pIntuiD

COLLABORATORS			
	TITLE : pIntuiD		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY		March 29, 2025	

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

Contents

1 pIntuiD **1**

1.1 pIntuiD.doc . . . 1
1.2 pintui.library/pIntuiAllgemeines() . . . 2
1.3 pintui.library/pOS_InitIntuiIOReq() . . . 4
1.4 pintui.library/pOS_SendInputEvent() . . . 4
1.5 pintui.library/pOS_CmpDoubleKlick() . . . 5
1.6 pintui.library/pOS_CheckDoubleKlick() . . . 6
1.7 pintui.library/pOS_SysIMessage() . . . 6
1.8 pintui.library/pOS_CreateIMessage() . . . 7
1.9 pintui.library/pOS_DeleteIMessage() . . . 8
1.10 pintui.library/pOS_DisplayBeep() . . . 9
1.11 pintui.library/pOS_LockIntuiBase() . . . 9
1.12 pintui.library/pOS_UnlockIntuiBase() . . . 10
1.13 pintui.library/pOS_LockIntuiGadgets() . . . 10
1.14 pintui.library/pOS_UnlockIntuiGadgets() . . . 11
1.15 pintui.library/pOS_LockIntuiGadget() . . . 11
1.16 pintui.library/pOS_UnlockIntuiGadget() . . . 12
1.17 pintui.library/pOS_OpenScreenA() . . . 12
1.18 pintui.library/pOS_CloseScreen() . . . 13
1.19 pintui.library/pOS_LockPubScreen() . . . 14
1.20 pintui.library/pOS_UnlockPubScreen() . . . 14
1.21 pintui.library/pOS_PubScreenStatus() . . . 15
1.22 pintui.library/pOS_GetIColorMap() . . . 16
1.23 pintui.library/pOS_GetWindowBorder() . . . 16
1.24 pintui.library/pOS_OpenWindowA() . . . 17
1.25 pintui.library/pOS_CloseWindow() . . . 20
1.26 pintui.library/pOS_ChangeWindowBox() . . . 21
1.27 pintui.library/pOS_ModifyIDCMP() . . . 21
1.28 pintui.library/pOS_ModifySharedIDCMP() . . . 23
1.29 pintui.library/pOS_WindowToFront() . . . 24

1.30 pintui.library/pOS_WindowToBack() . . . 24
1.31 pintui.library/pOS_BeginRefresh() . . . 25
1.32 pintui.library/pOS_EndRefresh() . . . 25
1.33 pintui.library/pOS_RefreshWindowFrame() . . . 26
1.34 pintui.library/pOS_InvalidWindowRect() . . . 27
1.35 pintui.library/pOS_ActivateWindow() . . . 27
1.36 pintui.library/pOS_ZipWindow() . . . 28
1.37 pintui.library/pOS_GetWindowIBox() . . . 28
1.38 pintui.library/pOS_CheckVisibleIBox() . . . 29
1.39 pintui.library/pOS_CheckVisibleIRect() . . . 30
1.40 pintui.library/pOS_SetWindowLimits() . . . 31
1.41 pintui.library/pOS_SetWindowTitles() . . . 31
1.42 pintui.library/pOS_InactivateWindow() . . . 32
1.43 pintui.library/pOS_SetWindowBusy() . . . 33
1.44 pintui.library/pOS_SetWindowTask() . . . 33
1.45 pintui.library/pOS_AddGadgets() . . . 34
1.46 pintui.library/pOS_RemGadgets() . . . 35
1.47 pintui.library/pOS_AddGadget() . . . 36
1.48 pintui.library/pOS_RemGadget() . . . 36
1.49 pintui.library/pOS_RefreshGadgets() . . . 37
1.50 pintui.library/pOS_RefreshLayerGadgets() . . . 38
1.51 pintui.library/pOS_NewIObjectA() . . . 38
1.52 pintui.library/pOS_DisposeIObject() . . . 39
1.53 pintui.library/pOS_SetGadgetAttrsA() . . . 40
1.54 pintui.library/pOS_GetGadgetAttr() . . . 40
1.55 pintui.library/pOS_ActivateGadget() . . . 41
1.56 pintui.library/pOS_EnableGadget() . . . 42
1.57 pintui.library/pOS_RefreshGadgetsMd() . . . 43
1.58 pintui.library/pOS_LayoutGadgets() . . . 44
1.59 pintui.library/pOS_InactivateGadget() . . . 44
1.60 pintui.library/pOS_IsGadgetActive() . . . 45
1.61 pintui.library/pOS_DrawIObj() . . . 46
1.62 pintui.library/pOS_IObjDisposeList() . . . 47
1.63 pintui.library/pOS_IObjGetBoxList() . . . 47
1.64 pintui.library/pOS_IObjDrawList() . . . 48
1.65 pintui.library/pOS_PreLayoutMenu() . . . 49
1.66 pintui.library/pOS_LayoutMenu() . . . 49
1.67 pintui.library/pOS_CreateMenuTagA() . . . 50
1.68 pintui.library/pOS_DeleteMenu() . . . 51

1.69 pintui.library/pOS_SetMenuHighLight() . . . 52
1.70 pintui.library/pOS_MenuHandleIEvent() . . . 52
1.71 pintui.library/pOS_OpenHandleMenuIEvent() . . . 53
1.72 pintui.library/pOS_CloseHandleMenuIEvent() . . . 54
1.73 pintui.library/pOS_GetMenuItemFromNum() . . . 54
1.74 pintui.library/pOS_GetMenuNumFromItem() . . . 55
1.75 pintui.library/pOS_SetWindowMenuChecker() . . . 55
1.76 pintui.library/pOS_GetWindowMenuChecker() . . . 56
1.77 pintui.library/pOS_EnableWindowMenu() . . . 57
1.78 pintui.library/pOS_CreateBubbleHelpA() . . . 57
1.79 pintui.library/pOS_DeleteBubbleHelp() . . . 58
1.80 pintui.library/pOS_CreateRequestWinA() . . . 59
1.81 pintui.library/pOS_DeleteRequestWin() . . . 60
1.82 pintui.library/pOS_EasyRequestArgs() . . . 61
1.83 pintui.library/pOS_RequestWinHandler() . . . 62
1.84 pintui.library/pOS_AddDrag() . . . 63
1.85 pintui.library/pOS_RemDrag() . . . 63
1.86 pintui.library/pOS_CreateDragA() . . . 64
1.87 pintui.library/pOS_DeleteDrag() . . . 65
1.88 pintui.library/pOS_CreateDragGfx() . . . 65
1.89 pintui.library/pOS_DeleteDragGfx() . . . 66
1.90 pintui.library/pOS_ConstructDragList() . . . 66
1.91 pintui.library/pOS_DestructDragList() . . . 67
1.92 pintui.library/pOS_LayoutDragList() . . . 67
1.93 pintui.library/pOS_CreateDragListGfx() . . . 68
1.94 pintui.library/pOS_DeleteDragListGfx() . . . 68
1.95 pintui.library/pOS_RenderDragList() . . . 69
1.96 pintui.library/pOS_MoveDragList() . . . 69
1.97 pintui.library/pOS_RestoreDragList() . . . 70
1.98 pintui.library/pOS_SaveDragList() . . . 71
1.99 pintui.library/pOS_GetDropAttFromDragList() . . . 71

Chapter 1

pIntuiD

1.1 pIntuiD.doc

pintui.library

pIntuiAllgemeines()	pOS_ActivateGadget()
pOS_ActivateWindow()	pOS_AddDrag()
pOS_AddGadget()	pOS_AddGadgets()
pOS_AllocIMsg()	pOS_BeginRefresh()
pOS_ChangeWindowBox()	pOS_CheckDoubleKlick()
pOS_CheckVisibleIBox()	pOS_CheckVisibleIRect()
pOS_CloseHandleMenuIEvent()	pOS_CloseScreen()
pOS_CloseWindow()	pOS_CmpDoubleKlick()
pOS_ConstructDragList()	pOS_CreateBubbleHelpA()
pOS_CreateDragA()	pOS_CreateDragGfx()
pOS_CreateDragListGfx()	pOS_CreateIMessage()
pOS_CreateMenuTagA()	pOS_CreateRequestWinA()
pOS_DeleteBubbleHelp()	pOS_DeleteDrag()
pOS_DeleteDragGfx()	pOS_DeleteDragListGfx()
pOS_DeleteIMessage()	pOS_DeleteMenu()
pOS_DeleteMenuTag()	pOS_DeleteRequestWin()
pOS_DestructDragList()	pOS_DisplayBeep()
pOS_DisposeIObject()	pOS_DrawIObj()
pOS_EasyRequestArgs()	pOS_EnableGadget()
pOS_EnableWindowMenu()	pOS_EndRefresh()
pOS_GetDropAttFromDragList()	pOS_GetGadgetAttr()
pOS_GetIColorMap()	pOS_GetMenuItemFromNum()
pOS_GetMenuNumFromItem()	pOS_GetWindowBorder()
pOS_GetWindowIBox()	pOS_GetWindowMenuChecker()
pOS_InactivateGadget()	pOS_InactivateWindow()
pOS_InitIntuiIOReq()	pOS_InvalidWindowRect()
pOS_IObjDisposeList()	pOS_IObjDrawList()
pOS_IObjGetBoxList()	pOS_IsGadgetActive()
pOS_LayoutDragList()	pOS_LayoutGadgets()
pOS_LayoutMenu()	pOS_LockIntuiBase()
pOS_LockIntuiGadget()	pOS_LockIntuiGadgets()
pOS_LockPubScreen()	pOS_MenuHandleIEvent()
pOS_ModifyIDCMP()	pOS_ModifySharedIDCMP()
pOS_MoveDragList()	pOS_NewIObjectA()
pOS_OpenHandleMenuIEvent()	pOS_OpenScreenA()
pOS_OpenWindowA()	pOS_PreLayoutMenu()

```
pOS_PubScreenStatus()          pOS_RefreshGadgets()
pOS_RefreshGadgetsMd()         pOS_RefreshLayerGadgets()
pOS_RefreshWindowFrame()       pOS_RemDrag()
pOS_RemGadget()                pOS_RemGadgets()
pOS_RenderDragList()           pOS_RequestWinHandler()
pOS_RestoreDragList()          pOS_SaveDragList()
pOS_SendInputEvent()           pOS_SetGadgetAttrsA()
pOS_SetMenuHighLight()         pOS_SetWindowBusy()
pOS_SetWindowLimits()          pOS_SetWindowMenuChecker()
pOS_SetWindowTask()            pOS_SetWindowTitles()
pOS_SysIMessage()              pOS_UnlockIntuiBase()
pOS_UnlockIntuiGadget()        pOS_UnlockIntuiGadgets()
pOS_UnlockPubScreen()          pOS_WindowToBack()
pOS_WindowToFront()            pOS_ZipWindow()
```

1.2 pintui.library/pIntuiAllgemeines()

```
        pIntui.doc - ROM-Library
        Stand: 03.12.96, Michael Christoph


 STRUKTUREN

        struct pOS_BubbleHelp
        struct pOS_DoubleKlick
        struct pOS_Drag
        struct pOS_DragList
        struct pOS_DrawInfo
        struct pOS_DropAttribute
        struct pOS_EasyStruct
        struct pOS_GadgetDrop
        struct pOS_GadgetMethod
        struct pOS_IBorder
        struct pOS_IClassInfo
        struct pOS_IntuiBorder
        struct pOS_IntuiDevice
        struct pOS_IntuiImage
        struct pOS_IntuiMessage
        struct pOS_IntuiMethod
        struct pOS_IntuiObj
        struct pOS_IntuiSpcData
        struct pOS_IntuiStdReq
        struct pOS_IntuiSymbol
        struct pOS_IntuiText
        struct pOS_IntuiUnit
        struct pOS_IObjectMethod
        struct pOS_LayoutDamage
        struct pOS_MenuDraw
        struct pOS_MenuItem
        struct pOS_MenuMethod
        struct pOS_MenuNum
        struct pOS_MenuTag
        struct pOS_PubScreenNode
        struct pOS_RastPort
        struct pOS_Screen
```

```
        struct pOS_ScreenBase
        struct pOS_Window

ENUMERATION

        enum pOS_BubbleType
        enum pOS_EasyTags
        enum pOS_FrameClassType
        enum pOS_GadgetClassMethods
        enum pOS_GadgetClassRender
        enum pOS_GadgetClassResult
        enum pOS_GadgetLayoutFlags
        enum pOS_IntuiBubTags
        enum pOS_IntuiClassMethods
        enum pOS_IntuiClassTags
        enum pOS_IntuiDeviceUnitNum
        enum pOS_IntuiMessageClass
        enum pOS_IntuIOReqCommands
        enum pOS_IntuiTextFlags
        enum pOS_IntuiUnitFlags
        enum pOS_IntuiUnitHelpFlags
        enum pOS_IObjectClassLayout
        enum pOS_IObjectClassMethods
        enum pOS_IObjectDrawMode
        enum pOS_MenuClassMethods
        enum pOS_MenuClassRender
        enum pOS_MenuClassResult
        enum pOS_MenuDrawFlags
        enum pOS_MenuItemFlags
        enum pOS_MenuNumPck
        enum pOS_MenuTagType
        enum pOS_OPAClassTags
        enum pOS_PubScreenNodeFlags
        enum pOS_ScreenAllocFlags
        enum pOS_ScreenFlags
        enum pOS_ScreenINames
        enum pOS_ScreenIObjs
        enum pOS_ScreenPens
        enum pOS_ScreenTags
        enum pOS_WindowFlags
        enum pOS_WindowIntuiFlags

DEFINES

        pOS_INTUIIMAGECNT

INCLUDES

        pIntui/BubTags.h
        pIntui/EasyReq.h
        pIntui/EasyTags.h
        pIntui/FraClass.h
        pIntui/GClass.h
        pIntui/ICI.h
        pIntui/IClass.h
        pIntui/IntuBase.h
        pIntui/IntuMsg.h
```

```
      pIntui/IntuUnit.h
      pIntui/IObj.h
      pIntui/LayDam.h
      pIntui/Menu.h
      pIntui/MClass.h
      pIntui/PIObj.h
      pIntui/Tags.h
      pIntui/OClass.h
      pIntui/OPATags.h
      proto/pIntui.h

      typedef int (*pOS_ScrCMDTab)(pOS_ScreenBase*,pOS_IntuiStdReq*);
```

