UObject Classes:

Class	Fields
TObject	
TCollection	size dynStart holeStart holeSize holeStd
TList	
TArray	recordBytes
TString	
TFile	path password scanners
TScanner	collection position increment scanDone atEnd
TListScanner	
TArrayScanner	
TStringScanner	actual
TFileScanner	accesses refnum error

UDraw Classes:

Class	Fields
TObject	*(defined in UObject)*
TArea	innerRect outerRect parentBranch
TPad	port viewedLRect visLRect availLRect scrollOffset origin cdOffset clippedRect padRes viewedRes scaled scaleFactor zoomFactor
TBranchArea	arrangement elderFirst resizability elderChild youngerChild

UABC Classes:

Class	**Fields**
TObject	*(defined in UObject)*
TProcess	*(global variables)*
TDocDirectory	window classWorld
TDocManager	files dataSegment docHeap window pendingNote openedAsTool
TClipboard	hasView hasPicture hasUniversalText hasIcon cuttingTool cuttingProcessID clipCopy
TCommand	cmdNumber image undoable doing revelation unHiliteBefore hiliteAfter
TCutCopyCommand	isCut
TPasteCommand	
TImage	extentLRect view allowMouseOutside
TView	panel clickLPt printManager res screenPad fitPagesPerfectly isPrintable isMainView stdScroll scrollPastEnd
TPaginatedView	unpaginatedView pageSize workingInMargins
TPageView	
THeading	printManager pageAlignment offsetFromAlignment oddOnly evenOnly minPage maxPage
TPrintManager	view pageView breaks pageMargins headings canEditPages layoutDialogBox frameBody paperLRect printableLRect contentLRect printerMetrics pageRiseDirection
TSelection	window panel view kind anchorLPt currLPt boundLRect coSelection canCrossPanels
TArea	*(defined in UDraw)*
TWindow	panels panelTree dialogBox selectPanel undoSelPanel clickPanel undoClickPanel selectWindow undoSelWindow wmgrID isResizable believeWmgr maxInnerSize changes lastCmd printerMetrics pgSzOK pgRgOK panelToPrint objectToFree
TDialogBox	keyResponse menuResponse downInMainWindowResponse freeOnDismissal
TBand	window panes panel scroller scrollDir
TSideBand	topOrLeft
TPanel	window panes currentView view paginatedView selection undoSelection bands scrollBars abilities minInnerDiagonal resizeBranch zoomed zoomFactor previewMode lastClick contentRect tlSideBandSize brSideBandSize deletedSplits
TPad	*(defined in UDraw)*
TPane	currentView panel
TMarginPad	view pageNumber bodyPad
TBodyPad	marginPad nonNullBody
TScroller	scrollBar band sBoxID
TScrollBar	firstBox isVisible
TMenuBar	isLoaded mapping numMenus numCommands

UUnivText Classes:

Class	Fields
TObject	*(defined in UObject)*
TTKUnivText	paragraphDescriptor characterDescriptor maxDataSize data itsOurTString
TTKReadUnivText	buffer columnCount dataBeforeTab
TTKWriteUnivText	

UText Classes:

Class	Fields
TObject	*(defined in UObject)*
TParaFormat	dfltTStyle wordWrap quad firstIndent leftIndent rightIndent spaceAbovePara spaceBelowPara lineSpacing tabs refCount permanent styleSheet
TCollection	*(defined in UObject)*
TString	*(defined in UObject)*
TParagraph	typeStyles
TEditPara	bsCount nestLevel format beingFiltered images
TLineInfo	valid startLP lastDrawnLP endLP lineLRect lineAscent
TImage	*(defined in UABC)*
TParaImage	paragraph height lineList changed tickCount startLP endLP textImage wasOffset
TTextImage	text imageList tickCount growsDynamically minHeight formerBottom updateLRect firstLinePixel useFirstPixel firstIndex startLP endLP prevTxtImg nextTxtImg headTxtImg tailTxtImg
TView	*(defined in UABC)*
TTextView	textImage valid
TStyleSheet	formats
TTextRange	firstPara firstIndex lastPara lastIndex lastLP
TText	paragraphs styleSheet txtImgList
TTKWriteUnivText	*(defined in UUnivText)*
TTextWriteUnivText	textSelection currIndex currPara currLP currStyleIndex currTStyles
TSelection	*(defined in UABC)*
TTextSelection	textImage textRange isWordSelection isParaSelection viewTick amTyping currTypeStyle
TInsertionPoint	typingCmd styleCmdNumber newestLP justReturned nextHighTransit nextTransitTime
TOneParaSelection	anchorBegin anchorEnd
TMultiParaSelection	anchorPara anchorIndex anchorBegin anchorEnd
TCommand	*(defined in UABC)*
TClearTextCmd	savedText text
TStyleCmd	text textSelection firstFiltParaIndex lastFiltParaIndex filtFirstLP filtLastLP currFilteredPara filteredStyles
TCutCopyCommand	*(defined in UABC)*
TTextCutCopy	text
TPasteCommand	*(defined in UABC)*
TTextPaste	savedText pasteRange text origIsPara origIsWord clipIsPara
TTypingCmd	savedText text newCharCount newParaCount typingRange otherInsPts

UDialog Classes:

