TIMER TEXTCONSTANT TOGGLEBUTTON TOOLBAR TOOLBARITEM WINDOW

Event-Suppressing Symbols

NOT-SUPPRESSED SUPPRESSED

Menu Item Style Symbols

MT-NORMAL MT-SEPARATOR MT-SUBMENU MT-MDICASCADE MT-MDITILE MT-MDIARRANGE MT-WINDOWMENU MT-CUT MT-COPY MT-PASTE MT-DELETE MT-UNDO

Menu Item Symbols

UNCHECKED CHECKED

Separator Symbols

END-OF-LINE

For details on END-OF-LINE, see the section **NGU-MESSAGEBOX Dialog**.

HORIZONTAL-TAB

For details on HORIZONTAL-TAB, see the section **SET-TABS Action**.

Tool Bar Symbols

TB-TOP TB-BOTTOM TB-LEFT TB-RIGHT