1.3 pintui.library/pOS_InitIntuiIOReq()

```
PROTOTYP
      pOS_Process *res = pOS_InitIntuiIOReq
        (
          pOS_IntuiDevice *intuibase,
          pOS_IntuiStdReq *intuistdreq
        );

FUNKTION
      Installieren einer pOS_IntuiStdReq-Struktur

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      intuistdreq (_R_A0)
        Zeiger auf die zu installierende pOS_IntuiStdReq-Struktur

ERGEBNIS
      res (_R_D0)
        Zeiger auf den eigenen Process.

SIEHE AUCH
      %

AMIGA FUNKTION
      %
```

1.4 pintui.library/pOS_SendInputEvent()

```
PROTOTYP
      VOID pOS_SendInputEvent
        (
          pOS_IntuiDevice *intuibase,
          const pOS_InputEvent *inputevent
        );
```

```
FUNKTION
      Bearbeiten von InputEvents, die entsprechend der Daten im
      Betriebssystem weitergeleitet werden.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      inputevent (_R_A0)
        Zeiger auf den ausgefüllten InputEvent, der im System
        versendet werden soll.

HINWEIS
      Die Funktion sollte mit Vorsicht benutzt werden. Wenn Sie
      im falschen Moment aufgerufen wird, kann das den ganzen
      Rechner blockieren.
      So z.B. innerhalb des Input-Devices oder bei gelockter
      IntuiBase. Je nach InputEvent können auch gelockte Gadgets
      oder Layers zu Problemen führen.

BEISPIEL
      /* Simulieren eines linken Mausklicks bei Position 100,100 */
      pOS_InputEvent ie={ 0 };
      ie.ie_Class    = IECLASS_RAWMOUSE;
      ie.ie_Code     = IECODE_LBUTTON;
      ie.ie_Qualifier= IEQUALIFIER_LEFTBUTTON;
      ie.ie_xy.ie_x=100;
      ie.ie_xy.ie_y=100;
      pOS_SendInputEvent(gb_IntuiBase,&inputevent);

SIEHE AUCH
      %

AMIGA FUNKTION
      %
```

1.5 pintui.library/pOS_CmpDoubleKlick()

```
PROTOTYP
      BOOL res = pOS_CmpDoubleKlick
        (
          pOS_IntuiDevice *intuibase,
          const pOS_DoubleKlick *old,
          const pOS_DoubleKlick *just
        );

FUNKTION
      Testen, ob die beiden Zeitwerte sich innerhalb der eingestellten
      Maus-Doppelklick-Zeit befinden.
      Es muß sich dabei um die selbe gedrückte Maustaste handeln.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      (_R_A0)
```

```
        erster (älterer) Zeitwert (dk_Seconds und dk_Micros)
      (_R_A1)
        zweiter (neuerer) Zeitwert (dk_Seconds und dk_Micros)

ERGEBNIS
      res (_R_D0)
        TRUE wenn ein Doppelklick vorliegt, sonst FALSE

SIEHE AUCH
      pOS_CheckDoubleKlick()

AMIGA FUNKTION
      LONG DoubleClick(ULONG,ULONG,ULONG,ULONG);
```

1.6 pintui.library/pOS_CheckDoubleKlick()

```
PROTOTYP
      BOOL res = pOS_CheckDoubleKlick
        (
          pOS_IntuiDevice *intuibase,
          const pOS_IntuiMessage *intuimessage
        );

FUNKTION
      Testen, ob die aktuelle und letzte IntuiMessage sich innerhalb
      der eingestellten Maus-Doppelklick-Zeit befinden.
      Es muß sich dabei um die selbe gedrückte Maustaste handeln.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      intuimessage (_R_A0)
        aktuelle IntuiMessage

ERGEBNIS
      res (_R_D0)
        TRUE wenn zwischen der letzten und aktuellen IntuiMessage-
        Struktur ein Maus-Doppelklick vorliegt.

SIEHE AUCH
      pOS_CmpDoubleKlick()

AMIGA FUNKTION
      LONG DoubleClick(ULONG,ULONG,ULONG,ULONG);
```

1.7 pintui.library/pOS_SysIMessage()

```
PROTOTYP
      BOOL res = pOS_SysIMessage
        (
```

```
          pOS_IntuiDevice *intuibase,
          pOS_IntuiMessage *intuimessage
        );

FUNKTION
      Weiterreichen einer Nachricht zur systeminternen Bearbeitung
      und evtl. Weiterleitung an weitere Objekte (z.B. Gadgets).

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      intuimessage (_R_A0)
        Nachricht, die mittels pOS_GetMsg() besorgt wurde

ERGEBNIS
      res (_R_D0)
        TRUE wenn die Nachricht vollständig bearbeitet und
        zurückgeschickt wurde. Bei FALSE muß die Nachricht
        mittels pOS_ReplyMsg() zurückgeschickt werden.

BEISPIEL

      {
        pOS_IntuiMessage *Msg;
        ...
        while(Msg=(pOS_IntuiMessage*)pOS_GetMsg(win->win_UserPort)) {
          if(!pOS_SysIMessage(Msg)) {
            switch(Msg->im_Class) {
              case :
              ...
            }
            pOS_ReplyMsg(&Msg->im_Message);
          }
        }
        ...
      }

SIEHE AUCH
      pexec.library/pOS_GetMsg(), pexec.library/pOS_ReplyMsg()

AMIGA FUNKTION
      %
```

1.8 pintui.library/pOS_CreateIMessage()

```
PROTOTYP
      __ARID__ pOS_IntuiMessage *res = pOS_CreateIMessage
        (
          pOS_IntuiDevice *intuibase,
          pOS_MsgPort *replyport,
          size_t size
        );

FUNKTION
```

```
      Reservieren und installieren einer pOS_IntuiMessage-Datenstruktur.
      In 'im_Data' befindet sich der Zeiger auf den privaten Datenteil.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      replyport (_R_A0)
        Zeiger auf den Messageport, an den die Nachricht zurückgeschickt
        werden soll (wird in res->mn_ReplyPort eingetragen)
      size (_R_D0)
        Größe der privaten Datenstruktur, falls private Daten mit in der
        Nachricht transportiert werden sollen.
        Size=0 => erzeugt eine 'normale' IDCMP-Message.

ERGEBNIS
      res (_R_D0)
        Adresse der reservierten Datenstruktur, die mittels
        pOS_DeleteIMessage() wieder freigegeben werden muß.

SIEHE AUCH
      pOS_DeleteIMessage()

AMIGA FUNKTION
      %
```

1.9 pintui.library/pOS_DeleteIMessage()

```
PROTOTYP
      VOID pOS_DeleteIMessage
        (
          pOS_IntuiDevice *intuibase,
          __ARID__ pOS_IntuiMessage *intuimsg
        );

FUNKTION
      Freigeben einer pOS_IntuiMessage-Datenstruktur.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      intuimsg (_R_A0)
        Zeiger, der von pOS_CreateIMessage() zurückgeliefert wurde
        Nach der Funktion darf auf den Zeiger nicht mehr
        zurückgegriffen werden.

SIEHE AUCH
      pOS_CreateIMessage()

AMIGA FUNKTION
      %
```

1.10 pintui.library/pOS_DisplayBeep()

```
PROTOTYP
      VOID pOS_DisplayBeep
        (
          pOS_IntuiDevice *intuibase,
          _R_A0 pOS_Screen *screen,
          _R_D0 ULONG type
        );

FUNKTION
      Aufblitzen eines Bildschirms veranlassen.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      screen (_R_A0)
        Zeiger auf den Screen, der aufblitzen soll.
        NULL wenn alle Bildschirme aufblitzen sollen.
      type (_R_D0) (enum pOS_BeepTypes)

          BEEPTYP_Warn   - Aktion kann nicht ausgeführt werden.
          BEEPTYP_Error  - Fehler aufgetretten
          BEEPTYP_Wackup - Signal, Zeit erreicht, Prozeß ist fertig

SIEHE AUCH
      %

AMIGA FUNKTION
      VOID DisplayBeep(struct Screen *screen);
```

1.11 pintui.library/pOS_LockIntuiBase()

```
PROTOTYP
      VOID pOS_LockIntuiBase
        (
          pOS_IntuiDevice *intuibase,
          pOS_IntuiUnit *unit
        );

FUNKTION
      Sperren der pOS_IntuiDevice-Unit gegen Zugriffe anderer Tasks.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      unit (_R_A0)
        Adresse der zu sperrenden Intui-Unit

HINWEIS
      Die Funktion sperrt die komplette interaktion (z.B. Mausbewegungen)
      und sollte deshalb nur kurzzeitig verwendet werden.
```

```
      Normalerweise ist nur eine Unit aktiv. In zukünftigen Versionen
      können aber mehrere Units für verschiedene Medien vorhanden sein.
      Z.Z. gibt es noch keine öffentliche Funktion, um die Unit-Adresse
      zu ermitteln.

SIEHE AUCH
      pOS_UnlockIntuiBase(), pOS_LockIntuiGadget()

AMIGA FUNKTION
      ULONG LockIBase(ULONG);
```

1.12 pintui.library/pOS_UnlockIntuiBase()

```
PROTOTYP
      VOID pOS_UnlockIntuiBase
        (
          pOS_IntuiDevice *intuibase,
          pOS_IntuiUnit *unit
        );

FUNKTION
      Aufheben einer Zugriffssperre auf die pOS_IntuiDevice-Unit.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      unit (_R_A0)
        Adresse der Intui-Unit, dessen Sperre aufgehoben werden soll

SIEHE AUCH
      pOS_LockIntuiBase(), pOS_UnlockIntuiGadget()

AMIGA FUNKTION
      VOID UnlockIBase(ULONG lock);
```

1.13 pintui.library/pOS_LockIntuiGadgets()

```
PROTOTYP
      VOID pOS_LockIntuiGadgets
        (
          pOS_IntuiDevice *intuibase,
          pOS_Screen *screen
        );

FUNKTION
      Sperren der Gadget-Liste gegen Zugriffe anderer Tasks.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
```

```
      screen (_R_A0)
        Zeiger auf den Screen, dessen Gadgets gegen Verändern
        gesperrt werden sollen

HINWEIS
      Normal genügt die Funktion pOS_LockIntuiGadget(). Nur bei
      Fensterübergreifenden Aktionen (z.B. Drag & Drop) müssen
      alle Gadgets des Bildschirms gesperrt werden.

SIEHE AUCH
      pOS_UnlockIntuiGadgets(), pOS_LockIntuiGadget()

AMIGA FUNKTION
      ULONG LockIBase(ULONG);
```

1.14 pintui.library/pOS_UnlockIntuiGadgets()

```
PROTOTYP
      VOID pOS_UnlockIntuiGadgets
        (
          pOS_IntuiDevice *intuibase
          pOS_Screen *screen
        );

FUNKTION
      Aufheben einer Zugriffssperre auf die Gadget-Liste.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      screen (_R_A0)
        Zeiger auf den Screen, dessen Gadgets wieder verändert
        werden dürfen

SIEHE AUCH
      pOS_LockIntuiGadgets()

AMIGA FUNKTION
      VOID UnlockIBase(ULONG lock);
```

1.15 pintui.library/pOS_LockIntuiGadget()

```
PROTOTYP
      VOID pOS_LockIntuiGadget
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window
        );

FUNKTION
```

```
      Sperren der Gadgetdaten-Strukturen gegen Zugriffe
      anderer Tasks.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Window, dessen Gadgets gegen Verändern
        gesperrt werden sollen

SIEHE AUCH
      pOS_UnlockIntuiGadget(), pOS_LockIntuiGadgets()

AMIGA FUNKTION
      ULONG LockIBase(ULONG);
```

1.16 pintui.library/pOS_UnlockIntuiGadget()

```
PROTOTYP
      VOID pOS_UnlockIntuiGadget
        (
          pOS_IntuiDevice *intuibase
          pOS_Window *window
        );

FUNKTION
      Aufheben einer Zugriffssperre auf die Gadgetdaten-Struktur.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Window, dessen Gadgets wieder verändert
        werden dürfen

SIEHE AUCH
      pOS_LockIntuiGadget()

AMIGA FUNKTION
      VOID UnlockIBase(ULONG lock);
```

1.17 pintui.library/pOS_OpenScreenA()

```
PROTOTYP
      __ARID__ pOS_Screen *res = pOS_OpenScreenA
        (
          pOS_IntuiDevice *intuibase,
          const pOS_TagItem *tagitem
        );
```

```
FUNKTION
      Öffnen eines Bildschirms.

      Diese Funktion ist in der ersten Release nicht verfügbar.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      tagitem (_R_A0) <pScreen/ScrTags.h> (enum pOS_ScreenTags)

          SCRTAG_LeftEdge (SLONG) - linke Ecke
          SCRTAG_TopEdge (SLONG) - obere Ecke
          SCRTAG_Width (ULONG) - Breite vom Screen
          SCRTAG_Height (ULONG) - Höhe vom Screen
          SCRTAG_Title (const CHAR*) - Titel
          SCRTAG_DefaultTitle scr_DefaultTitle (const CHAR*) - Titel
          SCRTAG_PubName (const CHAR*)
          SCRTAG_PubTask (struct pOS_Task*)
          SCRTAG_PubSignal (UBYTE) - Signal für SCRTAG_PubTask
          SCRTAG_TextFont (pOS_TextFont*)
          SCRTAG_MonFileName (const dofiname_t*)
          SCRTAG_MonLock (const pOS_MonLock*)
          SCRTAG_MonFile (const pOS_MonFile*)


ERGEBNIS
      res (_R_D0)
        Zeiger auf den geöffneten Bildschirm, oder NULL wenn der
        Bildschirm nicht geöffnet werden konnte.

SIEHE AUCH
      pOS_CloseScreen(), pLib.library/pOS_OpenScreen()

AMIGA FUNKTION
      struct Screen *OpenScreen(struct NewScreen *);
      struct Screen *OpenScreenTagList(struct NewScreen *, struct TagItem * ↩
         tagitem);
```

1.18 pintui.library/pOS_CloseScreen()

```
PROTOTYP
      BOOL res = pOS_CloseScreen
        (
          pOS_IntuiDevice *intuibase,
          __ARID__ pOS_Screen *screen
        );

FUNKTION
      Schließen eines Bildschirms.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      screen (_R_A0)
```

```
        Zeiger auf den von pOS_OpenScreenA() geöffneten Bildschirm

ERGEBNIS
      res (_R_D0)
        TRUE wenn der Bildschirm geschlossen werden konnte.
        FALSE wenn der Bildschirm weiterhin offen ist, da es sich um
        einen Public-Screen handelt, bzw. noch Fenster auf dem Bild-
        schirm offen sind.

SIEHE AUCH
      pOS_OpenScreenA()

AMIGA FUNKTION
      LONG CloseScreen(struct Screen *screen);
```