Class	Fields
TObject	*(defined in UObject)*
TArea	*(defined in UDraw)*
TWindow	*(defined in UABC)*
TDialogBox	*(defined in UABC)*
TDialogWindow	controlPanel dialogView mainDialog
TDialogDesignWindow	hostWindow hostDialogView fromDialogBox
TPageDesignWindow	hostView layoutPanel
TImage	*(defined in UABC)*
TTitleTab	layoutBox legend shouldDrawLegend
THeading	*(defined in UABC)*
TLegendHeading	masterLegend currentLegend topToBaseline borders
TView	*(defined in UABC)*
TDialogView	rootDialog nonDialogExtent currentDialogImage defaultButton hitButton isShowing paintFreeBoxes paintSense startedPainting styleSheet mouseIsDown magnetCursor
TPlannerView	viewBeingPlanned allowSketching retainPickedBox currentLayoutBox
TPagePlannerView	
TDialogImage	parent isActive isEditable withID
TLegend	location paragraph wouldBeDraggable usesSysFont
TRectImage	penState
TImageWithID	children id idNumber
TDialog	stringKey
TPageStatusDialog	currentHeading oddEvenCluster minPageFrame maxPageFrame alignCluster unitsCluster marginTitle leftCluster topCluster rightCluster bottomCluster
TButton	cmdNumber minWidth isHighlighted nextSameSizedButton legend buttonMetrics
TCheckbox	isSelected rectImage legend
TCluster	location hitBox hiLitBox lastBox
TInputFrame	textDialogImage prompt borders drawInputLRect drawHitLRect maxInputChars inputTypeStyle
TPicObject	picture boxAtCreation
TTextDialogImage	textImage wouldBeDraggable refCount
TLayoutBox	manipulee titleTab suppressDrawingManipulee isResizable borders wouldMakeSelection isDraggable shouldFrame hasDraggee
TLegendLayoutBox	textDialogImage
TButtonLayoutBox	nextSameSizedBox oldLegendTopLeft
TPageLayoutBox	
TLgHdngLayoutBox	legendLayoutBox
TSelection	*(defined in UABC)*
TFrameSelection	inputFrame
TLayPickSelection	layoutBox
TEditLegendSelection	legendLayoutBox hostLegend textDialogImage suppressHost tripleClick
TCommand	*(defined in UABC)*
TLayMoveCmd	layoutBox hOffset vOffset
TPrintManager	*(defined in UABC)*
TStdPrintManager	

```
1     1 --  UNIT UObject;
1     2 --  {Copyright 1983, 1984, Apple Computer, Inc.}
1     3 --  {Implementation is in UOBJECT2-3-4}
1     4 --
1     5 --  {$SETC IsIntrinsic := TRUE}
1     6 --
1     7 --  {$IFC IsIntrinsic}
1     8 --  INTRINSIC;
1     9 --  {$ENDC}
1    10 --
1    11 --  {$SETC ErrsToFile := TRUE }
1    12 --
1    13 --  {$IFC ErrsToFile}
1    14 --  {$E+}
1    15 --  {**************************************}     {$E ERRS.TEXT}     {**************************************}
1    16 --  {$ENDC}
1    17 --
1    18 --  {NOTE: The implementation of class TObject is quite obscure because this is actually system-type code}
1    19 --
1    20 --  {Segments: SgABCini(tialize), SgABCdat(a structures), SgABCdbg}
1    21 --
1    22 --  {
1    23 --  ****************************************** SPECIFICALLY IN UObject ******************************************
1    24 --
1    25 --  ----------CLASSES------     ------------------VARIABLES------------------- ------------ COMMENTS ----------
1    26 --
1    27 --    TObject
1    28 --
1    29 --      TCollection             size dynOffset holeStart holeSize holeStd       -- indexed access (At, InsAt,
1    30 --                                                                               Each)
1    31 --          TList                                                                -- contains object handles
1    32 --          TArray              recordBytes                                      -- contains records (even
1    33 --                                                                               lengths)
1    34 --          TString                                                              -- contains characters
1    35 --          TFile               path scanners                                    -- disk file (Exists, Rename)
1    36 --      TScanner                collection position increment scanDone atEnd    -- sequential access (Scan,
1    37 --                                                                               Insert)
1    38 --          TListScanner                                                         -- an object at a time
1    39 --          TArrayScanner                                                        -- a record at a time
1    40 --          TStringScanner      error actual                                     -- a character at a time (Xfer)
1    41 --              TFileScanner    accesses refnum                                  -- through a whole TFile
1    42 --
1    43 --
1    44 --  ************************************** IN ALL DATA STRUCTURE UNITS **************************************
1    45 --
1    46 --                  === KEY ===>   $ = in UObject  @ = in UHuge  * in UDb  # in UMac
1    47 --
1    48 --  ----------CLASSES----------     ---------------VARIABLES------------------- ------------ COMMENTS ----------
1    49 --
1    50 --  $ TObject
1    51 --
1    52 --  $   TCollection                   size dynOffset holeStart holeSize holeStd -- indexed access (At, InsAt,
1    53 --                                                                               Each)
1    54 --  $       TList                                                                -- contains object handles
1    55 --    @         TLinkList             head tail                                  -- stored in TLinks
1    56 --    @         THugeList             hugeArray                                  -- stored in linked blocks
1    57 --  $       TArray                    recordBytes                              -- contains records (even
1    58 --                                                                               lengths)
1    59 --    @         THugeArray            minBlockLength maxBlockLength blocks       -- impl. with linked blocks
1    60 --  $       TString                                                              -- contains characters
1    61 --  $       TFile                     path scanners                            -- disk file (Exists, Rename)
1    62 --    *     TDb                                                                -- contains keyed records
1    63 --    *         TDbFile               file rScanDesc                             -- key is a PAOC/String
1    64 --    *         TRsFile               endIncrement firstKey lastKey scanners     -- key is a LONGINT (SwapIn)
1    65 --    *             TDbRsFile         dbFile                                       -- implemented with a
1    66 --                                                                                  TDbFile
1    67 --          #       TMcRsFile         ???                                          -- implemented in the Mac
1    68 --                                                                                  ROM
1    69 --
1    70 --  $   TScanner                    collection position increment scanDone atEnd -- sequential access (Scan,
1    71 --                                                                               Insert)
1    72 --  $       TListScanner                                                         -- an object at a time
1    73 --    @         TLnkLstScanner        scanLink
1    74 --    @         THgeLstScanner        blkArrScanner
1    75 --  $       TArrayScanner                                                        -- a record at a time
1    76 --    @         THgeArrScanner        cacheBlock cacheIndex                      -- through a THugeArray
1    77 --  $       TStringScanner            error actual                             -- a character at a time (Xfer)
1    78 --  $           TFileScanner          accesses refnum                          -- through a whole TFile
1    79 --    *         TRsScanner            whichWay key buffer                        -- through a single resource
1    80 --    *     TDbScanner                error                                    -- a key at a time
1    81 --    *         TDbFiScanner          rScanDesc                                  -- through a TDbFile
1    82 --    *         TRsFiScanner                                                     -- a resource at a time
1    83 --    *             TDbRsFiScanner    dbRecSeq dbRecSize                           -- through a TDbRsFile
1    84 --          #       TMcRsFiScanner    ???                                          -- implemented in the Mac
1    85 --                                                                                  ROM
1    86 --
1    87 --    @ TLink                         element next                             -- has one element of a TLinkList
1    88 --  }
1    89 --
1    90 --  INTERFACE
1    91 --  {$SETC LibraryVersion := 30 } { 10 = 1.0 libraries; 13 = 1.3 libraries; 20 = Pepsi,
1    92 --                                  29 = V12.0 Libraries, 30 = V13.0+ libraries }
1    93 --  {$SETC compatibleLists := FALSE }
1    94 --
1    95 --  USES
1    96 --
1    97 --      UnitStd,
1    98 --      UnitHz,
1    99 --      {$U LIBOS/SysCall    } SysCall,
1   100 --  {$IFC LibraryVersion > 20}
1   101 --      {$U LIBTK/Passud   } Passud,
1   102 --  {$ENDC}
1   103 --  {$IFC LibraryVersion <= 20}
1   104 --      {$U UClascal}           UClascal,
1   105 --  {$ELSEC} {$IFC LibraryVersion < 30}
1   106 --      {$U LIBTK/UClascal}     UClascal,     {Needed for interface}
1   107 --  {$ELSEC}
1   108 --      {$U LIBPL/UClascal}     UClascal,     {Needed for interface}
1   109 --  {$ENDC}
1   110 --  {$ENDC}
```

```
1  111 --          { The next units needed to find out where the printer is located, from parameter memory,
1  112 --            so we can tell Paslib where it is. (Needed for debugger Output Redirect.) }
1  113 --        PwDecl,
1  114 --        Pwm,
1  115 --    {$IFC LibraryVersion > 10}
1  116 --        {$U LIBPL/PaslibCall} PaslibCall,
1  117 --        {$U LIBPL/PPasLibc } PPasLibC,
1  118 --    {$ENDC}
1  119 --
1  120 --      - {$U HWInt}              HWInt;
1  121 --
1  122 --
1  123 --    {$SETC fDbgOK := TRUE}{FALSE} {override UnitStd to test Tool Kit}
1  124 --    {$SETC fSymOK := TRUE}{FALSE} {override UnitStd to test Tool Kit}
1  125 --
1  126 --    {$SETC fDbgObject    := fDbgOK}
1  127 --    {$SETC fRngObject    := fDbgOK}
1  128 --    {$SETC fSymObject    := fSymOK}
1  129 --
1  130 --    {$SETC fDebugMethods := fDbgObject} {include debugging methods in the compilation}
1  131 --
1  132 --    {$SETC fCheckHeap    := fDbgObject} {if VAR also true, check heap}
1  133 --    {$SETC fTrace        := fDbgObject} {if VAR also true, trace entries/exits}
1  134 --    {$SETC fMaxTrace     := fTrace AND FALSE} {if TRUE trace entries/exits on minor procedures too}
1  135 --
1  136 --    {$SETC fCheckIndices := fDbgObject} {if VAR also true, check subscripts}
1  137 --
1  138 --    CONST
1  139 --
1  140 --        prcsLdsn         = 1;              {ldsn for the process data segment}
1  141 --        prcsDsBytes      = 15000;          {default heap size for a process data segment}
1  142 --
1  143 --        MaxBreaks        = 10;
1  144 --
1  145 --        outputRMargin    = 85;
1  146 --        erInternal       = 4200;           {Stolen from list of errors in UABC for newHeap}
1  147 --
1  148 --        MAXLINT          = $7FFFFFFF;
1  149 --
1  150 --    TYPE
1  151 --
1  152 --      {Aliases needed to compile QuickDraw}
1  153 --
1  154 --        Ptr = ^LONGINT;
1  155 --        ProcPtr = Ptr;
1  156 --        Handle = ^Ptr;
1  157 --
1  158 --      {Aliases for commonly used types}
1  159 --
1  160 --        SB = STRING[8];
1  161 --        S255 = STRING[255];
1  162 --
1  163 --        TFilePath = S255;    {Increased from 66 because of the new hierarchical file system;
1  164 --                              corresponds to Pathname in SYSCALL}
1  165 --        TFilePart = STRING[32]; {length of each level in a pathname; corresponds to e_name in SYSCALL}
1  166 --        TPassword = TFilePart;
1  167 --
1  168 --        THeap = Ptr;      {alias for THz in UnitHz}
1  169 --        TClass = Ptr;     {alias for TPSliceTable in UClascal}
1  170 --
1  171 --        Byte = -128..127;
1  172 --        TPString = ^S255;
1  173 --
1  174 --        TpINTEGER = ^INTEGER;
1  175 --        TpLONGINT = ^LONGINT;
1  176 --
1  177 --        TAuthorName = STRING[32];
1  178 --        TClassName  = STRING[8];
1  179 --
1  180 --        TClassWorld = RECORD     {Alias for TWorld in IMPLEMENTATION}
1  181 --            infRecs:     TArray {OF name, size, author, & version information};
1  182 --            classes:     TArray {OF TClass -- the pointer in each Clascal object};
1  183 --            authors:     TArray {OF PACKED ARRAY [1..SIZEOF(TAuthorName)] OF CHAR};
1  184 --            aliases:     TArray {OF PACKED ARRAY [1..SIZEOF(TClassName)] OF CHAR};
1  185 --            END;
1  186 --
1  187 --        TEnumAccesses = (fRead, fWrite, fAppend, fPrivate); {not allowing global_refnum at this time}
1  188 --        TAccesses = SET OF TEnumAccesses;
1  189 --        TIOMode = (fAbsolute, fRelative, fSequential);
1  190 --        xReadWrite = (xRead, xWrite);
1  191 --        SizeOfNumber = 1..4;
1  192 --
1  193 --        TScanDirection = (scanForward, scanBackward);
1  194 --
1  195 --        TConvResult = (cvValid, cvNoNumber, cvBadNumber, cvOverflow);
1  196 --
1  197 --
1  198 --      {Classes}
1  199 --
1  200 --        TObject = SUBCLASS OF NIL
1  201 --
1  202 --          {Creation and Destruction}
1  203 --            FUNCTION  TObject.CREATE(object: TObject; heap: THeap): TObject; ABSTRACT;
1  204 --            PROCEDURE TObject.Become(object: TObject);                       {SELF becomes obj and former SELF is freed}
1  205 --            FUNCTION  TObject.Class: TClass;                                 {its class pointer}
1  206 --            FUNCTION  TObject.CloneObject(heap: THeap): TObject;             {clones just the object, not its dependents}
1  207 --            FUNCTION  TObject.Clone(heap: THeap): TObject; DEFAULT;          {clones the object and its known dependents}
1  208 --            PROCEDURE TObject.FreeObject; DEFAULT;                           {frees just the object, not its dependents}
1  209 --            PROCEDURE TObject.Free; DEFAULT;                                 {frees the object and its known dependents}
1  210 --            FUNCTION  TObject.Heap: THeap;                                   {which heap it is in}
1  211 --            FUNCTION  TObject.HeapBytes: INTEGER;                            {number of bytes occupied in that heap}
1  212 --            PROCEDURE TObject.Read(s: TStringScanner);                       {reads the object & its known dependents}
1  213 --            PROCEDURE TObject.Write(s: TStringScanner);                      {writes the object & its known dependents}
1  214 --
1  215 --          {Debugging}
1  216 --            {$IFC fDebugMethods}
1  217 --            PROCEDURE TObject.Fields(PROCEDURE Field(nameAndType: S255)); DEFAULT; {See end of file for comment}
1  218 --            PROCEDURE TObject.Debug(numLevels: INTEGER; memberTypeStr: S255); DEFAULT;
1  219 --              {writes an object down to numLevels:
1  220 --                       numLevels=0 => write only class;
```

```
1  221 --                    numLevels=1 => write class, non-Object fields, and class of Object fields
1  222 --                    etc.)
1  223 --          {$ENDC}
1  224 --
1  225 --        {Version Conversion}
1  226 --          PROCEDURE TObject.Convert(fromVersion: Byte);  {Override it to finish conversion from an old version}
1  227 --          FUNCTION  TObject.JoinClass(newClass: TClass): TObject;        {Called for you by version conversion}
1  228 --
1  229 --          END;
1  230 --
1  231 --
1  232 --      TCollecHeader =  RECORD
1  233 --          cEassPtr:         TClass;
1  234 --          size:             LONGINT;     {number of real elements, not counting the hole}
1  235 --          dynStart:         INTEGER;     {bytes from the class ptr to the dynamic data; MAXINT if none alloued}
1  236 --          holeStart:        INTEGER;     {0 = at the beginning, size = at the end; MAXINT = none alloued}
1  237 --          holeSize:         INTEGER;     {measured in MemberBytes units}
1  238 --          holeStd:          INTEGER;     {if the holeSize goes to 0, how much to grou the collection by}
1  239 --          END;
1  240 --
1  241 --      TFastString = RECORD               {only access ch[i] when hole is at end & TString is not subclassed}
1  242 --          header:           TCollecHeader;
1  243 --          ch:               PACKED ARRAY[1..32740] OF CHAR;
1  244 --          END;
1  245 --      TPFastString = ↑TFastString;
1  246 --      THFastString = ↑TPFastString;
1  247 --
1  248 --      TArrayHeader =  RECORD
1  249 --          classPtr:         TClass;
1  250 --          size:             LONGINT;     {number of real elements, not counting the hole}
1  251 --          dynStart:         INTEGER;     {bytes from the class ptr to the dynamic data}
1  252 --          holeStart:        INTEGER;     {0 means hole at the beginning, size means hole at the end}
1  253 --          holeSize:         INTEGER;     {measured in MemberBytes units}
1  254 --          holeStd:          INTEGER;     {if the holeSize goes to 0, how much to grou the collection by}
1  255 --          recordBytes:      INTEGER;
1  256 --          END;
1  257 --
1  258 --
1  259 --      TCollection = SUBCLASS OF TObject
1  260 --
1  261 --        {Variables}
1  262 --          size:             LONGINT;     {number of real elements, not counting the hole}
1  263 --          dynStart:         INTEGER;     {bytes from the class ptr to the dynamic data}
1  264 --          holeStart:        INTEGER;     {0 means hole at the beginning, size means hole at the end}
1  265 --          holeSize:         INTEGER;     {measured in MemberBytes units}
1  266 --          holeStd:          INTEGER;     {if the holeSize goes to 0, how much to grou the collection by}
1  267 --
1  268 --          {The field "size" is a LONGINT for the benefit of huge collections like remote data bases.
1  269 --           It is always in the INTEGER range for non-subclassed TLists, TArrays, and TStrings.
1  270 --
1  271 --           The field "dynStart" is an offset from Handle(collection)↑ and tells where the dynamic part
1  272 --           of the data is stored, if any.  This convention allows subclasses to add fields.
1  273 --
1  274 --           When editing a collection, there may be an unused "hole" somewhere in the storage block.  The
1  275 --           fields "holeStart" and "holeSize" specify (in member-sized units) the starting index of the
1  276 --           hole and the length of the hole.  When holeSize is zero, there is no hole.  If members are
1  277 --           added when there is no hole, the storage block is expanded to allow for at least another
1  278 --           "holeStd" members.
1  279 --
1  280 --           CREATE has an argument that lets the initial collection have a hole at the end, so that
1  281 --           Ins- methods can be called to initialize the collection without any storage allocation.
1  282 --
1  283 --           StartEdit sets holeStd to its argument, which forces subsequent edit methods to leave intact
1  284 --           any hole they might form.  StopEdit squeezes out the hole and sets holeStd to zero, which
1  285 --           forces subsequent edit methods that get called with no hole to squeeze out any hole they may form.
1  286 --           Thus, every StartEdit that has a nonzero argument should be terminated by a call on StopEdit to
1  287 --           save space.}
1  288 --
1  289 --        {Creation and Destruction}
1  290 --          FUNCTION  TCollection.CREATE(object: TObject; heap: THeap; initialSlack: INTEGER): TCollection;
1  291 --          FUNCTION  TCollection.Clone(heap: THeap): TObject; OVERRIDE;
1  292 --
1  293 --        {Attributes}
1  294 --          FUNCTION  TCollection.MemberBytes: INTEGER; ABSTRACT;
1  295 --          FUNCTION  TCollection.Equals(otherCollection: TCollection): BOOLEAN;
1  296 --
1  297 --        {Slack control}
1  298 --          PROCEDURE TCollection.StartEdit(withSlack: INTEGER);
1  299 --          PROCEDURE TCollection.StopEdit;
1  300 --
1  301 --        {Generic Inserts}
1  302 --          PROCEDURE TCollection.InsManyAt(i: LONGINT; otherCollection: TCollection; index, howMany: LONGINT);
1  303 --          PROCEDURE TCollection.InsNullsAt(i, howMany: LONGINT);
1  304 --
1  305 --      (* BEGIN CONCEPTUAL METHODS (parameter types differ in subclasses; sometimes extra parameters required)
1  306 --
1  307 --        {Enumerate members}
1  308 --          PROCEDURE TCollection.Each(PROCEDURE DoToMember(member: "TMember")); CONCEPTUAL;
1  309 --          FUNCTION  TCollection.Pos(after: LONGINT; member: "TMember"): LONGINT; CONCEPTUAL;
1  310 --          FUNCTION  TCollection.Scanner: TScanner; CONCEPTUAL;                {c.ScannerFrom(-MaxLInt, scanForward)}
1  311 --          FUNCTION  TCollection.ScannerFrom(firstToScan: LONGINT; scanDirection: TScanDirection)
1  312 --                                          : TScanner; CONCEPTUAL;
1  313 --
1  314 --        {Inspect members}
1  315 --          FUNCTION  TCollection.At(i: LONGINT): "TMember"; CONCEPTUAL;
1  316 --          FUNCTION  TCollection.First: "TMember"; CONCEPTUAL;
1  317 --          FUNCTION  TCollection.Last: "TMember"; CONCEPTUAL;
1  318 --          FUNCTION  TCollection.ManyAt(i, howMany: LONGINT): "TCollection"; CONCEPTUAL;
1  319 --
1  320 --        {Insert members}
1  321 --          PROCEDURE TCollection.InsAt(i: LONGINT; member: "TMember"); CONCEPTUAL;
1  322 --          PROCEDURE TCollection.InsFirst(member: "TMember"); CONCEPTUAL;
1  323 --          PROCEDURE TCollection.InsLast(member: "TMember"); CONCEPTUAL;
1  324 --
1  325 --        {Delete members}
1  326 --          PROCEDURE TCollection.DelAll; CONCEPTUAL;
1  327 --          PROCEDURE TCollection.DelAt(i: LONGINT); CONCEPTUAL;
1  328 --          PROCEDURE TCollection.DelFirst; CONCEPTUAL;
1  329 --          PROCEDURE TCollection.DelLast; CONCEPTUAL;
1  330 --          PROCEDURE TCollection.DelManyAt(i, howMany: LONGINT); CONCEPTUAL;
```

```
1  331 --
1  332 --          {Change member}
1  333 --            PROCEDURE TCollection.PutAt(i: LONGINT; member: "TMember"); CONCEPTUAL;
1  334 --
1  335 --        END CONCEPTUAL METHODS *)
1  336 --
1  337 --          {Private methods -- to be called by subclasses only!!!}
1  338 --            {$IFC fRngObject}
1  339 --            PROCEDURE TCollection.CheckIndex(index: LONGINT);
1  340 --            {$ENDC}
1  341 --            FUNCTION  TCollection.AddrMember(i: LONGINT): LONGINT;        {The address is only valid momentarily}
1  342 --            PROCEDURE TCollection.CopyMembers(dstAddr, startIndex, howMany: LONGINT);
1  343 --            PROCEDURE TCollection.EditAt(atIndex: LONGINT; deltaMembers: INTEGER);          {Transfers no data}
1  344 --            PROCEDURE TCollection.ResizeColl(membersPlusHole: INTEGER);      {Resizes at end, no fields changed}
1  345 --            PROCEDURE TCollection.ShiftColl(afterSrcIndex, afterDstIndex, howMany: INTEGER); {No fields changed}
1  346 --
1  347 --            END;
1  348 --
1  349 --        TList = SUBCLASS OF TCollection
1  350 --
1  351 --          {Variables}
1  352 --
1  353 --          {Creation and Destruction}
1  354 --            FUNCTION  TList.CREATE(object: TObject; heap: THeap; initialSlack: INTEGER): TList;
1  355 --            FUNCTION  TList.Clone(heap: THeap): TObject; OVERRIDE;
1  356 --            PROCEDURE TList.Free; OVERRIDE;
1  357 --
1  358 --          {Debugging}
1  359 --            {$IFC fDebugMethods}
1  360 --            PROCEDURE TList.Debug(numLevels: INTEGER; memberTypeStr: S255); OVERRIDE;
1  361 --                { numLevels=0   print just class of list;
1  362 --                          1   also print size of list;
1  363 --                          2   also print compacted list of member classes
1  364 --                        >=3   print class, size, and call Debug(numLevels-1) on members
1  365 --                }
1  366 --            PROCEDURE TList.DebugMembers;
1  367 --            {$ENDC}
1  368 --
1  369 --          {Attributes}
1  370 --            FUNCTION  TList.MemberBytes: INTEGER; OVERRIDE;
1  371 --
1  372 --          {Enumerate members}
1  373 --            PROCEDURE TList.Each(PROCEDURE DoToObject(object: TObject)); DEFAULT;
1  374 --            FUNCTION  TList.Pos(after: LONGINT; object: TObject): LONGINT;
1  375 --            FUNCTION  TList.Scanner: TListScanner;
1  376 --            FUNCTION  TList.ScannerFrom(firstToScan: LONGINT; scanDirection: TScanDirection)
1  377 --                                    : TListScanner; DEFAULT;
1  378 --
1  379 --          {Inspect members}
1  380 --            FUNCTION  TList.At(i: LONGINT): TObject; DEFAULT;
1  381 --            FUNCTION  TList.First: TObject; DEFAULT;
1  382 --            FUNCTION  TList.Last: TObject; DEFAULT;
1  383 --            FUNCTION  TList.ManyAt(i, howMany: LONGINT): TList; DEFAULT;
1  384 --
1  385 --          {Insert members}
1  386 --            PROCEDURE TList.InsAt(i: LONGINT; object: TObject); DEFAULT;
1  387 --            PROCEDURE TList.InsFirst(object: TObject);
1  388 --            PROCEDURE TList.InsLast(object: TObject);
1  389 --
1  390 --          {Delete members}
1  391 --            PROCEDURE TList.DelAll(freeOld: BOOLEAN); DEFAULT;
1  392 --            PROCEDURE TList.DelAt(i: LONGINT; freeOld: BOOLEAN); DEFAULT;
1  393 --            PROCEDURE TList.DelFirst(freeOld: BOOLEAN);
1  394 --            PROCEDURE TList.DelLast(freeOld: BOOLEAN);
1  395 --            PROCEDURE TList.DelManyAt(i, howMany: LONGINT; freeOld: BOOLEAN); DEFAULT;
1  396 --            PROCEDURE TList.DelObject(object: TObject; freeOld: BOOLEAN);
1  397 --            FUNCTION  TList.PopLast: TObject;
1  398 --
1  399 --          {Change member}
1  400 --            PROCEDURE TList.PutAt(i: LONGINT; object: TObject; freeOld: BOOLEAN); DEFAULT;
1  401 --
1  402 --            END;
1  403 --
1  404 --        TArray = SUBCLASS OF TCollection     {*** WARNING: The Ptrs below become invalid if the heap compacts!!!}
1  405 --
1  406 --          {Variables}
1  407 --            recordBytes: INTEGER;
1  408 --
1  409 --          {Creation and Destruction}
1  410 --            FUNCTION  TArray.CREATE(object: TObject; heap: THeap; initialSlack, bytesPerRecord: INTEGER): TArray;
1  411 --
1  412 --          {Attributes}
1  413 --            FUNCTION  TArray.MemberBytes: INTEGER; OVERRIDE;
1  414 --
1  415 --          {Enumerate members}
1  416 --            PROCEDURE TArray.Each(PROCEDURE DoToRecord(pRecord: Ptr)); DEFAULT;
1  417 --            FUNCTION  TArray.Pos(after: LONGINT; pRecord: Ptr): LONGINT;
1  418 --            FUNCTION  TArray.Scanner: TArrayScanner;
1  419 --            FUNCTION  TArray.ScannerFrom(firstToScan: LONGINT; scanDirection: TScanDirection)
1  420 --                                    : TArrayScanner; DEFAULT;
1  421 --
1  422 --          {Inspect members}
1  423 --            FUNCTION  TArray.At(i: LONGINT): Ptr; DEFAULT;
1  424 --            FUNCTION  TArray.First: Ptr;
1  425 --            PROCEDURE TArray.GetAt(i: LONGINT; pRecord: Ptr); DEFAULT;  {Sort of: pRecord↑ := SELF.At(i)↑}
1  426 --            FUNCTION  TArray.Last: Ptr;
1  427 --            FUNCTION  TArray.ManyAt(i, howMany: LONGINT): TArray; DEFAULT;
1  428 --
1  429 --          {Insert members}
1  430 --            PROCEDURE TArray.InsAt(i: LONGINT; pRecord: Ptr); DEFAULT;
1  431 --            PROCEDURE TArray.InsFirst(pRecord: Ptr);
1  432 --            PROCEDURE TArray.InsLast(pRecord: Ptr);
1  433 --
1  434 --          {Delete members}
1  435 --            PROCEDURE TArray.DelAll; DEFAULT;
1  436 --            PROCEDURE TArray.DelAt(i: LONGINT); DEFAULT;
1  437 --            PROCEDURE TArray.DelFirst;
1  438 --            PROCEDURE TArray.DelLast;
1  439 --            PROCEDURE TArray.DelManyAt(i, howMany: LONGINT); DEFAULT;
1  440 --
```

```
1  441 --          {Change member}
1  442 --            PROCEDURE TArray.PutAt(i: LONGINT; pRecord: Ptr); DEFAULT;
1  443 --
1  444 --            END;
1  445 --
1  446 --        TString = SUBCLASS OF TCollection
1  447 --
1  448 --          {Variables}
1  449 --
1  450 --          {Creation and Destruction}
1  451 --            FUNCTION  TString.CREATE(object: TObject; heap: THeap; initialSlack: INTEGER): TString;
1  452 --
1  453 --          {Attributes}
1  454 --            FUNCTION  TString.MemberBytes: INTEGER; OVERRIDE;
1  455 --
1  456 --          {Enumerate members}
1  457 --            PROCEDURE TString.Each(PROCEDURE DoToCharacter(character: CHAR));
1  458 --            FUNCTION  TString.Pos(after: LONGINT; character: CHAR): LONGINT;
1  459 --            FUNCTION  TString.Scanner: TStringScanner;
1  460 --            FUNCTION  TString.ScannerFrom(firstToScan: LONGINT; scanDirection: TScanDirection): TStringScanner;
1  461 --
1  462 --          {Inspect members}