1.19 pintui.library/pOS_LockPubScreen()

```
PROTOTYP
      pOS_Screen *res = pOS_LockPubScreen
        (
          pOS_IntuiDevice *intuibase,
          const CHAR *name
        );

FUNKTION
      Sperren eines öffentlichen Bildschirms, damit er nicht ge-
      schlossen werden kann.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      name (_R_A0)
        Name des öffentlichen Bildschirms, der gesperrt werden soll.
        Dabei wird NICHT zwischen Groß- und Kleinschreibung unterschieden.
        NULL für den Default-PubScreen.

ERGEBNIS
      res (_R_D0)
        Zeiger auf den gesperrten Bildschirm, der mittels pOS_UnlockPubScreen()
        wieder freigegeben werden muß. NULL im Fehlerfall (z.B. Bildschirm
        nicht vorhanden).

SIEHE AUCH
      pOS_UnlockPubScreen(), pOS_PubScreenStatus()

AMIGA FUNKTION
      struct Screen *LockPubScreen(STRPTR name);
```

1.20 pintui.library/pOS_UnlockPubScreen()

```
PROTOTYP
      BOOL res = pOS_UnlockPubScreen
        (
          pOS_IntuiDevice *intuibase,
          pOS_Screen *screen
        );

FUNKTION
      Aufheben der Sperre eines öffentlichen Bildschirms.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      screen (_R_A0)
        Zeiger auf den von pOS_LockPubScreen() gesperrten Bildschirm

ERGEBNIS
      res (_R_D0)
        TRUE wenn die Bildschirmsperre aufgehoben wurde.
        FALSE im Fehlerfall, dann liefert pOS_GetIoErr() die Fehlerursache.

SIEHE AUCH
      pOS_LockPubScreen(), pOS_PubScreenStatus()

AMIGA FUNKTION
      VOID UnlockPubScreen(STRPTR, struct Screen *screen);
```

1.21 pintui.library/pOS_PubScreenStatus()

```
PROTOTYP
      ULONG res = pOS_PubScreenStatus
        (
          pOS_IntuiDevice *intuibase,
          pOS_Screen *screen,
          ULONG flags
        );

FUNKTION
      Ändern des Status eines öffentlichen Bildschirms

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      screen (_R_A0)
        Zeiger auf den geöffneten Bildschirm, dessen Status geändert
        werden soll.
      flags (_R_D0) (enum pOS_PubScreenNodeFlags)
        PUBSCRF_Private : den Bildschirm nicht mehr öffentlich
                         zugängig machen

ERGEBNIS
      res (_R_D0)
```

```
        PUBSCRF_Private wenn der Bildschirm gegen Besucher-Fenster
        gesperrt wurde; sonst 0 (keine Änderung bzw. kein Public-
        Screen).

SIEHE AUCH
      pOS_LockPubScreen(), pOS_UnlockPubScreen()

AMIGA FUNKTION
      ULONG PubScreenStatus(struct Screnn *screen, ULONG flags);
```

1.22 pintui.library/pOS_GetIColorMap()

```
PROTOTYP
      pOS_ColorMap *res = pOS_GetIColorMap
        (
          pOS_IntuiDevice *intuibase,
          const pOS_Screen *screen,
          const pOS_Window *window
        );

FUNKTION
      Ermitteln der Farbtabelle eines Bildschirms oder Fensters.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      screen (_R_A0)
        Adresse des Screens, dessen Farbtabelle ermittelt werden soll
        oder NULL.
      window (_R_A1)
        Adresse des Windows, dessen Farbtabelle ermittelt werden soll
        evtl. wird die Farbtabelle des Screen geliefert;
        oder NULL.

ERGEBNIS
      res (_R_D0)
        Adresse der ermittelten Farbtabelle, die bis zum Schließen
        des Fensters/Bildschirms gültig ist.

HINWEIS
      Es muß entweder screen oder window verwendet werden.

SIEHE AUCH
      %

AMIGA FUNKTION
      %
```

1.23 pintui.library/pOS_GetWindowBorder()

```
PROTOTYP
      BOOL res = pOS_GetWindowBorder
        (
          pOS_IntuiDevice *intuibase,
          pOS_Screen *screen,
          pOS_WBox *box,
          ULONG windowflags,
          ULONG gadgetflags,
          ULONG gadgetact
        );

FUNKTION
      Ermitteln der Größe für Border-Gadgets

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      screen (_R_A0)
        Zeiger auf den Screen, auf dem das Fenster geöffnet werden soll
      box (_R_A1)
        Zeiger auf die ibox, die die ermittelten Koordinaten aufnehmen soll
      windowflags (_R_D0)
        Window-Flags für die Position und die Relation
        (WINFLGF_SizeBRight/WINFLGF_SizeBBottom)
      gagetflags (_R_D1)
        Gadget-Flags für die Position und Relation
        (GFLG_RelBottom/GFLG_RelRight/GFLG_RelWidth/GFLG_RelHeight)
      gadgetact (_R_D2)
        Position für das Gadget
        (GACT_RightBorder/GACT_LeftBorder/GACT_TopBorder/GACT_BottomBorder)

ERGEBNIS
      res (_R_D0)
        TRUE wenn die Werte ermittelt werden konnten;
        sonst FLASE.

BEISPIEL
      Datei "pos_RKRM/pIntui/SuperWin.c"

SIEHE AUCH
      %

AMIGA FUNKTION
      %
```

1.24 pintui.library/pOS_OpenWindowA()

```
PROTOTYP
      __ARID__ pOS_Window *res = pOS_OpenWindowA
        (
          pOS_IntuiDevice *intuibase,
          const pOS_TagItem *tagitem
```

```
        );

FUNKTION
      Öffnen eines Fensters.

      In der ersten Release muß immer SCRTAGF_SimpleRefresh verwendet werden.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      tagitem (_R_A0) <pScreen/ScrTags.h> (enum pOS_ScreenTags)

          SCRTAG_LeftEdge (SLONG) - linke Ecke
          SCRTAG_TopEdge (SLONG) - obere Ecke
          SCRTAG_Width (ULONG) - Window-Breite
          SCRTAG_Height (ULONG) - Window -Höhe
          SCRTAG_Title (const CHAR*) - Window-Titel Text
          SCRTAG_PubName (const CHAR*)
            Names des PubScreen, auf dem das Window geöffnet werden soll.

          SCRTAG_Screen (struct pOS_Screen*)
            Screen, auf dem das Window geöffnet werden soll.

          SCRTAG_Flags (ULONG) (enum pOS_WindowFlags)

          SCRTAG_IDCMP (ULONG) (enum pOS_IntuiMessageClass)

          SCRTAG_MinWidth (ULONG) - minimale Window-Breite (außen)
          SCRTAG_MaxWidth (ULONG) Default=~0
            maximale Window-Breite (außen) (~0 = Screen-Breite)

          SCRTAG_MinHeight (ULONG) - minimale Window-Höhe (außen)
          SCRTAG_MaxHeight (ULONG) Default=~0
            maximale Window-Höhe (außen) (~0 = Screen-Höhe)

          SCRTAG_TextFont (pOS_TextFont*) - Window-Font, Default ist Screen-Font
          SCRTAG_Gadget (pOS_Gadget*)
            einzubindendes gadget. Es dürfen mehrere SCRTAG_Gadget-Tags
            angegeben werden.

          SCRTAG_DragMode (ULONG) - Art den Drag-Mods

          SCRTAG_SharedUserPort (pOS_MsgPort*)
            Setzt einen bereits erzeugten Message-Port als 'win_UserPort'.
            Beim pOS_CloseWindow() werden alle noch ausstehenden Messages
            vom Port entfernt, der Port wird jedoch nicht entfernt.

          SCRTAG_SuperGfxMap (pOS_GfxMap*) - Refresh-BitMap

          SCRTAG_AutoAdjust (BOOL)
            Window paßt sich an die Screen-Dimensionen an.

          SCRTAG_ZipGadget (BOOL) - Zip-Gadget erzeugen

          SCRTAG_PriTask (struct pOS_Task*)
            Beim Aktivieren vom Window wird die Task-Priorität auf SCRTAG_SetPri
            geändert. Beim Deaktivieren wird der Orginalwert wieder gesetzt.
```

```
      Siehe pOS_SetWindowTask()

    SCRTAG_SetPri (SBYTE)  - zu setzende Aktiv-Priotität.

    SCRTAG_MenuOnWinPos (BOOL) Default=TRUE, Menü hängt am Window
    SCRTAG_BgLayerHook (pOS_Callback*)
     Layer-Callback setzen.
     ~0 == keinen Callback verwenden > der Hintergrund wird nicht gelöscht
     0 == default Handler

    SCRTAG_AutoSizeGadget (pOS_Gadget*) (Gadget for Window-Sizing)
      Dieser Tag darf nur einmal vorhanden sein.
      Der System-Layouter berechnet die Window-Größe anhand dem
      AutoSize-Gadget. Jeder Re-Size hat zur Folge, daß der Layouter
      das Gadget-Layout dem Window anpaßt.

    SCRTAG_LayerFlags (ULONG) (enum pOS_LayerFlags)

      LAYERF_SzDamgHoriz - eine horizontale Größenänderung macht den
                           gesamten Layer ungültig
      LAYERF_SzDamgVert  - eine horizontale Größenänderung macht den
                           gesamten Layer ungültig

    SCRTAG_HelpID (const CHAR*) - HelpID vom Window
    SCRTAG_HelpFile (const CHAR*) - Help-File Name (Pfad)
    SCRTAG_ScreenTitle (const CHAR*)
      Screen-Titel, der bei aktivem Window dargestellt wird.

    SCRTAG_UnderMouse (BOOL) - Window wird unter der Maus geöffnet
    SCRTAG_UserHandler (pOS_Callback*)
      ( BOOL(*cb_Code)(_R_A0 const pOS_Callback*,_R_A1  ↩
         pOS_ScreenCallbackData*) )

    SCRTAG_BackgroundPen (ULONG) - Window-Hintergrund-Pen ( ↩
       win_BackfillColor)
    SCRTAG_SizeMode (BOOL) default=1
    SCRTAG_InnerWidth (ULONG) - innere Window-Größe
    SCRTAG_InnerHeight (ULONG) - innere Window-Größe
    SCRTAG_BgFillGfxMap (const pOS_GfxMap*)
      Anstatt einer Farbe wird GfxMap als Hintergrund im Window angezeigt.
      Die GfxMap mu0 'friend' vom Screen-RTG sein.

    SCRTAG_AlwaysWinDrag (BOOL)
      TRUE - Jeder Mausklick ins Window, läßt es verschieben.

    SCRTAG_InnerMinWidth (ULONG) - minimale Window-Breite (innen)
    SCRTAG_InnerMinHeight (ULONG) - minimale Window-Höhe (innen)
    SCRTAG_InnerMaxWidth (ULONG) - maximale Window-Breite (innen)
    SCRTAG_InnerMaxHeight (ULONG) - maximale Window-Höhe (innen)


Flags:
    SCRTAGF_Zoom (BOOL) - Zip-Gadget erzeugen
    SCRTAGF_SizeGadget (BOOL) - Size-Gadget erzeugen
    SCRTAGF_DragBar (BOOL) - Window kann verschoben werden
    SCRTAGF_DepthGadget (BOOL) - Depth-Gadget erzeugen
    SCRTAGF_CloseGadget (BOOL) - Close-gadget erzeugen
```

```
          SCRTAGF_Backdrop (BOOL) - Window als Backdrop
          SCRTAGF_NoCareRefresh (BOOL) - keine Refresh-Message senden
          SCRTAGF_Borderless (BOOL) - Window ohne Rahmen erzeugen
          SCRTAGF_Activate (BOOL) - Window ist aktiv
          SCRTAGF_SizeBRight (BOOL) - vergrößerter Border rechts
          SCRTAGF_SizeBBottom (BOOL) - vergrößerter Border unten
          SCRTAGF_GimmeZeroZero (BOOL) - vollständiges Clipping
          SCRTAGF_SimpleRefresh (BOOL) - Window sendet Refresh-Messages


ERGEBNIS
      res (_R_D0)
        Zeiger auf das geöffnete Fenster, oder NULL wenn das
        Fenster nicht geöffnet werden konnte.

SIEHE AUCH
      pOS_CloseWindow(), pOS_WindowToFront(), pOS_WindowToBack(),
      pLib.library/pOS_OpenWindow()

AMIGA FUNKTION
      struct Window *OpenWindow(struct NewWindow *);
      struct Window *OpenWindowTagList(struct NewWindow *, struct TagItem * ↩
         tagitem);
```

1.25 pintui.library/pOS_CloseWindow()

```
PROTOTYP
      BOOL res = pOS_CloseWindow
        (
          pOS_IntuiDevice *intuibase,
          __ARID__ pOS_Window *window
        );

FUNKTION
      Schließen eines Fensters.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das von pOS_OpenWindowA() geöffnete Fenster
        Nach der Funktion darf auf diesen Zeiger nicht mehr zugegriffen
        werden.

ERGEBNIS
      res (_R_D0)
        TRUE wenn das Fenster geschlossen werden konnte.

SIEHE AUCH
      pOS_OpenWindowA()

AMIGA FUNKTION
      VOID CloseWindow(struct Window *window);
```

1.26 pintui.library/pOS_ChangeWindowBox()

```
PROTOTYP
      BOOL res = pOS_ChangeWindowBox
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          SLONG left,
          SLONG top,
          SLONG width,
          SLONG height
        );

FUNKTION
      Bewegen/vergrößern/verkleinern eines Fensters.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das zu ändernde Fenster
      left (_R_D0)
        Neue linke Ecke des Fensters
      top (_R_D1)
        Neue obere Ecke des Fensters
      width (_R_D2)
        Neue Breite des Fensters
      height (_R_D3)
        Neue Höhe des Fensters

ERGEBNIS
      res (_R_D0)
        TRUE wenn das Fenster sich in der gewünschten Position
        und der gewünschten Größe befindet. Sonst FALSE, wenn
        eine Koordinate ungültig ist oder das Fenster nicht
        auf die gewünschte Größe/Position geändert werden konnte.

HINWEIS
      Für left, top, width, height können ~0 angegeben werden,
      damit der jeweilige Wert nicht verändert wird.
      Es werden die mittels pOS_SetWindowLimits() eingestellten
      Grenzwerte beachtet.

SIEHE AUCH
      pOS_OpenWindow(), pOS_SetWindowLimits(), pOS_ZipWindow(),
      pOS_WindowToFront(), pOS_WindowToBack()

AMIGA FUNKTION
      VOID ChangeWindowBox(struct Window *window, LONG left, LONG top, LONG  ↩
         width, LONG height);
```