1  463 --            FUNCTION  TString.At(i: LONGINT): CHAR;
1  464 --            FUNCTION  TString.First: CHAR;
1  465 --            FUNCTION  TString.Last: CHAR;
1  466 --            FUNCTION  TString.ManyAt(i, howMany: LONGINT): TString;
1  467 --            PROCEDURE TString.ToPStr(pStr: TPString);
1  468 --            PROCEDURE TString.ToPStrAt(i, howMany: LONGINT; pStr: TPString);
1  469 --
1  470 --          {Insert members}
1  471 --            PROCEDURE TString.InsAt(i: LONGINT; character: CHAR);
1  472 --            PROCEDURE TString.InsFirst(character: CHAR);
1  473 --            PROCEDURE TString.InsLast(character: CHAR);
1  474 --            PROCEDURE TString.InsPStrAt(i: LONGINT; pStr: TPString);
1  475 --
1  476 --          {Delete members}
1  477 --            PROCEDURE TString.DelAll;
1  478 --            PROCEDURE TString.DelAt(i: LONGINT);
1  479 --            PROCEDURE TString.DelFirst;
1  480 --            PROCEDURE TString.DelLast;
1  481 --            PROCEDURE TString.DelManyAt(i, howMany: LONGINT);
1  482 --
1  483 --          {Change member}
1  484 --            PROCEDURE TString.PutAt(i: LONGINT; character: CHAR);
1  485 --
1  486 --          {QuickDraw}
1  487 --            PROCEDURE TString.Draw(i: LONGINT; howMany: INTEGER);
1  488 --            FUNCTION  TString.Width(i: LONGINT; howMany: INTEGER): INTEGER;
1  489 --
1  490 --            END;
1  491 --
1  492 --        TFile = SUBCLASS OF TCollection
1  493 --
1  494 --          {Variables}
1  495 --            path:       TFilePath;
1  496 --            password:   TPassword;              {The current password protecting this file, and used for all
1  497 --                                                    accesses to it;  client is responsible for setting this
1  498 --                                                    field after the TFile is created; ignored if
1  499 --                                                    LibraryVersion <= 20}
1  500 --            scanners:   TList {OF TScanner};
1  501 --
1  502 --          {Creation and Destruction}
1  503 --            FUNCTION  TFile.CREATE(object: TObject; heap: THeap; itsPath: TFilePath;
1  504 --                                   itsPassword: TPassword): TFile;
1  505 --                      {itsPassword is ignored from LibraryVersion <= 20}
1  506 --
1  507 --            PROCEDURE TFile.Free; OVERRIDE;                         {Frees the scanners as well}
1  508 --            FUNCTION  TFile.Clone(heap: THeap): TObject; OVERRIDE;  {Illegal}
1  509 --
1  510 --          {Attributes}
1  511 --            FUNCTION  TFile.MemberBytes: INTEGER; OVERRIDE;
1  512 --
1  513 --          {Enumerate members}
1  514 --            FUNCTION  TFile.Scanner: TFileScanner;                  {f.ScannerFrom(0, [fRead, fWrite])}
1  515 --            FUNCTION  TFile.ScannerFrom(firstToScan: LONGINT; manip: TAccesses): TFileScanner;
1  516 --
1  517 --          {Catalog}
1  518 --            PROCEDURE TFile.ChangePassword(VAR error: INTEGER; newPassword: TPassword);
1  519 --                      {also changes the password field, if successful}
1  520 --            PROCEDURE TFile.Delete(VAR error: INTEGER);
1  521 --            FUNCTION  TFile.Exists(VAR error: INTEGER): BOOLEAN;
1  522 --            FUNCTION  TFile.WhenModified(VAR error: INTEGER): LONGINT;
1  523 --            PROCEDURE TFile.Rename(VAR error: INTEGER; newFileName: TFilePath);
1  524 --            FUNCTION  TFile.VerifyPassword(VAR error: INTEGER; password: TPassword): BOOLEAN;
1  525 --
1  526 --            END;
1  527 --
1  528 --        TScanner = SUBCLASS OF TObject
1  529 --
1  530 --          {Variables}
1  531 --            collection:     TCollection;  {The collection being scanned}
1  532 --            position:       LONGINT;      {The current position (between members: 0=before first, size+1=after
1  533 --                                           last)}
1  534 --            increment:      INTEGER;      {1 if scanning forward, -1 if scanning backward}
1  535 --            scanDone:       BOOLEAN;      {TRUE if next.Scan call should return FALSE, leaving its VAR
1  536 --                                           parameter alone}
1  537 --            atEnd:          BOOLEAN;      {TRUE if next.Scan call will return FALSE because at end of collection}
1  538 --
1  539 --            FUNCTION  TScanner.CREATE(object: TObject; itsCollection: TCollection; itsInitialPosition: LONGINT;
1  540 --                                      scanDirection: TScanDirection): TScanner;
1  541 --
1  542 --          {Close and Reopen}
1  543 --            PROCEDURE TScanner.Close; DEFAULT;  {If disk-based, flush buffers and tell OS to close file,
1  544 --                                                 else no-op}
1  545 --            PROCEDURE TScanner.Open; DEFAULT;   {If disk-based, tell OS to reopen file and fill first buffer}
1  546 --
1  547 --          {Slack Control}
1  548 --            PROCEDURE TScanner.Allocate(slack: LONGINT); DEFAULT;   {Like collection.StartEdit(slack)}
1  549 --            PROCEDURE TScanner.Compact; DEFAULT;                    {Like collection.StopEdit}
1  550 --
```

```
1  551 --          {Positioning}
1  552 --            FUNCTION  TScanner.Advance(PROCEDURE DoToCurrent(anotherMember: BOOLEAN)): BOOLEAN;
1  553 --            PROCEDURE TScanner.Done; DEFAULT;                          {Set scanDone so that Scan will return FALSE}
1  554 --            PROCEDURE TScanner.Reverse; DEFAULT;                       {Reverse the scan direction}
1  555 --            PROCEDURE TScanner.Seek(newPosition: LONGINT); DEFAULT;    {Forces to legal places}
1  556 --            PROCEDURE TScanner.Skip(deltaPos: LONGINT); DEFAULT;       {Forces to legal places}
1  557 --
1  558 --        (* BEGIN CONCEPTUAL METHODS (parameter types differ in subclasses; sometimes extra parameters required)
1  559 --
1  560 --          {Data Transfer}
1  561 --            FUNCTION  TScanner.Obtain: "TMember"; CONCEPTUAL; {Return previous member (redundant right after
1  562 --                                                               Scan)}
1  563 --            FUNCTION  TScanner.Scan(VAR member: "TMember"): BOOLEAN; CONCEPTUAL; {Return next & advance past it}
1  564 --
1  565 --          {Editing}
1  566 --            PROCEDURE TScanner.Append(member: "TMember"); CONCEPTUAL;   {Add a new member after position, scan
1  567 --                                                                         past it}
1  568 --            PROCEDURE TScanner.Delete; CONCEPTUAL;                      {Delete previous member and adjust
1  569 --                                                                         position}
1  570 --            PROCEDURE TScanner.DeleteRest; CONCEPTUAL;                  {Delete everything after SELF.position}
1  571 --            PROCEDURE TScanner.Replace(member: "TMember"); CONCEPTUAL;  {Replace previous member and maintain
1  572 --                                                                         position}
1  573 --
1  574 --        END CONCEPTUAL METHODS *)
1  575 --
1  576 --          END;
1  577 --
1  578 --        TListScanner = SUBCLASS OF TScanner
1  579 --
1  580 --          {Variables}
1  581 --
1  582 --          {Creation and Destruction}
1  583 --            FUNCTION  TListScanner.CREATE(object: TObject; itsList: TList; itsInitialPosition: LONGINT;
1  584 --                                          itsScanDirection: TScanDirection): TListScanner;
1  585 --            PROCEDURE TListScanner.Free; OVERRIDE;
1  586 --
1  587 --          {Traversal}
1  588 --            FUNCTION  TListScanner.Obtain: TObject; DEFAULT; {Return previous member (redundant right after Scan)}
1  589 --            FUNCTION  TListScanner.Scan(VAR nextObject: TObject): BOOLEAN; DEFAULT;{Return next, advance past it}
1  590 --
1  591 --          {Editing}
1  592 --            PROCEDURE TListScanner.Append(object: TObject); DEFAULT;   {Add object after position, scan past it}
1  593 --            PROCEDURE TListScanner.Delete(freeOld: BOOLEAN); DEFAULT;  {Delete previous object, adjust position}
1  594 --            PROCEDURE TListScanner.DeleteRest(freeOld: BOOLEAN); DEFAULT; {Delete all objects after position}
1  595 --            PROCEDURE TListScanner.Replace(object: TObject; freeOld: BOOLEAN); DEFAULT;     {Replace previous}
1  596 --
1  597 --          END;
1  598 --
1  599 --        TArrayScanner = SUBCLASS OF TScanner
1  600 --
1  601 --          {Variables}
1  602 --
1  603 --          {Creation and Destruction}
1  604 --            FUNCTION  TArrayScanner.CREATE(object: TObject; itsArray: TArray; itsInitialPosition: LONGINT;
1  605 --                                           itsScanDirection: TScanDirection): TArrayScanner;
1  606 --            PROCEDURE TArrayScanner.Free; OVERRIDE;
1  607 --
1  608 --          {Traversal}
1  609 --            FUNCTION  TArrayScanner.Obtain: Ptr; DEFAULT; {Return previous member (redundant right after Scan)}
1  610 --            FUNCTION  TArrayScanner.Scan(VAR pNextRecord: Ptr): BOOLEAN; DEFAULT; {Return next & advance past it}
1  611 --
1  612 --          {Editing}
1  613 --            PROCEDURE TArrayScanner.Append(pRecord: Ptr); DEFAULT; {Add a new record after position, scan past it}
1  614 --            PROCEDURE TArrayScanner.Delete; DEFAULT;               {Delete previous record and adjust position}
1  615 --            PROCEDURE TArrayScanner.DeleteRest; DEFAULT;           {Delete all records after position}
1  616 --            PROCEDURE TArrayScanner.Replace(pRecord: Ptr); DEFAULT;{Replace previous record and maintain position}
1  617 --
1  618 --          END;
1  619 --
1  620 --        TStringScanner = SUBCLASS OF TScanner
1  621 --
1  622 --          {Variables}
1  623 --            actual: LONGINT;                                   {no. bytes last xfered}
1  624 --
1  625 --          {Creation and Destruction}
1  626 --            FUNCTION  TStringScanner.CREATE(object: TObject; itsString: TString; itsInitialPosition: LONGINT;
1  627 --                                            itsScanDirection: TScanDirection): TStringScanner;
1  628 --            PROCEDURE TStringScanner.Free; OVERRIDE;
1  629 --
1  630 --          {Traversal}
1  631 --            FUNCTION  TStringScanner.Obtain: CHAR; DEFAULT; {Return previous member (redundant right after Scan)}
1  632 --            FUNCTION  TStringScanner.Scan(VAR nextChar: CHAR): BOOLEAN; DEFAULT;   {Return next & advance past it}
1  633 --
1  634 --          {Editing}
1  635 --            PROCEDURE TStringScanner.Append(character: CHAR); DEFAULT; {Add char after position, scan past it}
1  636 --            PROCEDURE TStringScanner.Delete; DEFAULT;                  {Delete previous char, adjust position}
1  637 --            PROCEDURE TStringScanner.DeleteRest; DEFAULT;              {Delete all chars after position}
1  638 --            PROCEDURE TStringScanner.Replace(character: CHAR); DEFAULT;{Replace previous char, maintain position}
1  639 --
1  640 --          {Typed Sequential Data Transfer: characters are read/written from left to right regardless of increment}
1  641 --            FUNCTION  TStringScanner.ReadArray(heap: THeap; bytesPerRecord: INTEGER): TArray; {reads size first}
1  642 --            FUNCTION  TStringScanner.ReadNumber(numBytes: SizeOfNumber): LONGINT;  {iff numBytes is even
1  643 --                                                                                    then signed}
1  644 --            FUNCTION  TStringScanner.ReadObject(heap: THeap): TObject;    {tells object to Read(SELF)}
1  645 --            PROCEDURE TStringScanner.WriteArray(a: TArray);  {inverse of ReadArray: writes size but not
1  646 --                                                              recordBytes}
1  647 --            PROCEDURE TStringScanner.WriteNumber(value: LONGINT; numBytes: SizeOfNumber);{does not write size}
1  648 --            PROCEDURE TStringScanner.WriteObject(object: TObject);  {tells object to Write(SELF)}
1  649 --            PROCEDURE TStringScanner.XferContiguous(whichWay: xReadWrite; collection: TCollection);
1  650 --                                {xfers the size and members, non-recursively; xRead appends what it reads}
1  651 --            PROCEDURE TStringScanner.XferFields(whichWay: xReadWrite; object: TObject); {xfers all but the clas
1  652 --            PROCEDURE TStringScanner.XferPString(whichWay: xReadWrite; pStr: TPString); {it better be long enou
1  653 --
1  654 --          {Untyped Data Transfer: characters are read/written from left to right regardless of increment}
1  655 --            PROCEDURE TStringScanner.XferSequential(whichWay: xReadWrite; pFirst: Ptr; numBytes:
1  656 --                                                    LONGINT); DEFAULT;
1  657 --            PROCEDURE TStringScanner.XferRandom(whichWay: xReadWrite; pFirst: Ptr; numBytes: LONGINT;
1  658 --                                                mode: TIOMode; offset: LONGINT); DEFAULT;
1  659 --
1  660 --          END;
```

```
1  661 --
1  662 --      TFileScanner = SUBCLASS OF TStringScanner
1  663 --
1  664 --        {Variables}
1  665 --          accesses:    TAccesses;                                  {[fRead, fWrite, fAppend, fPrivate]}
1  666 --          refnum:      INTEGER;                                    {OS file refnum, or -1 if not open now}
1  667 --          error:       INTEGER;     {EOF is not an error}          {first error (or warning if no error) encountered}
1  668 --
1  669 --        {Creation and Destruction}
1  670 --          FUNCTION  TFileScanner.CREATE(object: TObject; itsFile: TFile; manip: TAccesses): TFileScanner;
1  671 --          PROCEDURE TFileScanner.FreeObject; OVERRIDE;             {also closes the OS file}
1  672 --          PROCEDURE TFileScanner.Free; OVERRIDE;                   {if the last scanner, frees the TFile, too}
1  673 --
1  674 --        {Close and Reopen}
1  675 --          PROCEDURE TFileScanner.Close; OVERRIDE;
1  676 --          PROCEDURE TFileScanner.Open; OVERRIDE;
1  677 --
1  678 --        {Slack Control}
1  679 --          PROCEDURE TFileScanner.Allocate(slack: LONGINT); OVERRIDE; {Get slack DIV pageSize unused disk pages}
1  680 --          PROCEDURE TFileScanner.Compact; OVERRIDE;                  {Return unused disk pages to free space}
1  681 --
1  682 --        {Positioning}
1  683 --          PROCEDURE TFileScanner.Seek(newPosition: LONGINT); OVERRIDE;
1  684 --          PROCEDURE TFileScanner.Skip(deltaPos: LONGINT); OVERRIDE;
1  685 --
1  686 --        {Traversal}
1  687 --          FUNCTION  TFileScanner.Obtain: CHAR; OVERRIDE; {Return previous member (redundant right after Scan)}
1  688 --          FUNCTION  TFileScanner.Scan(VAR nextChar: CHAR): BOOLEAN; OVERRIDE;   {Return next & advance past it}
1  689 --
1  690 --        {Editing}
1  691 --          PROCEDURE TFileScanner.Append(character: CHAR); OVERRIDE; {Acts like: Replace; Skip(1)}
1  692 --          PROCEDURE TFileScanner.Delete; OVERRIDE;                  {Acts like: Skip(-1)}
1  693 --          PROCEDURE TFileScanner.DeleteRest; OVERRIDE;              {Shorten file size to SELF.position}
1  694 --          PROCEDURE TFileScanner.Replace(character: CHAR); OVERRIDE;{Replace previous member and maintain
1  695 --                                                                     position}
1  696 --
1  697 --        {Untyped Data Transfer: characters are read/written from left to right regardless of increment}
1  698 --          PROCEDURE TFileScanner.XferSequential(whichWay: xReadWrite; pFirst: Ptr; numBytes: LONGINT); OVERRIDE;
1  699 --          PROCEDURE TFileScanner.XferRandom(whichWay: xReadWrite; pFirst: Ptr; numBytes: LONGINT;
1  700 --                                            mode: TIOMode; offset: LONGINT); OVERRIDE;
1  701 --
1  702 --         END;
1  703 --
1  704 --    {$IFC compatibleLists}
1  705 --
1  706 --        {Backward compatibility classes}
1  707 --
1  708 --        TDynamicArray = SUBCLASS OF TArray
1  709 --            ch: {UNPACKED} ARRAY [0..16370] OF CHAR;
1  710 --            FUNCTION  TDynamicArray.CREATE(object: TObject; heap: THeap; bytesPerRecord: INTEGER;
1  711 --                                           initialSize: INTEGER): TDynamicArray;
1  712 --            FUNCTION  TDynamicArray.NumRecords: INTEGER;
1  713 --            PROCEDURE TDynamicArray.BeSize(newSize: INTEGER);
1  714 --            END;
1  715 --
1  716 --        TIndexList = SUBCLASS OF TList
1  717 --            elements: ARRAY[1..1] OF TObject;
1  718 --            FUNCTION  TIndexList.CREATE(object: TObject; heap: THeap; initialSize: INTEGER): TIndexList;
1  719 --            FUNCTION  TIndexList.numElements: INTEGER;
1  720 --            END;
1  721 --
1  722 --        TLinkList = SUBCLASS OF TList
1  723 --            FUNCTION  TLinkList.CREATE(object: TObject; heap: THeap): TLinkList;
1  724 --            FUNCTION  TLinkList.numElements: INTEGER;
1  725 --            END;
1  726 --
1  727 --        TBlockList = SUBCLASS OF TList
1  728 --            FUNCTION  TBlockList.CREATE(object: TObject; heap: THeap; itsMinBlockSize: INTEGER): TBlockList;
1  729 --            FUNCTION  TBlockList.numElements: INTEGER;
1  730 --            END;
1  731 --
1  732 --        TFileStream = SUBCLASS OF TFileScanner
1  733 --            FUNCTION  TFileStream.CREATE(object: TObject; heap: THeap; path: S255; manip: TAccesses): TFileStream;
1  734 --            FUNCTION  TFileStream.Size: LONGINT;
1  735 --            END;
1  736 --
1  737 --    {$ENDC}
1  738 --
1  739 --
1  740 --    VAR
1  741 --
1  742 --        mainDsRefnum:           INTEGER;                    {refnum of the process data segment}
1  743 --        mainHeap:               THeap;                      {heap of the process}
1  744 --        mainLdsn:               INTEGER;                    {ldsn of the process data segment}
1  745 --        fCheckIndices:          BOOLEAN;
1  746 --
1  747 --        onDesktop:              BOOLEAN;                    {Is there a DM (Desktop Manager) to talk to?}
1  748 --        wmIsInitialized:        BOOLEAN;                    {Has OpenWM been done?}
1  749 --        isInitialized:          BOOLEAN;                    {Iff TRUE, shouldn't tell DM initFailed any more}
1  750 --        amDying:                BOOLEAN;                    {Iff TRUE, I have called ImDying}
1  751 --
1  752 --        myWorld:                TClassWorld;                {For Version Conversion}
1  753 --
1  754 --
1  755 --        { Variables for Debugging }
1  756 --            indentTrace:    INTEGER;
1  757 --
1  758 --              { stuff for the intelligent output }
1  759 --            currXPos:       INTEGER;
1  760 --            outputIndent:   INTEGER;
1  761 --
1  762 --            {$IFC fTrace}
1  763 --                { TRUE if we want to inhibit tracing; client must save and restore its value;
1  764 --                  normally this is needed only if you override the Debug method }
1  765 --            fDebugRecursion:    BOOLEAN;
1  766 --                { how often to call KeyPress from debugger to check for user interrupt }
1  767 --            keyPresLimit:       INTEGER;
1  768 --            {$ENDC}
1  769 --
1  770 --
```

```
1  771 --   {$IFC fCheckHeap}
1  772 --   FUNCTION CountHeap(heap: THeap): INTEGER;
1  773 --   {$ENDC}
1  774 --
1  775 --   FUNCTION  Min(i, j: LONGINT): LONGINT;
1  776 --   FUNCTION  Max(i, j: LONGINT): LONGINT;
1  777 --
1  778 --   PROCEDURE XferLeft(source, dest: Ptr; nBytes: INTEGER);
1  779 --   PROCEDURE XferRight(source, dest: Ptr; nBytes: INTEGER);
1  780 --   FUNCTION  EqualBytes(source, dest: Ptr; nBytes: INTEGER): BOOLEAN;
1  781 --
1  782 --   FUNCTION  LIntAndLInt(i, j: LONGINT): LONGINT;
1  783 --   FUNCTION  LIntOrLInt(i, j: LONGINT): LONGINT;
1  784 --   FUNCTION  LIntXorLInt(i, j: LONGINT): LONGINT;
1  785 --
1  786 --   FUNCTION  NewObject(heap: THeap; itsClass: TClass): TObject;
1  787 --   FUNCTION  NewDynObject(heap: THeap; itsClass: TClass; dynBytes: INTEGER): TObject;
1  788 --   PROCEDURE ResizeDynObject(object: TObject; newTotalBytes: INTEGER);
1  789 --   FUNCTION  NewOrRecycledObject(heap: THeap; itsClass: TClass; VAR chainHead: TObject): TObject;
1  790 --   PROCEDURE RecycleObject(object: TObject; VAR chainHead: TObject);
1  791 --   PROCEDURE Free(object: TObject);
1  792 --
1  793 --   {$IFC compatibleLists}
1  794 --      {Backward compatibility procedures}
1  795 --
1  796 --   FUNCTION  SubObject(super: TObject; itsClass: TClass): TObject;
1  797 --   PROCEDURE FileDelete(path: S255);
1  798 --   PROCEDURE FileLookup(VAR error: INTEGER; path: S255);
1  799 --   PROCEDURE FileRename(oldPath, newPath: S255);
1  800 --   FUNCTION  FileModified(path: S255): LONGINT;
1  801 --   {$ENDC}
1  802 --
1  803 --   FUNCTION  Superclass(class: TClass): TClass;
1  804 --   FUNCTION  ClassDescendsFrom(descendant, ancestor: TClass): BOOLEAN;
1  805 --
1  806 --   PROCEDURE NameOfClass(class: TClass; VAR className: TClassName);
1  807 --   FUNCTION  SizeOfClass(class: TClass): INTEGER;
1  808 --
1  809 --   {The next 3 can only be called from a class-init block or a subroutine of a class-init block}
1  810 --   PROCEDURE UnitAuthor(companyAndAuthor: TAuthorName);                    {required once per unit}
1  811 --   PROCEDURE ClassAuthor(companyAndAuthor: TAuthorName; classAlias: TClassName);   {optional}
1  812 --   PROCEDURE ClassVersion(itsVersion, oldestItCanRead: Byte);              {optional}
1  813 --
1  814 --   FUNCTION  ValidObject(hndl: Handle): BOOLEAN;
1  815 --
1  816 --   PROCEDURE ABCBreak(s: S255; errCode: LONGINT);
1  817 --
1  818 --   PROCEDURE ClascalError(error: INTEGER);
1  819 --
1  820 --   {Some useful procedures; we should decide once and for all whether or not to keep any or all of these}
1  821 --   PROCEDURE LIntToHex(decNumber: LONGINT; hexNumber: TPString);
1  822 --               {NOTE: hexNumber must be >= 8 characters, regardless of size of decNumber}
1  823 --   PROCEDURE LIntToStr(decNumber: LONGINT; str: TPString);
1  824 --               {NOTE: str must be >= 11 characters (sign + 10 digits), regardless of size of decNumber}
1  825 --   PROCEDURE IntToStr(decNumber: INTEGER; str: TPString);
1  826 --               {NOTE: str must be >= 6 characters (sign + 5 digits), regardless of size of decNumber}
1  827 --   PROCEDURE HexStrToLInt(hexString: TPString; VAR decNumber: LONGINT; VAR result: TConvResult);
1  828 --   PROCEDURE StrToLInt(str: TPString; VAR decNumber: LONGINT; VAR result: TConvResult);
1  829 --   PROCEDURE StrToInt(str: TPString; VAR decNumber: INTEGER; VAR result: TConvResult);
1  830 --
1  831 --   PROCEDURE TrimBlanks(str: TPString);
1  832 --   FUNCTION  CharUpperCased(ch: CHAR): CHAR;
1  833 --   PROCEDURE StrUpperCased(str: TPString);
1  834 --
1  835 --   PROCEDURE SplitFilePath(VAR fullPath, itsCatalog, itsFilePart: TFilePath);
1  836 --      {fullPath = CONCAT(itsCatalog, itsFilePart)}
1  837 --
1  838 --   PROCEDURE LatestError(newError: INTEGER; VAR previousError: INTEGER);
1  839 --         {This is used to handle error codes returned by multiple operations, so that you end up with
1  840 --            the first error number or warning number (error code < 0) if there was no error.
1  841 --          You should pass in the latest error as 'newError' and the variable that is to be the final
1  842 --            error code as 'previousError'.  Here is the actual code of LatestError:
1  843 --
1  844 --                              IF ((newError > 0) AND (previousError <= 0) OR
1  845 --                                 (newError < 0) AND (previousError = 0)) THEN
1  846 --                                previousError := newError
1  847 --         }
1  848 --
1  849 --   {$IFC fDbgObject}
1  850 --   PROCEDURE EntDebugger(inputStr, enterReason: S255);
1  851 --   PROCEDURE DumpVar(pVariable: Ptr; nameAndType: S255); {used mainly by TProcess.DumpGlobals}
1  852 --   PROCEDURE WrStr(str: S255);                  { write a string with wrap-around }
1  853 --   PROCEDURE WrLn;                              { goto next line, and output indent }
1  854 --   {$IFC fDebugMethods}
1  855 --   PROCEDURE WrObj(object: TObject; numLevels: INTEGER; memberTypeStr: S255);
1  856 --   {$ENDC}
1  857 --   {$ENDC}
1  858 --
1  859 --   {$IFC fDbgObject OR fDebugMethods}
1  860 --   FUNCTION CheckKeyPress(routine: S255): BOOLEAN;
1  861 --   {$ENDC}
1  862 --
1  863 --   FUNCTION  NewHeap(VAR error: INTEGER; heapStart, numBytes: LONGINT; numObjects: INTEGER): THeap;
1  864 --   FUNCTION  MakeDataSegment(VAR error, dsRefnum: INTEGER; firstTryVolume, thenTryVolume: TFilePath;
1  865 --                             ldsn, memBytes, diskBytes: INTEGER): LONGINT;
1  866 --
1  867 --   PROCEDURE SetHeap(heap: THeap);
1  868 --   PROCEDURE GetHeap(VAR heap: THeap);
1  869 --      {We can't USE Unit Storage because of type name conflicts (Ptr, Handle, ProcPtr)}
1  870 --
1  871 --
1  872 --   FUNCTION  NeedConversion(exClassWorld: TClassWorld; VAR olderVersion, newerVersion: BOOLEAN): BOOLEAN;
1  873 --   PROCEDURE ConvertHeap(heap: THeap; exClassWorld: TClassWorld);
1  874 --
1  875 --   PROCEDURE MarkHeap(heap: THeap; mpAddress: LONGINT);
1  876 --   PROCEDURE SweepHeap(heap: THeap; report: BOOLEAN);
1  877 --
1  878 --   {$IFC fTrace}
1  879 --   PROCEDURE BP(MyTraceLevel: integer);
1  880 --                  {Trace entry to method and write SELF (unless CREATE, Debug, or FreeObject)}
```

```
1  881 --   PROCEDURE EP;  {Trace entry from method and write SELF (unless CREATE, Debug, FreeObject, or Free)}
1  882 --   {$ENDC}
1  883 --
1  884 --
1  885 --           (* ========================  RULES FOR WRITING A Fields FUNCTION  ====================
1  886 --
1  887 --            This function must be defined in every class until the compiler generates this info automatically!
1  888 --
1  889 --               PROCEDURE TWhatever.Fields{{PROCEDURE Field(nameAndType: S255)}};
1  890 --               BEGIN  {THE FIELDS MUST BE LISTED IN DECLARED ORDER, NONE OMITTED AND NONE ADDED}
1  891 --                 {Tell the superclass first (unnecessary if it is TObject)}
1  892 --                   SUPERSELF.Fields(Field);
1  893 --                 {The following type names are recognized by the parser}
1  894 --                   Field('flag: BOOLEAN');
1  895 --                   Field('coCode: Byte');
1  896 --                   Field('inputChar: CHAR');
1  897 --                   Field('version: INTEGER');
1  898 --                   Field('width: LONGINT');
1  899 --                   Field('viewLPt: LPoint');
1  900 --                   Field('boundLRect: LRect');
1  901 --                   Field('size: Point');
1  902 --                   Field('ptr: Ptr');
1  903 --                   Field('boundRect: Rect');
1  904 --                   Field('someName: STRING[100]');
1  905 --                 {If the last field is a Byte or a BOOLEAN, force padding to a word boundary by...
1  906 --                   Field('');
1  907 --                 {Every Registered Class name is recognized}
1  908 --                   Field('miscObj: TObject');
1  909 --                   Field('myPanel: TPanel');
1  910 --                   Field('mySel: TMySelection');
1  911 --                   Field('appSpecific: TAppSpecific');
1  912 --                 {You may report more than one field in a single call to reduce code space}
1  913 --                   Field('boundLRect: LRect; size: Point; ptr: Ptr; mySel: TMySelection');
1  914 --                 {Unpacked invariant RECORDs are recognized}
1  915 --                   Field('info: RECORD version: INTEGER; size: Point END');
1  916 --                 {If the record has variants, select among them before calling Field()}
1  917 --                   CASE SELF.variant OF
1  918 --                        flavor1: Field('RECORD version: INTEGER; size: Point END');
1  919 --                        flavor2: Field('RECORD viewLPt: LPoint END');
1  920 --                        END;
1  921 --                 {Unpacked ARRAYs with literal bounds are recognized}
1  922 --                   Field('desc: ARRAY [1..99] OF RECORD version: INTEGER; id: ARRAY [1..2] OF CHAR END');
1  923 --                 {Other constructs and type names are NOT recognized; substitute one of the above forms}
1  924 --                 {As a last resort, use ARRAY [1..SIZEOF(SELF.fieldName)] OF Byte}
1  925 --               END;
1  926 --           *)
1  927 --
1  928 --   IMPLEMENTATION
1  929 --
2    1 --
2    2 --
1  930 --   {$I LIBTK/UOBJECT2.text}  {Objects, Classes, Streams, and Resources}
3    1 --
3    2 --
1  931 --   {$I LIBTK/UOBJECT3.text}  {Arrays and Lists}
4    1 --
4    2 --
1  932 --   {$I LIBTK/UOBJECT4.text}  {Debugger}
1  933 --
1  934 --   {----------
1  935 --   {$I UOBJECT2.text}  {Objects, Classes, Streams, and Resources}
1  936 --   {$I UOBJECT3.text}  {Arrays and Lists}
1  937 --   {$I UOBJECT4.text}  {Debugger}
1  938 --   ----------}
1  939 --
1  940 --   END.
1  941 --
1  942 --
```