1.27 pintui.library/pOS_ModifyIDCMP()

```
PROTOTYP
      BOOL res = pOS_ModifyIDCMP
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          ULONG idcmp
        );

FUNKTION
      Ändern der IDCMP-Flags eines Fensters.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, dessen IDCMP-Flags geändert werden
        sollen
      idcmp (_R_D0) <pIntui/IntuiMsg.h> (enum pOS_IntuiMessageClass)
        Neue IDCMP-Werte. Mehrere Werte können Oder-Verknüpft werden .

        IDCMP_NewSize - Window wurde vergrößert
        IDCMP_RefreshWindow - Window-Inhalt muß erneuert werden
        IDCMP_MouseButtons - Maustaste wurde betätigt
        IDCMP_MouseMove - Maus wurde verschoben
        IDCMP_GadgetDown - Gadget wurde eingedrückt (GACT_Immediate)
        IDCMP_GadgetUp - Gadget wurde ausgedrückt,
                         Eingabe beendet (GACT_RelVerify)
        IDCMP_GadgetAbort - Gadget-Eingabe wurde abgebrochen (rechte Maustask)
        IDCMP_MenuPick - Menü-Punkt wurde ausgewählt
        IDCMP_CloseWindow - Close-Gadget wurde betätigt (SCRTAGF_CloseGadget)
        IDCMP_RawKey - rohe Tastatureingabe
        IDCMP_DragDrop - Drag-Drop wurde bearbeitet
        IDCMP_ActiveWindow - Window wurde aktiviert
        IDCMP_InactiveWindow - Window wurde de-aktiviert
        IDCMP_UpdateGadget - Gadget hat sich während der Eingabe verändert.
                             Wird erst nach GadgetDown gesenden.
                             Z.B. ein Schieber wird bewegt, ein Text wird
                             eingegeben.
        IDCMP_VanillaKey - aufbereitete Tastatureingabe, nur Ascii-Zeichen.
                           Alle anderen Eingaben werden als IDCMP_RawKey
                           gesendet.
        IDCMP_IntuiTicks - Bei aktiviem Window werden 10 Ticks pro Sekunde
                           gesendet. Werden die Ticks nicht bearbeitet, so
                           wird der Sender seine Tätigkeit aussetzen.
        IDCMP_ChangeWindow - Window wurde vergrößert oder verschoben
        IDCMP_Help - Help wurde ausglöst

        IDCMP_StdSysMsg - Bits für die allgemeinen Standard-Gadget-Klassen.


ERGEBNIS
      res (_R_D0)
        TRUE wenn die neuen Parameter gesetzt werden konnten.
        FALSE im Fehlerfall (Speichermangel bei neuem Message-Port)
```

```
HINWEIS
      Hatte das Fenster bisher noch keine IDCMP-Flags, so wird mit den
      neuen Flags ein Message-Port eingerichtet. Wird für idcmp 0 angegeben,
      so wird ein bestehender Message-Port entfernt.
      Wurde der Port mit SCRTAG_SharedUserPort dem Window mitgeteilt, so
      wird der Port nie gelöscht.

SIEHE AUCH
      pOS_OpenWindowA(), pOS_ModifySharedIDCMP()

AMIGA FUNKTION
      LONG ModifyIDCMP(struct Window *window, ULONG idcmp);
```

1.28 pintui.library/pOS_ModifySharedIDCMP()

```
PROTOTYP
      BOOL res = pOS_ModifySharedIDCMP
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          ULONG idcmp,
          ULONG mask
        );

FUNKTION
      Ändern der SharedIDCMP-Flags eines Fensters.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeicher auf das Fenster, dessen SharedIDCMP-Flags geändert
        werden sollen
      idcmp (_R_D0)
        Neue IDCMP-Werte. Mehrere Werte können Oder-Verknüpft werden .
        siehe <pIntui/IntuiMsg.h> (enum pOS_IntuiMessageClass)
      mask (_R_D1)
        Alle IDCMPs, die auf diese Maske passen und zum angegebenen Window
        gehören, werden aus dem Port korrekt und nichtbearbeitet gelöscht.

ERGEBNIS
      res (_R_D0)
        TRUE wenn die neuen Parameter gesetzt werden konnten.
        FALSE im Fehlerfall.

SIEHE AUCH
      pOS_OpenWindowA(), pOS_ModifyIDCMP()

AMIGA FUNKTION
      %
```

1.29 pintui.library/pOS_WindowToFront()

```
PROTOTYP
      BOOL res = pOS_WindowToFront
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window
        );

FUNKTION
      Verlagern eines Fensters in den Vordergrund

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, daß ganz nach vorne geholt werden soll

ERGEBNIS
      res (_R_D0)
        TRUE wenn sich das Fenster ganz vorne befindet,
        FALSE im Fehlerfall (Speichermangel)

SIEHE AUCH
      pOS_OpenWindowA(), pOS_WindowToBack()

AMIGA FUNKTION
      VOID WindowToFront(struct Window *window);
```

1.30 pintui.library/pOS_WindowToBack()

```
PROTOTYP
      BOOL res = pOS_WindowToBack
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window
        );

FUNKTION
      Verlagern eines Fensters in den Hintergrund

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, daß ganz nach hinten gelegt werden soll

ERGEBNIS
      res (_R_D0)
        TRUE wenn sich das Fenster ganz hinten befindet,
        FALSE im Fehlerfall (Speichermangel)
```

```
SIEHE AUCH
      pOS_OpenWindowA(), pOS_WindowToFront()

AMIGA FUNKTION
      VOID WindowToBack(struct Window *window);
```

1.31 pintui.library/pOS_BeginRefresh()

```
PROTOTYP
      BOOL res = pOS_BeginRefresh
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          const pOS_List *list
        );

FUNKTION
      Vorbereiten eines Fensters für den Refresh

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das zu refreshende Fenster
      list (_R_A1) (Nodes vom Typ const struct pOS_RegionRectangle*)
        Liste auf zusätzliche Clip-Bereiche des Fenster
        oder NULL.

ERGEBNIS
      res (_R_D0)
        FALSE wenn nichts zu refreshen existiert oder bei Speichermangel,
        sonst TRUE. Dann muß pOS_EndRefresh() benutzt werden.

HINWEIS
      Da innerhalb einer Refresphase kein neues Clipping
      benutzt werden darf, können über die Liste entsprechende
      Clip-Regionen bestimmt werden.
      Siehe pOS_BeginLayerUpdate()

SIEHE AUCH
      pOS_EndRefresh()

AMIGA FUNKTION
      VOID BeginRefresh(struct Window *window);
```

1.32 pintui.library/pOS_EndRefresh()

```
PROTOTYP
      BOOL res = pOS_EndRefresh
        (
```

```
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          ULONG compelte
        );

FUNKTION
      Beenden des Fensterrefreshes

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, daß sich im Refresh-Zustand befindet
      complete (_R_D0)
        TRUE wenn der Neuaufbau vollständig abgeschlossen ist,
        sonst FALSE. Dann kann z.B. eine andere Funktion mittels
        pOS_BeginRefresh() und pOS_EndRefresh() weitere Veränderungen
        im Fenster vornehmen.

ERGEBNIS
      res (_R_D0)
        Im Normalfall TRUE; bei Speichermangel FALSE.

HINWEIS
      Siehe pOS_EndLayerUpdate()

SIEHE AUCH
      pOS_BeginRefresh()

AMIGA FUNKTION
      VOID EndRefresh(struct Window *window, LONG flags);
```

1.33 pintui.library/pOS_RefreshWindowFrame()

```
PROTOTYP
      VOID pOS_RefreshWindowFrame
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window
        );

FUNKTION
      Neuzeichnen des Fensterrahmens einschließlich aller darin
      enthaltenen Border-Gadgets und der Titelleiste.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, dessen Rahmen und evtl. enthaltene
        Border-Gadgets neu gezeichnet werden sollen

SIEHE AUCH
```

```
      pOS_OpenWindow(), pOS_SetWindowTitles(), pOS_BeginRefresh(),  ↩
         pOS_EndRefresh()

AMIGA FUNKTION
      VOID RefreshWindowFrame(struct Window *window);
```

1.34 pintui.library/pOS_InvalidWindowRect()

```
PROTOTYP
      VOID pOS_InvalidWindowRect
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          const pOS_Rectangle *rectangle,
          ULONG mode
        );

FUNKTION
      Markieren eines Fensterbereiches als ungültig. Dadurch wird
      beim nächsten Refresh der Inhalt neu gezeichnet.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das betroffene Fenster
      rectangle (_R_A1)
        Bereich, der ungültig erklärt werden soll
      mode (_R_D0)

          INVALWINF_None      - keine grafische Aktion auslösen
          INVALWINF_CreateMsg - IDCMP_Refresh-Message wird erzeugt
          INVALWINF_ClearBg   - Background wird gelöscht


SIEHE AUCH
      pOS_BeginRefresh(), pOS_EndRefresh()

AMIGA FUNKTION
      %
```

1.35 pintui.library/pOS_ActivateWindow()

```
PROTOTYP
      VOID pOS_ActivateWindow
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window
        );
```

```
FUNKTION
      Aktivieren eines Fensters

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, daß aktiviert werden soll

SIEHE AUCH
      pOS_OpenWindow(), pOS_InactivateWindow()

AMIGA FUNKTION
      ActivateWindow(struct Window *window);
```

1.36 pintui.library/pOS_ZipWindow()

```
PROTOTYP
      BOOL res = pOS_ZipWindow
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window
        );

FUNKTION
      Ändern der Position und Größe eines Fenster auf die
      alternativen Zoom-Werte (entspricht dem Drücken des
      Zip-Gadgets im Fensterrahmen).

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, das sein Größe und Position
        entsprechend der alternativen Werte ändern soll

ERGEBNIS
      res (_R_D0)
        TRUE wenn das Fenster entsprechend der alternativen Werte
        geändert werden konnte; sonst FALSE.

SIEHE AUCH
      pOS_OpenWindow()

AMIGA FUNKTION
      VOID ZipWindow(struct Window *window);
```

1.37 pintui.library/pOS_GetWindowIBox()

```
PROTOTYP
```

```
      VOID pOS_GetWindowIBox
        (
          pOS_IntuiDevice *intuidevice,
          pOS_Window *window,
          pOS_IBox *ibox,
          const pOS_Layer *layer
        );

FUNKTION
      Berechnet die Fläche, die innerhalb des Fensters liegt.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, dessen Größendaten ermittelt werden sollen
      ibox (_R_A1)
        Zeiger auf die ibox, die die ermittelten Daten aufnehmen soll
      layer (_R_A2)
        für zukünftige Erweiterungen; z.Z. NULL

SIEHE AUCH
      %

AMIGA FUNKTION
      %
```

1.38 pintui.library/pOS_CheckVisibleIBox()

```
PROTOTYP
      BOOL res = pOS_CheckVisibleIBox
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          const pOS_IBox *ibox,
          const pOS_Layer *layer
        );

FUNKTION
      Prüfen ob die angegebene Box innerhalb des Fenster
      (evtl. nur  teilweise) sichtbar ist.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster oder NULL
      ibox (_R_A1)
        pOS_IBox mit den zu prüfenden Koordinaten und Ausmaßen
      layer (_R_A2)
        Zeiger auf einen Layer des Fenster (bei mehreren) oder NULL

ERGEBNIS
      res (_R_D0)
```

```
        TRUE wenn mindestens ein Pixel der ibox innerhalb des
        windows oder des layers sichtbar ist; sonst FALSE.

HINWEIS
      Es muß entweder window oder layer angegeben werden.

SIEHE AUCH
      pOS_CheckVisibleIRect(), pOS_CheckVisibleRect()

AMIGA FUNKTION
      %
```

1.39 pintui.library/pOS_CheckVisibleIRect()

```
PROTOTYP
      BOOL res = pOS_CheckVisibleIRect
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          const pOS_Rectangle *rectangle,
          const pOS_Layer *layer
        );

FUNKTION
      Prüfen ob die angegebene Box innerhalb des Fenster
      (evtl. nur teilweise) sichtbar ist.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster oder NULL
      rectangle (_R_A1)
        pOS_Rectangle mit den zu prüfenden Koordinaten
      layer (_R_A2)
        Zeiger auf einen Layer des Fenster (bei mehreren) oder NULL

ERGEBNIS
      res (_R_D0)
        TRUE wenn mindestens ein Pixel der ibox innderhalb des
        windows oder des layers sichtbar ist; sonst FALSE.

HINWEIS
      Es muß entweder window oder layer angegeben werden.

SIEHE AUCH
      pOS_CheckVisibleIBox(), pOS_CheckVisibleRect()

AMIGA FUNKTION
      %
```

1.40 pintui.library/pOS_SetWindowLimits()

```
PROTOTYP
      BOOL res = pOS_SetWindowLimits
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          ULONG minwidth,
          ULONG minheight,
          ULONG maxwidth,
          ULONG maxheight
        );

FUNKTION
      Ändern der Minimal- und Maximal-Größen eines Fensters

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Adresse des Fenster, dessen minimale und maximale Größen
        verändert werden sollen
      minwidth (_R_D0)
        neue Minimalbreite oder 0 für nicht verändern
      minheight (_R_D1)
        neue Minimalhöhe oder 0 für nicht verändern
      maxwidth (_R_D2)
        neue Maximalbreite oder 0 für nicht verändern
      maxheight (_R_D3)
        neue Maximalhöhe oder 0 für nicht verändern

ERGEBNIS
      res (_R_D0)
        TRUE wenn die neuen Werte gültig sind und im Fenster
        vermerkt wurden; sonst FALSE.

HINWEIS
      Alle Angaben beziehen sich auf die äußeren Fensterkanten.
      Die Größen der Fensterrahmen müssen Sie bei Bedarf
      entsprechend selber berücksichtigen.

SIEHE AUCH
      pOS_OpenWindow(), pOS_ZipWindow()

AMIGA FUNKTION
      LONG WindowLimits(struct Widow *window, LONG minwidth, LONG minheight,  ↩
         LONG maxwidth, LONG maxheight);
```