1. libtk/uobject.TEXT
2. LIBTK/UOBJECT2.text
3. LIBTK/UOBJECT3.text
4. LIBTK/UOBJECT4.text

-A-
ABCBreak 816*(1)
accesses 665 (1)
actual 583 (1)
AddrMember 341*(1)
Advance 562*(1)
aliases 184*(1)
Allocate 548*(1) 679*(1)
amDying 750*(1)
Append 592*(1) 613*(1) 635*(1) 691*(1)
At 380*(1) 423*(1) 463*(1)
atEnd 537*(1)
authors 183*(1)

-B-
Become 204*(1)
BeSize 713*(1)
BP 879*(1)
Byte 171*(1)

-C-
ch 243*(1) 709 (1)
ChangePassword 518*(1)
CHAR 243 (1) 463 (1) 464 (1) 465 (1) 631 (1) 687 (1) 709 (1) 832 (1)
CharUpperCased 832*(1)
CheckIndex 339*(1)
CheckKeyPress 860*(1)
ClascalError 818*(1)
Class 205*(1)
ClassAuthor 811*(1)
ClassDescendsFro 804*(1)
classes 182*(1)
classPtr 233*(1) 249*(1)
ClassVersion 812*(1)
Clone 207*(1) 291*(1) 355*(1) 508*(1)
CloneObject 206*(1)
Close 543*(1) 675*(1)
collection 531 (1)
Compact 549*(1) 680*(1)
Convert 226*(1)
ConvertHeap 873*(1)
CopyMembers 342*(1)
CountHeap 772*(1)
CREATE 203 (1) 290*(1) 354 (1) 410*(1) 451 (1) 503*(1) 539*(1) 583 (1) 604 (1) 626*(1)
670*(1) 710*(1) 718*(1) 723 (1) 728 (1) 733 (1)
currXPos 759*(1)
cvBadNumber 195*(1)
cvNoNumber 195*(1)
cvOverflow 195*(1)
cvValid 195*(1)

-D-
Debug 218*(1) 360*(1)
DebugMembers 366*(1)
DelAll 391*(1) 435*(1) 477*(1)
DelAt 392*(1) 436*(1) 478*(1)
Delete 520*(1) 593*(1) 614*(1) 636*(1) 692*(1)
DeleteRest 594*(1) 615*(1) 637*(1) 693*(1)
DelFirst 393*(1) 437*(1) 479*(1)
DelLast 394*(1) 438*(1) 480*(1)
DelManyAt 395*(1) 439*(1) 481*(1)
DelObject 396*(1)
Done 553*(1)
Draw 487*(1)
DumpVar 851*(1)
dynStart 235*(1) 251*(1) 263*(1)

-E-
Each 373*(1) 416*(1) 457*(1)
EditAt 343*(1)
elements 717 (1)
EntDebugger 850*(1)
EP 881*(1)
EqualBytes 780*(1)
Equals 295*(1)
erInternal 146*(1)
error 667*(1)
Exists 521*(1)

-F-
fAbsolute 189*(1)
fAppend 187*(1)
fCheckIndices 745*(1)
fDebugRecursion 765*(1)
Fields 217*(1)
FileDelete 797*(1)
FileLookup 798*(1)
FileModified 800*(1)
FileRename 799*(1)
First 381*(1) 424*(1) 464*(1)
fPrivate 187*(1)
fRead 187*(1)
Free 209*(1) 356*(1) 507*(1) 585*(1) 606*(1) 628*(1) 672*(1) 791*(1)
FreeObject 208*(1) 671*(1)
fRelative 189*(1)
fSequential 189*(1)
fWrite 187*(1)

-G-
GetAt 425*(1)
GetHeap 868*(1)

-H-
Handle 156*(1)
header 242*(1)

Heap 210•(1)
HeapBytes 211•(1)
HexStrToLInt 827•(1)
holeSize 237•(1) 253•(1) 265•(1)
holeStart 236•(1) 252•(1) 264•(1)
holeStd 238•(1) 254•(1) 266•(1)
HWInt 120•(1)

-I-
increment 534•(1)
indentTrace 756•(1)
infRecs 181•(1)
InsAt 386•(1) 430•(1) 471•(1)
InsFirst 387•(1) 431•(1) 472•(1)
InsLast 388•(1) 432•(1) 473•(1)
InsManyAt 302•(1)
InsNullsAt 303•(1)
InsPStrAt 474•(1)
INTRINSIC 8•(1)
IntToStr 825•(1)
isInitialized 749•(1)

-J-
JoinClass 227•(1)

-K-
keyPresLimit 767•(1)

-L-
Last 382•(1) 426•(1) 465•(1)
LatestError 838•(1)
LIntAndLInt 782•(1)
LIntOrLInt 783•(1)
LIntToHex 821•(1)
LIntToStr 823•(1)
LIntXorLInt 784•(1)

-M-
mainDsRefnum 742•(1)
mainHeap 743•(1)
mainLdsn 744•(1)
MakeDataSegment 864•(1)
ManyAt 383•(1) 427•(1) 466•(1)
MarkHeap 875•(1)
Max 776•(1)
MaxBreaks 143•(1)
MAXLINT 148•(1)
MemberBytes 294•(1) 370•(1) 413•(1) 454•(1) 511•(1)
Min 775•(1)
myWorld 752•(1)