1.41 pintui.library/pOS_SetWindowTitles()

```
PROTOTYP
      VOID pOS_SetWindowTitles
```

```
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          const CHAR *windowtitle,
          const CHAR *screentitle
        );

FUNKTION
      Setzen eines neuen Fenster- und Bildschirmtitels

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, dessen Fenster- und Bildschirmtitel
        geändert werden sollen
      windowtitle (_R_A1)
        Neuer Text für die Titelzeile des Fensters
        oder NULL für keinen Titel anzeigen
        oder ~0 für nicht verändern
      screentitle (_R_A2)
        Neuer Text für die Titelzeile des Bildschirms bei aktivem Fenster
        oder NULL für keinen Titel anzeigen
        oder ~0 für nicht verändern

SIEHE AUCH
      pOS_OpenScreenA(), pOS_OpenWindow()

AMIGA FUNKTION
      VOID SetWindowTitles(struct Window *window, STRPTR windowtitle, STRPTR  ↩
         screentitle);
```

1.42 pintui.library/pOS_InactivateWindow()

```
PROTOTYP
      VOID pOS_InactivateWindow
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window
        );

FUNKTION
      Inaktivieren eines Fenster

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, das inaktiviert werden soll

HINWEIS
      Das Fenster wird nur bei aktivem Zustand inaktiviert;
      dann ist kein Fenster auf dem Bildschirm mehr aktiv!
      Ansonsten bleibt ein anderes aktives Fenster unbeeinflußt.
```

```
SIEHE AUCH
      pOS_ActivateWidow()

AMIGA FUNKTION
      %
```

1.43 pintui.library/pOS_SetWindowBusy()

```
PROTOTYP
      ULONG res = pOS_SetWindowBusy
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          ULONG status
        );

FUNKTION
      Ändern der Empfangsbereitschaft eines Fensters auf Eingaben.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, dessen Empfangsbereitschaft
        geändert werden soll
      status (_R_D0)

          WINBUSYF_None        - Normalzustand
          WINBUSYF_DisActivate - Window kann nicht aktiviert werden
          WINBUSYF_DisDrop     - das Window verweigert Drop
          WINBUSYF_DisDrag     - das Window verweigert Drag
          WINBUSYF_Inactivate  - dem Window wird der Fokus entzogen
          WINBUSYF_RemIDCMP    - Alle windowbezogene IDCMPs wie
                                 IDCMP_MouseButtons, IDCMP_GadgetDown,
                                 IDCMP_GadgetUp, IDCMP_VanillaKey,
                                 IDCMP_RawKey werden entfernt.
                                 => verhindert späte Reaktionen

ERGEBNIS
      res (_R_D0)
        Alter Empfangs-Status.

SIEHE AUCH
      %

AMIGA FUNKTION
      %
```

1.44 pintui.library/pOS_SetWindowTask()

```
PROTOTYP
      pOS_Task *res = pOS_SetWindowTask
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          pOS_Task *task,
          SLONG priactiv,
          SLONG prideactiv
        );

FUNKTION
      Verändern der Task-Prioritäten für ein Fenster.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Adresse des betroffenden Fensters
      task (_R_A1)
        Task dessen Prioritäten je nach Fensterzustand (aktiv/inaktiv)
        geändert werden sollen.
      priactiv (_R_D0)
        Task-Priorität wenn das Fenster aktiv ist
      prideactiv (_R_D1)
        Task-Priorität wenn das Fenster inaktiv ist

ERGEBNIS
      res (_R_D0)
        Übergebener Task-Zeiger

HINWEIS
      Durch die beiden Task-Prioritäten kann ein aktives Fenster
      auf Wunsch mehr Rechnenzeit zur Verfügung gestellt bekommen.
      Ohne Verwendung dieser Funktion, sind beide Prioritäten
      identisch. Siehe SCRTAG_PriTask, SCRTAG_SetPri

SIEHE AUCH
      pexec.library/pOS_SetTaskPriority()

AMIGA FUNKTION
      %
```

1.45 pintui.library/pOS_AddGadgets()

```
PROTOTYP
      VOID pOS_AddGadgets
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          pOS_List *list
        );
```

```
FUNKTION
      Hinzufügen von Gadgets in ein Fenster

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, in das die Gadgets eingehängt
        werden sollen
      list (_R_A1)
        Zeiger auf die Liste der einzuhängenden Gadgets

HINWEIS
      Die neuen Gadgets werdem immer am Ende einer bestehenden
      Gadgetliste eingefügt.

SIEHE AUCH
      pOS_RemGadgets(), pOS_AddGadget(), pOS_RemGadget()

AMIGA FUNKTION
      LONG AddGList(struct Window *window, struct Gadget *gadget, LONG position, ↩
          LONG number, struct Requester *requester);
```

1.46 pintui.library/pOS_RemGadgets()

```
PROTOTYP
      VOID pOS_RemGadgets
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          const pOS_Gadget *gadget,
          pOS_List *list,
          ULONG anzahl,
          ULONG reserved
        );

FUNKTION
      Entfernen von Gadgets aus einem Fenster

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, aus dem die Gadgets entfernt
        werden sollen
      gadget (_R_A1)
        Zeiger auf das erste Gadget, ab dem alle folgenden entfernt
        werden sollen oder NULL für alle Gadgets des Fenster
        ACHTUNG: bei NULL werden auch die Systemgadgets entfernt!
      list (_R_A2)
        Zeiger auf eine Liste, die die entfernten Gagets aufnehmen soll
      anzahl (_R_D0)
        Anzahl der zu entfernenden Gadgets, ~0 für alle folgenden
      reserved (_R_D1)
```

```
        für zukünftige Erweiterungen; z.Z. 0

SIEHE AUCH
      pOS_AddGadgets(), pOS_AddGadget(), pOS_RemGadget()

AMIGA FUNKTION
      LONG RemoveGList(struct Window *window, struct Gadget *gadget, LONG number ↩
         );
```

1.47 pintui.library/pOS_AddGadget()

```
PROTOTYP
      VOID pOS_AddGadget
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          pOS_Gadget *gadget
        );

FUNKTION
      Hinzufügen eines Gadgets in ein Fenster

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, in das das Gadgets eingehängt
        werden soll
      gadget (_R_A1)
        Zeiger auf das einzuhängende Gadget

HINWEIS
      Das neue Gadget wird immer am Ende einer bestehenden
      Gadgetliste eingefügt.

SIEHE AUCH
      pOS_RemGadget(), pOS_AddGadgets(), pOS_RemGadgets()

AMIGA FUNKTION
      LONG AddGadget(struct Window *window, struct Gadget *gadget, LONG position ↩
         );
```

1.48 pintui.library/pOS_RemGadget()

```
PROTOTYP
      VOID pOS_RemGadget
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          pOS_Gadget *gadget
```

```
        );

FUNKTION
      Entfernen eines Gadgets aus einem Fenster

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, aus dem das Gadgets entfernt
        werden soll
      gadget (_R_A1)
        Zeiger auf das zu entfernende Gadget

SIEHE AUCH
      pOS_AddGadget(), pOS_AddGadgets(), pOS_RemGadgets()

AMIGA FUNKTION
      VOID RemoveGaget(struct Window *window, struct Gadget *gadget);
```

1.49 pintui.library/pOS_RefreshGadgets()

```
PROTOTYP
      VOID pOS_RefreshGadgets
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          const pOS_Gadget *gadget,
          ULONG number
        );

FUNKTION
      Neuzeichnen der Gadgets eines Fensters

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, dessen Gadgets neu dargestellt
        werden sollen
      gadget (_R_A1)
        Zeiger auf das erste Gadget, ab dem alle folgenden
        neu gezeichnet werden sollen, oder NULL für das erste Gadget.
      number (_R_D0)
        Anzahl der neuzuzeichnenden Gadgets oder ~0 für alle.

HINWEIS
      Wenn das Fenster zusätzliche Tocher-Layers mit Gadgets
      enthält, werden diese nicht mitgezeichnet, sondern müssen
      mittels pOS_RefreshLayerGadgets() refresht werden.

SIEHE AUCH
      pOS_RefreshLayerGadgets(), pOS_RefreshGadgetsMd()
```

```
AMIGA FUNKTION
      VOID RefreshGadgets(struct Gadget *gadget, struct Window *window, struct  ↩
         Requester *requester);
```

1.50 pintui.library/pOS_RefreshLayerGadgets()

```
PROTOTYP
      VOID pOS_RefreshLayerGadgets
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          const pOS_Layer *layer
        );

FUNKTION
      Neuzeichnen der Gadgets eines Fensters, die sich in einem
      zusätzlichen Tochter-Layer befinden.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, in dem sich der Layer befindet
      layer (_R_A1)
        Zeiger auf den Layer, dessen Gadgets neu gezeichnet
        werden sollen

HINWEIS
      Wenn Sie mehrere Layers innerhalb eines Fensters verwenden,
      müssen Sie für jeden Layer der Gadgets enthält, diese
      Funktion aufrufen.

SIEHE AUCH
      pOS_RefreshGadgets(), pOS_RefreshGadgetsMd()

AMIGA FUNKTION
      %
```

1.51 pintui.library/pOS_NewIObjectA()

```
PROTOTYP
      __ARID__ APTR res = pOS_NewIObjectA
        (
          pOS_IntuiDevice *intuibase,
          pOS_NClass *class,
          const CHAR *name,
          ULONG version,
          const pOS_TagItem *tagitem
        );
```

```
FUNKTION
      Erzeugen eines neuen Klassen-Objektes entsprechend der Angaben.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      class (_R_A0)
        Zeiger auf eine bereits vorhandene Klasse oder NULL
      name (_R_A1)
        Name der gewünschten Klasse
      version (_R_D0)
        Version der gewünschten Klasse
      tagitem (_R_A2)
        Beschreibung der Klasseneigenschaften

ERGEBNIS
      res (_R_D0)
        Zeiger auf das erzeugte Klassen-Objekt, das mittels
        pOS_DisposeIObject() wieder freigegeben werden muß
        oder NULL im Fehlerfall.

HINWEIS
      Es muß entweder class oder name angegeben werden.

SIEHE AUCH
      pOS_DisposeIObject(), pOS_NewGObject()

AMIGA FUNKTION
      %
```

1.52 pintui.library/pOS_DisposeIObject()

```
PROTOTYP
      VOID pOS_DisposeIObject
        (
          pOS_IntuiDevice *intuibase,
          __ARID__ APTR iobject
        );

FUNKTION
      Freigeben eines Klassen-Objektes

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      iobject (_R_A0)
        Zeiger auf das freizugebende Objekt, wie er von pOS_NewIObjectA()
        geliefert wurde.
        ACHTUNG: Nach der Funktion darf dieser Zeiger nicht mehr
                 verwendet werden.

SIEHE AUCH
      pOS_NewIObjectA(), pOS_IObjDisposeList(), pOS_DisposeGObject()
```

```
AMIGA FUNKTION
      %
```

1.53 pintui.library/pOS_SetGadgetAttrsA()

```
PROTOTYP
      ULONG res = pOS_SetGadgetAttrsA
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          pOS_Gadget *gadget,
          const pOS_TagItem *tagitem
        );

FUNKTION
      Verändern der Attribute eines Gadgets

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, in dem sich das Gadget befindet
        NULL => Gadget wird bei Veränderung nicht refresht und
                die Gadget-Semaphore wird nicht gelockt.
                Bei nicht reentranten Gadgets kann es zu Problemen
                kommen.
      gadget (_R_A1)
        Zeiger auf das Gadget, das verändert werden soll
      tagitem (_R_A2) (enum pOS_IntuiClassTags)
        Tagliste mit den zu ändernden, klassenspezifischen,
        Argumenten

ERGEBNIS
      res (_R_D0)
        klassenabhängiger Rückgabewert

SIEHE AUCH
      pOS_GetGadgetAttr(), pLib.library/pOS_SetGadgetAttrs(), gadget.class,
      pOS_SetObjectData(), pOS_SetAbsObjectData()

AMIGA FUNKTION
      ULONG SetGadgetAttrsA(struct Gadget *gadget, struct Window *window, struct ↩
          Requester *requester, struct TagItem *taglist);
      VOID GT_GetGadgetAttrsA(struct Gadget *gadget, struct Window *window,  ↩
         struct Requester *requester, struct TagItem *taglist);
```

1.54 pintui.library/pOS_GetGadgetAttr()

```
PROTOTYP
      BOOL res = pOS_GetGadgetAttr
```

```
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          pOS_Gadget *gadget,
          ULONG tagid,
          ULONG *result
        );

FUNKTION
      Ermitteln von Attributen eines Gadgets

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        NULL => Die Gadget-Semaphore wird nicht gelockt.Bei nicht
                reentranten Gadgets kann es zu Problemen kommen.

      gadget (_R_A1)
        Zeiger auf das Gadget, von dem Werte ermittelt werden sollen
      tagid (_R_D0) (enum pOS_IntuiClassTags)
        klassenspezifischer Tag für den zu ermittelnden Wert
      result (_R_A2)
        Variable, die den Wert aufnehmen soll

ERGEBNIS
      res (_R_D0)
        TRUE wenn die gewünschten Werte ermittelt werden konnten;
        sonst FALSE.

BEISPIEL
      {
        /* ermitteln der Schieberposition eines Proportional-Gadgets */
        ULONG Data;
        pOS_GetGadgetAttr(gb_IntuiBase,window,gadget,ICLTAG_Selected,&Data);
        printf("Aktivzustand = %ld\n",Data);
      }

SIEHE AUCH
      pOS_SetGadgetAttrsA(), pOS_GetObjectData(), pOS_GetAbsObjectData()

AMIGA FUNKTION
      ULONG GetAttr(ULONG id, APTR object, ULONG *attr);
      LONG GT_GetGadgetAttrsA(struct Gadget *gadget, struct Window *window,  ↩
         struct Requester *requester, struct TagItem *taglist);
```

1.55 pintui.library/pOS_ActivateGadget()

```
PROTOTYP
      BOOL res = pOS_ActivateGadget
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          pOS_Gadget *gadget
```

```
        );

FUNKTION
      Aktivieren eines Gadgets zur Eingabe

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, in dem sich das Gadget befindet
      gadget (_R_A1)
        Zeiger auf das Gadget, daß aktiviert werden soll

ERGEBNIS
      res (_R_D0)
        TRUE wenn das Gadget, im allgemeinen zur Eingabe,
        aktiviert werden konnte; sonst FALSE

HINWEIS
      Normalerweise können nur Texteingabe-Gadgets aktiviert
      werden. Dazu muß aber das Fenster aktiv sein; es darf
      kein anderes Gadget aktiv sein und das Gadget darf nicht
      gegen Eingaben gesperrt sein.
      ListView-Gadgets können ebenfalls "aktiviert" werden.
      Dann kann mittels der Cursor-Tasten durch die Liste
      gescrollt werden.
      Ist das aktive Gadget kein GACT_FixFocus-Gadget, dann wird der
      Eingabefokus dem neuen Gadget 'gadget' zugewiesen.

SIEHE AUCH
      pOS_EnableGadget(), pOS_InactivateGadget(), pOS_IsGadgetActive()

AMIGA FUNKTION
      LONG ActivateGadget(struct Gadget *gadget, struct Window *window, struct  ↩
         Requester *requester);
```

1.56 pintui.library/pOS_EnableGadget()

```
PROTOTYP
      VOID pOS_EnableGadget
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          pOS_Gadget *gadget,
          ULONG enable
        );

FUNKTION
      Ändern der Wählbarkeit eines Gadgets

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
```

```
        Zeiger auf das Fenster, in dem sich das Gadget befindet
      gadget (_R_A1)
        Zeiger auf das Gadget, dessen Wählbarkeit geändert werden soll
      enable (_R_D0)
        1 für wählbar
        0 für gesperrt (mit Gitter überzogen)

SIEHE AUCH
      pOS_ActivateGadget()

AMIGA FUNKTION
      VOID OnGadget(struct Gadget *gadget, struct Window *window, struct  ↩
         Requester *requester);
      VOID OffGadget(struct Gadget *gadget, struct Window *window, struct  ↩
         Requester *requester);
```

1.57 pintui.library/pOS_RefreshGadgetsMd()

```
PROTOTYP
      VOID pOS_RefreshGadgetsMd
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          const pOS_Gadget *gadget,
          ULONG number,
          ULONG mode
        );

FUNKTION
      Neuzeichnen der Gadgets mit Modus

PARAMETER
          intuibase (_R_LB)
            Zeiger auf pIntui-Library
          window (_R_A0)
            Zeiger auf das Fenster, dessen Gadgets neu dargestellt
            werden sollen
          gadget (_R_A1)
            Zeiger auf das erste Gadget, ab dem alle folgenden
            neu gezeichnet werden sollen, oder NULL für das erste Gadget
          number (_R_D0)
            Anzahl der neuzuzeichnenden Gadgets oder ~0 für alle
            folgenden
          mode (_R_D1) (enum pOS_GadgetClassRender)
            GCLMTHRE_Update     : nur Änderungen zeichnen
      ~     GCLMTHRE_Redraw     : alles zeichnen
            GCLMTHRE_Toggle     : Aussehen vom Aktivzustand zeichnen
            GCLMTHRE_TickFrame  : nächstes Frame anzeigen (bei Animationen)
            GCLMTHRE_ToggleDrop : Aussehen vom Drop-Zustand zeichnen

SIEHE AUCH
      pOS_RefreshGadgets(), pOS_RefreshLayerGadgets()

AMIGA FUNKTION
```

```
      %
```

1.58 pintui.library/pOS_LayoutGadgets()

```
PROTOTYP
      VOID pOS_LayoutGadgets
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          const pOS_IBox *ibox,
          ULONG flags
        );

FUNKTION
      Systembenutzte Funktion zum Berechnen der Gadget-Positionen
      und -größen für ein Fenster.

PARAMETER
          intuibase (_R_LB)
            Zeiger auf pIntui-Library
          window (_R_A0)
            Zeiger auf das Fenster, dessen Gadgets neu berechnet werden sollen
          ibox (_R_A1)
            Bereich in dem sich die Gadgets befinden sollen
          flags (_R_D0) (enum pOS_GadgetLayoutFlags)
            GCLMTHLYF_ReSize   : Fenster-Größenänderung
      ~     GCLMTHLYF_AddGList : Gadgets wurden hinzugefügt
            GCLMTHLYF_Mask     : Maske für den Typ
            GCLMTHLYF_LD       : Layouter arbeitet mit Regionen (imly_LD)
            GCLMTHLYF_SizeAbort: Bearbeitung wird beim ReSize sofort abgebrochen

SIEHE AUCH
      pOS_RefreshGadgets(), pOS_RefreshLayerGadgets(),
      pOS_RefreshGadgetsMd()

AMIGA FUNKTION
      %
```

1.59 pintui.library/pOS_InactivateGadget()

```
PROTOTYP
      BOOL res = pOS_InactivateGadget
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          pOS_Gadget *gadget
        );

FUNKTION
      Inaktivieren eines Gadgets
```

```
PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, in dem sich das Gadget befindet
      gadget (_R_A1)
        Zeiger auf das gadget, das inaktiviert werden soll

ERGEBNIS
      res (_R_D0)
        TRUE wenn das Gadget inaktiviert wurde, sonst FALSE.

HINWEIS
      Das Gadget muß aktiviert sein, damit es inaktiviert werden
      kann. Ansonsten bewirkt diese Funktion nichts.

SIEHE AUCH
      pOS_ActivateGadget(), pOS_Enable(), pOS_InactivateWindow(),
      pOS_IsGadgetActive()

AMIGA FUNKTION
      %
```

1.60 pintui.library/pOS_IsGadgetActive()

```
PROTOTYP
      BOOL res = pOS_IsGadgetActive
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window window *window,
          const pOS_Gadget *gadget
        );

FUNKTION
      Prüfen ob ein Gadget aktiv ist.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, in dem sich das Gadget befindet
      gadget (_R_A1)
        Zeiger auf das zu prüfende Gadget

ERGEBNIS
      res (_R_D0)
        TRUE wenn das Gadget aktiv ist, sonst FALSE.

SIEHE AUCH
      pOS_ActivateGadget(), pOS_InactivateGadget()

AMIGA FUNKTION
```

```
      %
```

1.61 pintui.library/pOS_DrawIObj()

```
PROTOTYP
      VOID pOS_DrawIObj
        (
          pOS_IntuiDevice *intuibase,
          SLONG x,
          SLONG y,
          const pOS_IntuiObj *intuiobj,
          pOS_RastPort *rastport,
          const pOS_DrawInfo *drawinfo,
          ULONG mode,
          const pOS_Rectangle *rectangle
        );

FUNKTION
      Klassen-Objekt zeichnen

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      x (_R_D0)
        Positionsangabe
      y (_R_D1)
        Positionsangabe
      intuiobj (_R_A0)
        Zeiger auf das auszugebende Objekt
      rastport (_R_A1)
        Zeiger auf den RastPort, in dem das Objekt gezeichnet werden soll
      drawinfo (_R_A2)
        Zeiger auf die grafischen Ausgabedaten des Fensters
      mode (_R_D2) (enum pOS_IObjectDrawMode)
        IOBDMF_Normal       : normal
        IOBDMF_Selected     : selektiert
        IOBDMF_Disabled     : gesperrt
        IOBDMF_Busy         : nicht bereit
        IOBDMF_Border       : Gadget ist im Border
        IOBDMF_ActiveWindow : Fenster ist aktiv
        IOBDMF_Drag         : Object im Drag-Modus
        IOBDMF_Drop         : Object in Drop-Modus
        IOBDMF_Update       : vergleichbar mit GCLMTHRE_Update,
                              nur die Veränderung zeichnen
      rectangle (_R_A3)
        Clip-Bereich innerhalb dessen das Objekt sich zeichnen muß

SIEHE AUCH
      pOS_IObjDrawList()

AMIGA FUNKTION
      %
```

1.62 pintui.library/pOS_IObjDisposeList()

```
PROTOTYP
      VOID pOS_IObjDisposeList
        (
          pOS_IntuiDevice *intuibase,
          pOS_IntuiObj *intuiobj
        );

FUNKTION
      Freigeben mehrerer Klassen-Objekte

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      intuiobj (_R_A0)
        Zeiger auf das erste Objekt der Liste, die mittels
        intuiobj->iobj_Next verkettet sind

HINWEIS
      Ruft für jedes Objekt die Funktion pOS_DisposeIObj() auf.
      Die Objkete sind mittel 'iobj_Next' verkettet.

SIEHE AUCH
      pOS_IObjGetBoxList(), pOS_IObjDrawList(), pOS_DisposeIObj()

AMIGA FUNKTION
      %
```

1.63 pintui.library/pOS_IObjGetBoxList()

```
PROTOTYP
      VOID pOS_IObjGetBoxList
        (
          pOS_IntuiDevice *intuibase,
          const pOS_IntuiObj *intuiobj,
          pOS_IObjectMethod *method,
          pOS_Rectangle *rectangle
        );

FUNKTION
      Berechnen der maximalen Größe einer Objekt-Liste
      Die Objket sind mittel 'iobj_Next' verkettet.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      intuiobj (_R_A0)
        Zeiger auf das Klassen-Objekt, dessen Größe ermittelt
        werden soll
      method (_R_A1)
        IOBMTH_GetIBox
```

```
      rectangle (_R_A2)
        nimmt das Ergebnis auf

SIEHE AUCH
      pOS_IObjDisposeList()

AMIGA FUNKTION
      %
```

1.64 pintui.library/pOS_IObjDrawList()

```
PROTOTYP
      VOID pOS_IObjDrawList
        (
          pOS_IntuiDevice *intuibase,
          const pOS_IntuiObj *intuiobj,
          pOS_IObjectMethod *method,
          const pOS_Rectangle *layoutrect,
          ULONG mode,
          ULONG layout,
          ULONG newtick,
          ULONG oldtick,
          const pOS_Rectangle *rectangle
        );

FUNKTION
      Zeichnen von mehreren Klassen-Objekten (auch Animationen)

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      intuiobj (_R_A0)
        Zeiger auf das zu zeichende Objekt
      method (_R_A1)
        Datenstruktur zur Bearbeitung. Die Funtion setzt
          imth_Method           =IOBMTH_DrawAll;
          imth_Draw.imdr_LayBox =layoutrect;
          imth_Draw.imdr_Mode   =mode;
          imth_Draw.imdr_Layout =layout;
          imth_Draw.imdr_ClipBox=rectangle;
          imth_Draw.imdr_FrameNum = wird berechnet

      layoutrect (_R_A2)
        Größen-Bereich zum Ausrichten, entsprechend layout
      mode (_R_D0) (enum pOS_IObjectDrawMode)
      layout (_R_D1) (enum pOS_IObjectClassLayout)
        Siehe gadget.class ICLTAG_RenderLayMode
      newtick (_R_D2)
        aktuell ausgegebener Tick einer Animation oder 0
      oldtick (_R_D3)
        zuletzt ausgegebener Tick einer Animation oder 0 bei Einzelbildern
      rectangle (_R_A3)
        Clip-Bereich innerhalb dessen das Objekt sich zeichnen muß
```

```
SIEHE AUCH
      pOS_DrawIObj()

AMIGA FUNKTION
      %
```

1.65 pintui.library/pOS_PreLayoutMenu()

```
PROTOTYP
      ULONG res = pOS_PreLayoutMenu
        (
          pOS_IntuiDevice *intuibase,
          pOS_MenuList *list,
          const pOS_DrawInfo *drawinfo
        );

FUNKTION
      Vorberechnen von Menüeinträgen

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      list (_R_A0)
        Zeiger auf die Liste der Menübeschreibung
      drawinfo (_R_A1)
        Zeiger auf die grafischen Ausgabedaten des Fensters

ERGEBNIS
      res (_R_D0)
        Liefert einen Fehlercode (entsprechend pOS_GetIoErr())
        oder 0 für ok zurück.

SIEHE AUCH
      pOS_LayoutMenu(), pOS_CreateMenuTagA(), pOS_SetWindowMenu()

AMIGA FUNKTION
      %
```

1.66 pintui.library/pOS_LayoutMenu()

```
PROTOTYP
      BOOL res = pOS_LayoutMenu
        (
          pOS_IntuiDevice *intuibase,
          pOS_ExList *list,
          const pOS_DrawInfo *drawinfo,
          SLONG x,
          SLONG y
        );
```

```
FUNKTION
      Berechnen der Menüeintragspositionen

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      list (_R_A0)
        Zeiger auf die Liste der Menübeschreibung
      drawinfo (_R_A1)
        Zeiger auf die grafischen Ausgabedaten des Fensters
      x (_R_D0)
        Offset zum Screen, an dem die Menüs geöffnet werden sollen
      y (_R_D1)
        Offset zum Screen, an dem die Menüs geöffnet werden sollen

ERGEBNIS
      res (_R_D0)
        TRUE bei Erfolg, sonst FALSE. Dann liefert pOS_GetIoErr()
        die Fehlerursache.

SIEHE AUCH
      pOS_PreLayoutMenu(), pOS_CreateMenuTagA(), pOS_SetWindowMenu()

AMIGA FUNKTION
      LONG LayoutMenusA(struct Menu *menu, APTR visualinfo, struct TagItem * ↩
         taglist);
```

1.67 pintui.library/pOS_CreateMenuTagA()

```
PROTOTYP
      __ARID__ pOS_MenuList *res = pOS_CreateMenuTagA
        (
          pOS_IntuiDevice *intuibase,
          const pOS_DrawInfo *drawinfo,
          const pOS_MenuTag *menutag,
          ULONG *reserved
        );

FUNKTION
      Anlegen und Installieren eines Menüeintrages

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      drawinfo (_R_A0)
        Zeiger auf die grafischen Ausgabedaten des Fensters
      menutag (_R_A1)
        Zeiger auf die Beschreibung des neuen Menüpunktes

        mt_Type (enum pOS_MenuTagType)
          MENUTAGTYP_End      - Endmarkierung des Arrays
          MENUTAGTYP_Title    - Menü-Titel (obere Leiste)
          MENUTAGTYP_Item     - Menüeintrag (auswählbares Feld)
          MENUTAGTYP_Sub      - Unter-Menüeintrag (auswählbares Feld)
```

```
          MENUTAGTYP_ItemBar  - Trennlinie im Menüeintrag
          MENUTAGTYP_SubBar   - Trennlinie im Unter-Menüeintrag

        mt_Lable - Text-String
        mt_CommKeyMenuItem - Ein Byte Test-String für den ShortKey
        mt_Flags (enum pOS_MenuItemFlags)
          MENUITF_Disabled  - Menü ist nicht wählbar
          MENUITF_Hook      - Menu ist bzw. kann abgehakt werden
          MENUITF_Toggle    - Menu-Haken wechselt autom. ON/OFF
          MENUITF_IsChecked - Haken ist gesetzt

        mt_MutualExclude
          Automatischer Ausschluß von anderen Menü-Eintragen.
          Siehe <pOS_RKRM/pIntui/WinMenu.c>

        mt_Tags - optionale Parameter


      reserved (_R_A2)
        für zukünftige Erweiterungen; z.Z. NULL

ERGEBNIS
      res (_R_D0)
        Zeiger auf den erzeugten Menüeintrag,
        oder NULL im Fehlerfall.

SIEHE AUCH
      pOS_DeleteMenuTag(), pOS_LayoutMenu()

AMIGA FUNKTION
      struct Menu *CreateMenusA(struct NewMenu *newmenu, struct TagList *taglist ↩
         );
```

1.68 pintui.library/pOS_DeleteMenu()

```
PROTOTYP
      VOID pOS_DeleteMenu
        (
          pOS_IntuiDevice *intuibase,
          __ARID__ pOS_MenuList *list
        );

FUNKTION
      Freigeben von Menüeinträgen

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      list (_R_A0)
        Zeiger auf die von pOS_CreateMenuTagA() erzeugte
        Menübeschreibung
        ACHTUNG: nach der Funktion darf dieser Zeiger nicht mehr
                 verwendet werden
```

```
SIEHE AUCH
      pOS_CreateMenuTagA()

AMIGA FUNKTION
      VOID FreeMenus(struct Menu *menu);
```

1.69 pintui.library/pOS_SetMenuHighLight()

```
PROTOTYP
      ULONG res = pOS_SetMenuHighLight
        (
          pOS_IntuiDevice *intuibase,
          pOS_MenuDraw *menudraw,
          pOS_MenuItem *menuitem
        );

FUNKTION
      Systembenutzte Funktion, um den aktiven Menüpunkt hervorzuheben

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      menudraw (_R_A0)
        Zeiger auf die neue Beschreibung des Menüpunktes
      menuitem (_R_A1)
        Zeiger auf den aktiv zu zeichnenden Menüpunkt

ERGEBNIS
      res (_R_D0)
        Liefert den Fehlercode (laut pOS_GetIoErr())
        oder 0 für alles ok.

SIEHE AUCH
      pOS_CreateMenuTagA()

AMIGA FUNKTION
      %
```

1.70 pintui.library/pOS_MenuHandleIEvent()

```
PROTOTYP
      ULONG res = pOS_MenuHandleIEvent
        (
          pOS_IntuiDevice *intuibase,
          pOS_MenuDraw *menudraw,
          const pOS_InputEvent *inputevent
        );

FUNKTION
      Systembenutzte Funktion zum Bearbeiten von InputEvents für ein Menü
```

```
PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      menudraw (_R_A0)
        Zeiger auf die pOS_MenuDraw-Daten, wie sie bei pOS_OpenHandleMenuIEvent ↩
            ()
        angegeben wurden
      inputevent (_R_A1)
        Beschreibung des InputEvents, der Bearbeitet werden soll

ERGEBNIS
      res (_R_D0) (enum pOS_MenuClassResult)
        Liefert den Status
        MCLMTHR_None      : keine Aktion
        MCLMTHR_Activate  : Menüpunkt aktivieren
        MCLMTHR_AbortIE   : InputEvent wird nicht weitergereicht
        MCLMTHR_InputDone : InputHandle beenden

SIEHE AUCH
      pOS_OpenHandleMenuIEvent()

AMIGA FUNKTION
      %
```

1.71 pintui.library/pOS_OpenHandleMenuIEvent()

```
PROTOTYP
      BOOL res = pOS_OpenHandleMenuIEvent
        (
          pOS_IntuiDevice *intuibase,
          pOS_MenuDraw *menudraw
        );

FUNKTION
      Systembenutzte Funktion zum Öffnen eines Input-Handlers für Menüs

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      menudraw (_R_A0)
        Zeiger auf die pOS_MenuDraw-Daten

ERGEBNIS
      res (_R_D0)
        TRUE wenn der Handler erzeugt werden konnte,
        sonst FALSE. Dann liefert pOS_GetIoErr() die Fehlerursache.

SIEHE AUCH
      pOS_CloseHandleMenuIEvent(), pOS_MenuHandleIEvent()

AMIGA FUNKTION
      %
```

1.72 pintui.library/pOS_CloseHandleMenuIEvent()

```
PROTOTYP
      VOID pOS_CloseHandleMenuIEvent
        (
          pOS_IntuiDevice *intuibase,
          pOS_MenuDraw *menudraw
        );

FUNKTION
      Systembenutzte Funktion zum Schließen des Input-Handler für Menüs

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      menudraw (_R_A0)
        Zeiger auf die pOS_MenuDraw-Daten, wie sie bei pOS_OpenHandleMenuIEvent ↩
            ()
        angegeben wurden

SIEHE AUCH
      pOS_OpenHandleMenuIEvent()

AMIGA FUNKTION
      %
```

1.73 pintui.library/pOS_GetMenuItemFromNum()

```
PROTOTYP
      pOS_MenuItem *res = pOS_GetMenuItemFromNum
        (
          pOS_IntuiDevice *intuibase,
          const pOS_ExList *list,
          const pOS_MenuNum *number
        );

FUNKTION
      Ermitteln der Adresse eines Menüpunktes anhand seiner
      Positionsangabe in der Beschreibung.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      list (_R_A0)
        Zeiger auf die rohe Menüliste
      number (_R_A1)
        Beschreibung des Menüpunktes, dessen Adresse ermittelt werden soll

        men_Pck[ MENNUPCK_Title ] => Nummer des Menü-Titel (0,1,2,...)
                                     ~0 => kein Menü-Titel gewählt
        men_Pck[ MENNUPCK_Item ]  => Nummer des Menü-Item (0,1,2,...)
                                     ~0 => kein Menü-Item gewählt
```

```
        men_Pck[ MENNUPCK_Sub ]   => Nummer des Menü-Sub (0,1,2,...)
                                     ~0 => kein Menü-Sub gewählt

ERGEBNIS
      res (_R_D0)
        Ermittelte Adresse des Menüpunktes oder
        NULL wenn der Eintrag nicht vorhanden ist

SIEHE AUCH
      pOS_GetMenuNumFromItem()

AMIGA FUNKTION
      UWORD MENUNUM(UWORD number);
```

1.74 pintui.library/pOS_GetMenuNumFromItem()

```
PROTOTYP
      VOID pOS_GetMenuNumFromItem
        (
          pOS_IntuiDevice *intuibase,
          const pOS_MenuItem *menuitem,
          pOS_MenuNum *number
        );

FUNKTION
      Ermitteln der Position eines Menüpunktes anhand seiner Adresse
      in der Menüliste

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      menuitem (_R_A0)
        Zeiger auf den Menüeintrag, dessen Positionsangabe ermittelt
        werden soll
      number (_R_A1)
        nimmt das ermittelte Ergebnis auf

SIEHE AUCH
      pOS_GetMenuItemFromNum()

AMIGA FUNKTION
      %
```

1.75 pintui.library/pOS_SetWindowMenuChecker()

```
PROTOTYP
      BOOL res = pOS_SetWindowMenuChecker
        (
          pOS_IntuiDevice *intuibase,
          const pOS_Window *window,
```

```
          const pOS_MenuNum *menunum,
          BOOL set
        );

FUNKTION
      Setzen/Entfernen des Hakens für einen Menüpunkt

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, in dem sich das Menü befindet
      menunum (_R_A1)
      set (_R_D0)
        TRUE für abgehakt
        FALSE für nicht abgehackt

ERGEBNIS
      res (_R_D0)
        TRUE wenn der Punkt geändert werden konnte, sonst FALSE.

SIEHE AUCH
      pOS_GetWindowMenuChecker()

AMIGA FUNKTION
      %
```

1.76 pintui.library/pOS_GetWindowMenuChecker()

```
PROTOTYP
      SLONG res = pOS_GetWindowMenuChecker
        (
          pOS_IntuiDevice *intuibase,
          const pOS_Window *window,
          const pOS_MenuNum *number
        );

FUNKTION
      Ermittelt, ob ein Menüpunkt abgehakt ist

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, in dem sich das Menü befindet
      number (_R_A1)
        Positionsangabe des Menüpunktes, der geprüft werden soll

ERGEBNIS
      res (_R_D0)
        TRUE wenn der angegebene Menüpunkt abgehakt ist,
        sonst FALSE

SIEHE AUCH
```

```
      pOS_SetWindowMenuChecker()

AMIGA FUNKTION
      %
```

1.77 pintui.library/pOS_EnableWindowMenu()

```
PROTOTYP
      BOOL res = pOS_EnableWindowMenu
        (
          pOS_IntuiDevice *intuibase,
          const pOS_Window *window,
          const pOS_MenuNum *menunum,
          BOOL enable
        );

FUNKTION
      Wählbarkeit eines Menüpunktes ändern

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, in dem sich das Menü befindet
      menunum (_R_A1)
        Zeiger auf den Menüpunkt, dessen Wählbarkeit geändert
        werden soll
      enable (_R_D0)
        TRUE damit der Menüpunkt wählbar wird,
        FALSE damit der Punkt gesperrt wird.
        Bei Haupt-Menüs werden alle Menüpunkte gesperrt,
        bei normalen Menüs alle evtl. vorhandenen Sub-Menüs.

ERGEBNIS
      res (_R_D0)
        TRUE wenn der gewünschte Zustand hergestellt werden
        konnte; sonst FALSE.

SIEHE AUCH
      pOS_SetWindowMenu()

AMIGA FUNKTION
      %
```

1.78 pintui.library/pOS_CreateBubbleHelpA()

```
PROTOTYP
      __ARID__ pOS_BubbleHelp *res = pOS_CreateBubbleHelpA
        (
          pOS_IntuiDevice *intuibase,
```

```
          const pOS_TagItem *tagitem
        );

FUNKTION
      Systembenutzte Funktion zum Erzeugen einer BubbleHelp-Struktur
      anhand einer Beschreibung

PARAMETER
          intuibase (_R_LB)
            Zeiger auf pIntui-Library
          tagitem (_R_A0) (enum pOS_IntuiBubTags)
            BUBTAG_Screen  : zugehöriger Screen
      ~     BUBTAG_Text    : Hilfe-Text
            BUBTAG_HotSpot : Referenzposition
            BUBTAG_Type    : BUBTYP_Point oder BUBTYP_Round

ERGEBNIS
      res (_R_D0)
        Zeiger auf die erzeugte Struktur, die mittels pOS_DeleteBubbleHelp()
        wieder freigegeben werden muß; oder NULL im Fehlerfall.

SIEHE AUCH
      pOS_DeleteBubbleHelp()

AMIGA FUNKTION
      %
```

1.79 pintui.library/pOS_DeleteBubbleHelp()

```
PROTOTYP
      VOID pOS_DeleteBubbleHelp
        (
          pOS_IntuiDevice *intuibase,
          __ARID__ pOS_BubbleHelp *bubblehelp
        );

FUNKTION
      Systembenutzte Funktion zum Freigeben einer BubbleHelp-Struktur

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      bubblehelp (_R_A0)
        Zeiger auf die freizugebende BubbleHelp-Struktur, wie er
        von pOS_CreateBubbleHelpA() geliefert wurde.

SIEHE AUCH
      pOS_CreateBubbleHelpA()

AMIGA FUNKTION
      %
```

1.80 pintui.library/pOS_CreateRequestWinA()

```
PROTOTYP
      pOS_Window *res = pOS_CreateRequestWinA
        (
          pOS_IntuiDevice *intuibase,
          const pOS_TagItem *tagitem
        );

FUNKTION
      Erzeugen und Anzeigen eines Standard-Dialogfensters

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      tagitem (_R_A0) (enum pOS_EasyTags)

        EASYTAG_ButLine (const CHAR*)
          Button-String, wobei die einzelnen Gadget mit '|' getrennt werden.
          Z.B. "Ok|Suchen|Abbrechen" => erzeugt 3 Gadget: "Ok" "Suchen" " ↩
             Abbrechne"
          Ein ein '_' mitangegeben, so wird das Gadget mit einem ShortKey
          ausgestattet.
          Bei der Nachrichtenauswertung hat das links stehende Gadget immer den
          Wert 1 (TRUE) und das ganz rechts stehende Gadget immer
          den Wert 0 (FALSE). Weitere Gadget bekommen einen aufsteigenden Wert.
          (oben: "Suchen" == 2)

        EASYTAG_BodyLine (const CHAR*)
          Anzuzeigender Erklärungstext. Der String kann mit 'LF' umgebrochen
          werden.

        EASYTAG_ES (const pOS_EasyStruct*)
          Requester-Daten aus der Standard-Struktur lesen oder NULL:

        EASYTAG_Args (const ULONG*)
          Argument-Array für 'RawDoFmt'.
          Zuerst wird EASYTAG_BodyLine interpretiert und danach EASYTAG_ButLine.

        EASYTAG_Gadget (pOS_Gadget*)
          Mehrere oder NULL-Zeiger auf bereits erzeugte Gadget.
          Alle so eingebundenen Gadgets werden vertikal aufgereiht
          und das Window wird Re-Sizeable. Sämtliche Gadgets werden
          NICHT autom. beim Window-Close gelöscht.

        EASYTAG_ActivateGadget (pOS_Gadget*)
          Zu aktivierendes Gadget beim OpenWindow.

        EASYTAG_WindowBusy (struct pOS_Window*)
          Window, das 'busy' werden soll.
          NULL => wird ignoriert.

        EASYTAG_Type (enum pOS_EasyTypes)
          Art den Requester.
```

```
        EASYTAG_GfxFile (const dosname_t*)
          Anzuzeigende Grafik.

        EASYTAG_OkGadget (const pOS_Gadget*)
          Gadget, das Requester-Ok auslösen soll.
          ~0 => Gadget-ID wird ausgelesen (>0 => OK)

        EASYTAG_CancelGadget (const pOS_Gadget*)
          Gadget, das Requester-Cancel auslösen soll.

        EASYTAG_Title (const CHAR*)
          Window-Titel im Requester

        EASYTAG_Screen (pOS_Screen*)
          Screen, auf dem der Requester geöffnet werden soll.

        SCRTAG_PubName (const CHAR*)
          PubScreen, auf dem der Requester geöffnet werden soll.

        SCRTAG_IDCMP (ULONG)
          IDCMP-Message, auf die der Requester reagieren soll.
          Default = IDCMP_GadgetUp | IDCMP_VanillaKey | IDCMP_IntuiTicks;


ERGEBNIS
      res (_R_D0)
        Zeiger auf die erzeugte Fensterstruktur für den Dialog-
        Requester oder NULL im Fehlerfall (Speichermangel).

SIEHE AUCH
      pOS_DeleteRequestWin(), pOS_EasyRequestArgs()

AMIGA FUNKTION
      struct Window *BuildEasyRequestArgs(struct Window *window, struct  ↩
         EasyStruct *easystruct, ULONG idcmpflags, APTR args);
      struct Window *BuildSysRequest(struct Window *window, struct IntuiText * ↩
         bodytext,
                     struct IntuiText *positivetext, struct IntuiText * ↩
                        negativetext,
                     ULONG flags, LONG width, LONG height);
```

1.81 pintui.library/pOS_DeleteRequestWin()

```
PROTOTYP
      VOID pOS_DeleteRequestWin
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window
        );

FUNKTION
      Schließen und Entfernen eines Standard-Dialogfensters

PARAMETER
```

```
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf die Fensterdaten, die von pOS_CreateRequestWinA()
        geliefert wurde.

SIEHE AUCH
      pOS_CreateRequestWinA()

AMIGA FUNKTION
      VOID FreeSysRequest(struct Window *window);
```

1.82 pintui.library/pOS_EasyRequestArgs()

```
PROTOTYP
      SLONG res = pOS_EasyRequestArgs
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          const pOS_EasyStruct *easystruct,
          ULONG *idcmp,
          const ULONG *array
        );

FUNKTION
      Ausgeben eines Standard-Dialogfenster und warten auf die
      Benutzerbestätigung

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Fenster, zu dem der Easy-Requester gehört,
        oder NULL für den Vorgabe-Public-Screen.
      easystruct (_R_A1)
        Zeiger auf die Beschreibungsdaten des Requesters
      idcmp (_R_A2)
        Zeiger auf IDCMP-Flags, die ein schließen des Requesters
        verursachen können oder NULL.

         es_StructSize - sizeof(struct pOS_EasyStruct) */
         es_Flags - 0
         es_Type - (enum pOS_EasyTypes) (EASYTAG_Type)
         es_Title - Title (EASYTAG_Title)
         es_TextFormat - (EASYTAG_BodyLine)
         es_GadgetFormat - (EASYTAG_ButLine)
         es_HelpID - Window-HelpID, NULL => default
         es_HelpFile - Window-HelpFile, NULL => default
         es_Tags - (enum pOS_EasyTags) oder NULL

      array (_R_A3)
        Argumente, die entsprechend easystruct->es_TextFormat
        ausgewertet und eingesetzt werden.
```

```
ERGEBNIS
      res (_R_D0)
        ID des angeklickten Schalters: von links ab 1, der rechteste
        (normal Abbruch oder Nein) 0.
        Genauere Infos bei pOS_RequestWinHandler().

SIEHE AUCH
      pOS_CreateRequestWinA(), pOS_RequestWinHandler()

AMIGA FUNKTION
      LONG EasyRequestArgs(struct Widow *window, struct EasyStruct *easystruct,  ↩
         ULONG *idcmpflags, APTR args);
```

1.83 pintui.library/pOS_RequestWinHandler()

```
PROTOTYP
      SLONG res = pOS_RequestWinHandler
        (
          pOS_IntuiDevice *intuibase,
          pOS_Window *window,
          ULONG *idcmp,
          BOOL wait
        );

FUNKTION
      Bearbeiten von Eingaben eines Standard-Dialogfensters

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      window (_R_A0)
        Zeiger auf das Dialogfenster, wie er von pOS_CreateRequestWinA()
        geliefert wurde
      idcmp (_R_A1)
        Zeiger auf IDCMP-Flags, die ein schließen des Requesters
        verursachen können oder NULL.
      wait (_R_D0)
        TRUE wenn auf auf eine Nachricht gewartet werden soll,
        FALSE wenn die Funktion sofort zurückkehren soll.

ERGEBNIS
      res (_R_D0)
        ID des angeklickten Schalters: von links ab 1, der rechteste
        (normal Abbruch oder Nein) 0, oder -1 wenn ein gesetztes IDCMP-
        Signal aufgetreten ist. Dann enthält die idcmp-Variable das
        eingetroffene Signal.
        -2 wenn keine Nachricht vorliegt (bei wait = FALSE)
        Bei Speichermangel wird ebenfalls 0 (wie bei Abbruch) zurückgegeben.

SIEHE AUCH
      pOS_CreateRequestWinA(), pOS_DeleteRequestWin(),
      pOS_EasyRequestArgs()

AMIGA FUNKTION
```

```
      LONG SysReqHandler(struct Window *window, ULONG *idcmpflags, LONG wait);
```

1.84 pintui.library/pOS_AddDrag()

```
PROTOTYP
      VOID pOS_AddDrag
        (
          pOS_IntuiDevice *intuibase,
          pOS_IntuiUnit *unit,
          pOS_Drag *drag
        );

FUNKTION
      Systembenutzte Funktion zum

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      unit (_R_A0)
        Zeiger auf die pIntui-Unit
      drag (_R_A1)

SIEHE AUCH
      pOS_RemDrag();

AMIGA FUNKTION
      %
```

1.85 pintui.library/pOS_RemDrag()

```
PROTOTYP
      VOID pOS_RemDrag
        (
          pOS_IntuiDevice *intuibase,
          pOS_IntuiUnit *unit,
          pOS_Drag *drag
        );

FUNKTION
      Systembenutzte Funktion zum

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      unit (_R_A0)
        Zeiger auf die pIntui-Unit
      drag (_R_A1)

SIEHE AUCH
      pOS_AddDrag()
```

```
AMIGA FUNKTION
      %
```

1.86 pintui.library/pOS_CreateDragA()

```
PROTOTYP
      __ARID__ pOS_Drag *res = pOS_CreateDragA
        (
          pOS_IntuiDevice *intuibase,
          const pOS_TagItem *taglist
        );

FUNKTION
      Das Gadget kann bei 'GCLMTH_BeginDrag' ein Drag-Objekt dem System
      bereitstellen.

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      taglist (_R_A0) (enum pOS_IntuiDragTags)

        DRAGTAG_Render (pOS_IntuiObj*)
          Dieser Objekt wird beim Drag angezeigt und sollte das verschiebende
          Daten-Objket symbolisieren.

        DRAGTAG_LeftEdge (SLONG)
          Offset auf das IObjekt.
          'imth_BDrag.imbd_AMouse' => Maus klebt an der linke obere Ecke.

        DRAGTAG_TopEdge (SLONG) - Offset auf das IObjekt
        DRAGTAG_GadDrag (pOS_GadgetDrag*)
          Attribute, normalerweise => ( Gadget->gad_Drag )

        DRAGTAG_AutoDT (BOOL)
          Der dazugehörige DtType wird autom. über das IObjekt erzeugen.
          (Nicht alle IObjekte lassen diese Vorgehensweise zu.)

        DRAGTAG_DeleteRender (BOOL)
          Das bei DRAGTAG_Render gesetzte IObjekt wird am Ende autom.
          gelöscht. (pOS_DisposeIObject();)



ERGEBNIS
      res (_R_D0)
        Zeiger auf das erzeugte Drag-Objekt, das mittels pOS_DeleteDrag()
        wieder freigegeben werden muß; oder NULL im Fehlerfall.
        Normalerweise gibt der Gadget-Dispatcher das Objekt in
        'imth_BDrag.imbd_Result' zurück.

SIEHE AUCH
      pOS_DeleteDrag()
```

```
AMIGA FUNKTION
      %
```

1.87 pintui.library/pOS_DeleteDrag()

```
PROTOTYP
      VOID pOS_DeleteDrag
        (
          pOS_IntuiDevice *intuibase,
          __ARID__ pOS_Drag *drag
        );

FUNKTION
      Systembenutzte Funktion zum

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      drag (_R_A0)
        Zeiger auf das zu löschende Drag-Objekt, wie er von
        pOS_CreateDragA() geliefert wurde.
        ACHTUNG: nach dieser Funktion darf der Zeiger nicht mehr
                 benutzt werden.

SIEHE AUCH
      pOS_CreateDragA()

AMIGA FUNKTION
      %
```

1.88 pintui.library/pOS_CreateDragGfx()

```
PROTOTYP
      BOOL res = pOS_CreateDragGfx
        (
          pOS_IntuiDevice *intuibase,
          pOS_RastPort *rastport,
          pOS_Drag *drag
        );

FUNKTION
      Systembenutzte Funktion zum

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      rastport (_R_A0)
        Zeiger auf den RastPort, in dem das Drag-Objekt gezeichnet
        werden soll
      drag (_R_A1)
```

```
        Drag-Objekt, für das eine grafische Anzeige geschaffen
        werden soll

ERGEBNIS
      res (_R_D0)
        TRUE wenn das grafische Drag-Objekt erzeugt wurde;
        sonst FALSE.

SIEHE AUCH
      pOS_DeleteDragGfx(), pOS_CreateDragListGfx()

AMIGA FUNKTION
      %
```

1.89 pintui.library/pOS_DeleteDragGfx()

```
PROTOTYP
      VOID pOS_DeleteDragGfx
        (
          pOS_IntuiDevice *intuibase,
          pOS_Drag *drag
        );

FUNKTION
      Systembenutzte Funktion zum

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      drag (_R_A0)
        Zeiger auf das zu Drag-Objekt, von dem die grafische
        Ausgabe entfernt werden soll;
        Zeiger wie er von pOS_CreateDragGfx() geliefert wurde

SIEHE AUCH
      pOS_CreateDragGfx(), pOS_DeleteDragListGfx()

AMIGA FUNKTION
      %
```

1.90 pintui.library/pOS_ConstructDragList()

```
PROTOTYP
      VOID pOS_ConstructDragList
        (
          pOS_IntuiDevice *intuibase,
          pOS_DragList *draglist
        );

FUNKTION
```

```
      Systembenutzte Funktion zum

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      draglist (_R_A0)

SIEHE AUCH
      pOS_DestructDragList()

AMIGA FUNKTION
      %
```

1.91 pintui.library/pOS_DestructDragList()

```
PROTOTYP
      VOID pOS_DestructDragList
        (
          pOS_IntuiDevice *intuibase,
          pOS_DragList *draglist
        );

FUNKTION
      Systembenutzte Funktion zum

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      draglist (_R_A0)
        Zeiger wie er bei pOS_ConstructDragList() verwendet wurde

SIEHE AUCH
      pOS_ConstructDragList()

AMIGA FUNKTION
      %
```

1.92 pintui.library/pOS_LayoutDragList()

```
PROTOTYP
      BOOL res = pOS_LayoutDragList
        (
          pOS_IntuiDevice *intuibase,
          pOS_DragList *draglist,
          pOS_RastPort *rastport
        );

FUNKTION
      Systembenutzte Funktion zum
```

```
PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      draglist (_R_A0)
      rastport (_R_A1)

ERGEBNIS
      res (_R_D0)
        TRUE wenn ;
        sonst FALSE.

SIEHE AUCH
      pOS_ConstructDragList(), pOS_CreateDragListGfx(),
      pOS_RenderDragList()

AMIGA FUNKTION
      %
```

1.93 pintui.library/pOS_CreateDragListGfx()

```
PROTOTYP
      BOOL res = pOS_CreateDragListGfx
        (
          pOS_IntuiDevice *intuibase,
          pOS_DragList *draglist,
          pOS_RastPort *rastport
        );

FUNKTION
      Systembenutzte Funktion zum

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      draglist (_R_A0)
      rastport (_R_A1)

ERGEBNIS
      res (_R_D0)
        TRUE wenn ;
        sonst FALSE.

SIEHE AUCH
      pOS_DeleteDragListGfx()

AMIGA FUNKTION
      %
```

1.94 pintui.library/pOS_DeleteDragListGfx()

```
PROTOTYP
      VOID pOS_DeleteDragListGfx
        (
          pOS_IntuiDevice *intuibase,
          pOS_DragList *draglist
        );

FUNKTION
      Systembenutzte Funktion zum

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      draglist (_R_A0)

SIEHE AUCH
      pOS_CreateDragListGfx()

AMIGA FUNKTION
      %
```

1.95 pintui.library/pOS_RenderDragList()

```
PROTOTYP
      VOID pOS_RenderDragList
        (
          pOS_IntuiDevice *intuibase,
          pOS_DragList *draglist,
          ULONG mode
        );

FUNKTION
      Systembenutzte Funktion zum

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      draglist (_R_A0)
      mode (_R_D0)
        Zeichenmodus

SIEHE AUCH
      pOS_LayoutDragList()

AMIGA FUNKTION
      %
```

1.96 pintui.library/pOS_MoveDragList()

```
PROTOTYP
      VOID pOS_MoveDragList
        (
          pOS_IntuiDevice *intuibase,
          pOS_DragList *darglist,
          SLONG dx,
          SLONG dy,
          ULONG mode
        );

FUNKTION
      Systembenutzte Funktion zum

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      draglist (_R_A0)
      dx (_R_D0)
        horizontale Verschiebung
      dy (_R_D1)
        vertikale Verschiebung
      mode (_R_D2)
        zeichenmodus

SIEHE AUCH

AMIGA FUNKTION
      %
```

1.97 pintui.library/pOS_RestoreDragList()

```
PROTOTYP
      VOID pOS_RestoreDragList
        (
          pOS_IntuiDevice *intuibase,
          pOS_DragList *draglist
        );

FUNKTION
      Systembenutzte Funktion zum

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      draglist (_R_A0)

SIEHE AUCH

AMIGA FUNKTION
      %
```

1.98 pintui.library/pOS_SaveDragList()

```
PROTOTYP
      VOID pOS_SaveDragList
        (
          pOS_IntuiDevice *intuibase,
          pOS_DragList *draglist
        );

FUNKTION
      Systembenutzte Funktion zum

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      draglist (_R_A0)

SIEHE AUCH

AMIGA FUNKTION
      %
```

1.99 pintui.library/pOS_GetDropAttFromDragList()

```
PROTOTYP
      const pOS_DropAttribute *res = pOS_GetDropAttFromDragList
        (
          pOS_IntuiDevice *intuibase,
          pOS_DragList *draglist,
          const pOS_GadgetDrop *gadgetdrop,
          const pOS_DropAttribute *dropattribute
        );

FUNKTION
      Systembenutzte Funktion zum

PARAMETER
      intuibase (_R_LB)
        Zeiger auf pIntui-Library
      draglist (_R_A0)
      gadgetdrop (_R_A1)
      dropattribute (_R_A2)

ERGEBNIS
      res (_R_D0)

SIEHE AUCH

AMIGA FUNKTION
      %

END
```