-N-
NameOfClass 806•(1)
NeedConversion 872•(1)
NewDynObject 787•(1)
NewHeap 863•(1)
NewObject 786•(1)
NewOrRecycledObj 789•(1)
numElements 719•(1) 724•(1) 729•(1)
NumRecords 712•(1)

-O-
Obtain 588•(1) 609•(1) 631•(1) 687•(1)
onDesktop 747•(1)
Open 545•(1) 676•(1)
outputIndent 760•(1)
outputRMargin 145•(1)

-P-
PaslibCall 116•(1)
Passwd 101•(1)
password 496•(1)
path 495 (1)
PmDecl 113•(1)
Pmm 114•(1)
PopLast 397•(1)
Pos 374•(1) 417•(1) 458•(1)
position 532•(1)
PPasLibC 117•(1)
prcsDsBytes 141•(1)
prcsLdsn 140•(1)
ProcPtr 155•(1)
Ptr 154•(1) 155 (1) 156 (1) 168 (1) 169 (1) 423 (1) 424 (1) 426 (1) 609 (1)
PutAt 400•(1) 442•(1) 484•(1)

-R-
Read 212•(1)
ReadArray 641•(1)
ReadNumber 642•(1)
ReadObject 644•(1)
recordBytes 255•(1) 407 (1)
RecycleObject 790•(1)
refnum 666•(1)
Rename 523•(1)
Replace 595•(1) 616•(1) 638•(1) 694•(1)
ResizeColl 344•(1)
ResizeDynObject 788•(1)
Reverse 554•(1)

-S-
S255 161•(1) 163 (1) 172 (1)
S8 160•(1)
Scan 589•(1) 610•(1) 632•(1) 688•(1)
scanBackward 193•(1)
scanDone 535•(1)
scanForward 193•(1)
Scanner 375•(1) 418•(1) 459•(1) 514•(1)
ScannerFrom 376•(1) 419•(1) 460•(1) 515•(1)
scanners 500•(1)
Seek 555•(1) 683•(1)

SetHeap 867•(1)
ShiftColl 345•(1)
Size 734•(1)
size 234•(1) 250•(1) 262 (1)
SizeOfClass 807•(1)
SizeOfNumber 191•(1)
Skip 556•(1) 684•(1)
SplitFilePath 835•(1)
StartEdit 298•(1)
StopEdit 299•(1)
STRING 160 (1) 161 (1) 165 (1) 177 (1) 178 (1)
StrToInt 829•(1)
StrToLInt 828•(1)
StrUpperCased 833•(1)
SubObject 796•(1)
Superclass 803•(1)
SweepHeap 876•(1)
SysCall 99•(1)

-T-
TAccesses 188•(1) 665 (1)
TArray 181 (1) 182 (1) 183 (1) 184 (1) 404•(1) 410 (1) 427 (1) 641 (1) 708 (1)
TArrayHeader 248•(1)
TArrayScanner 418 (1) 420 (1) 599•(1) 605 (1)
TAuthorName 177•(1)
TBlockList 727•(1) 728 (1)
TClass 169•(1) 205 (1) 233 (1) 249 (1) 803 (1)
TClassName 178•(1)
TClassWorld 180•(1) 752 (1)
TCollecHeader 232•(1) 242 (1)
TCollection 259•(1) 290 (1) 349 (1) 404 (1) 446 (1) 492 (1) 531 (1)
TConvResult 195•(1)
TDynamicArray 708•(1) 711 (1)
TEnumAccesses 187•(1) 188 (1)
TFastString 241•(1) 245 (1)
TFile 492•(1) 504 (1)
TFilePart 165•(1) 166 (1)
TFilePath 163•(1) 495 (1)
TFileScanner 514 (1) 515 (1) 662•(1) 670 (1) 732 (1)
TFileStream 732•(1) 733 (1)
THeap 168•(1) 210 (1) 743 (1) 863 (1)
THFastString 246•(1)
TIndexList 716•(1) 718 (1)
TIOMode 189•(1)
TLinkList 722•(1) 723 (1)
TList 349•(1) 354 (1) 383 (1) 500 (1) 716 (1) 722 (1) 727 (1)
TListScanner 375 (1) 377 (1) 578•(1) 584 (1)
TObject 200•(1) 203 (1) 206 (1) 207 (1) 227 (1) 259 (1) 291 (1) 355 (1) 380 (1) 381 (1)
382 (1) 397 (1) 508 (1) 528 (1) 588 (1) 644 (1) 717 (1) 786 (1) 787 (1) 789 (1)
796 (1)
ToPStr 467•(1)
ToPStrAt 468•(1)
TPassword 166•(1) 496 (1)
TPFastString 245•(1) 246 (1)
TpINTEGER 174•(1)
TpLONGINT 175•(1)
TPString 172•(1)
TrimBlanks 831•(1)
TScanDirection 193•(1)
TScanner 528•(1) 540 (1) 578 (1) 599 (1) 620 (1)
TString 446•(1) 451 (1) 466 (1)
TStringScanner 459 (1) 460 (1) 620•(1) 627 (1) 662 (1)

-U-
UClascal 104•(1) 106•(1) 108•(1)
UnitAuthor 810•(1)
UnitHz 98•(1)
UnitStd 97•(1)
UObject 1•(1)

-V-
ValidObject 814•(1)
VerifyPassword 524•(1)

-W-
WhenModified 522•(1)
Width 488•(1)
wmIsInitialized 748•(1)
Write 213•(1)
WriteArray 645•(1)
WriteNumber 647•(1)
WriteObject 648•(1)
WrLn 853•(1)
WrObj 855•(1)
WrStr 852•(1)

-X-
XferContiguous 649•(1)
XferFields 651•(1)
XferLeft 778•(1)
XferPString 652•(1)
XferRandom 657•(1) 699•(1)
XferRight 779•(1)
XferSequential 655•(1) 698•(1)
xRead 190•(1)
xReadWrite 190•(1)
xWrite 190•(1)

*** End Xref: 264 id's 478 references [421720 bytes/4735 id's/43245 refs]

```
1     1 --  UNIT UDraw;
1     2 --  {Copyright 1983, 1984, Apple Computer, Inc.}
1     3 --
1     4 --  {changed 05/01 1503 Changes to allow people to use Clascal on the Workshop}
1     5 --
1     6 --  {$Setc IsIntrinsic := TRUE }
1     7 --
1     8 --  {$IFC IsIntrinsic}
1     9 --  INTRINSIC;
1    10 --  {$ENDC}
1    11 --
1    12 --  INTERFACE
1    13 --
1    14 --  USES
1    15 --      {$U UnitStd     } UnitStd,      {Client should not USE UnitStd}
1    16 --      {$U UnitHz      } UnitHz,       {Client should not USE UnitHz and MUST NOT USE Storage}
1    17 --      {$U libtk/UObject} UObject,         {Client must USE UObject}
1    18 --      {$U LIBOS/SysCall} SysCall,     {Client may USE SysCall}
1    19 --  {$IFC LibraryVersion > 10}
1    20 --      {$U LIBPL/PaslibCall} PaslibCall,
1    21 --      {$U LIBPL/PPasLibc } PPasLibC,
1    22 --  {$ENDC}
1    23 --  {$IFC LibraryVersion <= 20}
1    24 --      {$U FontMgr     } FontMgr,      {Client should USE UFont instead of FontMgr before QuickDraw}
1    25 --  {$ENDC}
1    26 --      {$U QuickDraw   } QuickDraw,    {Client must USE QuickDraw (unless we provide a type-stub for it)}
1    27 --  {$IFC LibraryVersion > 20}
1    28 --      {$U FontMgr     } FontMgr,      {Client should USE UFont instead of FontMgr after QuickDraw}
1    29 --  {$ENDC}
1    30 --      {$U WM.Events   } Events,
1    31 --      {$U WM.Folders  } Folders,
1    32 --      {$U FilerComm   } FilerComm;
1    33 --
1    34 --  {$SETC fDbgDraw     := fDbgOK}
1    35 --  {$SETC fRngDraw     := fDbgOK}
1    36 --  {$SETC fSymDraw     := fSymOK}
1    37 --
1    38 --  {$SETC fDebugMethods := fDbgDraw} {if VAR also true, trace entries and/or exits}
1    39 --
1    40 --  CONST
1    41 --      {there should be at most 10 families and they should be in consecutive order; otherwise
1    42 --              the command number constants in UABC should be changed}
1    43 --      famSystem       =   0;
1    44 --
1    45 --      famMin          =   1;  {minimum family number that appears in the font menu}
1    46 --      famModern       =   1;
1    47 --      famClassic      =   2;
1    48 --      famMax          =   2;
1    49 --
1    50 --      {there should be at most 20 families and they should be in consecutive order; otherwise
1    51 --              the command number constants in UABC should be changed}
1    52 --      sizeMin         =   1;
1    53 --      size20Pitch     =   1;  { 8 Point 20 Pitch      NOTE:  Modern available only}
1    54 --      size15Pitch     =   2;  { 8 Point 15 Pitch      NOTE:  Modern available only}
1    55 --      size12Pitch     =   3;  {10 Point 12 Pitch}
1    56 --      size10Pitch     =   4;  {12 Point 10 Pitch}
1    57 --      size12Point     =   5;  {12 Point proportional}
1    58 --      size14Point     =   6;  {14 Point proportional}
1    59 --      size18Point     =   7;  {18 Point proportional}
1    60 --      size24Point     =   8;  {24 Point proportional}
1    61 --      sizeMax         =   8;
1    62 --
1    63 --      {font IDs to be used in QuickDraw}
1    64 --      fIDSystem       =    0;     {Reserved for application generated text, that cannot be edited by user;
1    65 --                                      does not print properly}
1    66 --      fID20Pitch      =   19;
1    67 --      fID15Pitch      =    7;
1    68 --      fIDm12Pitch     =    8;
1    69 --      fIDc12Pitch     =   13;
1    70 --      fIDm10Pitch     =    9;
1    71 --      fIDc10Pitch     =   14;
1    72 --      fIDm12Point     =    4;
1    73 --      fIDc12Point     =   10;
1    74 --      fIDm14Point     =   15;
1    75 --      fIDc14Point     =   16;
1    76 --      fIDm18Point     =    5;
1    77 --      fIDc18Point     =   11;
1    78 --      fIDm24Point     =    6;
1    79 --      fIDc24Point     =   12;
1    80 --
1    81 --      fIDRulers       =   25;     {Ruler Icons}
1    82 --
1    83 --      {font IDs below this line are to be used only in special cases, there is no guarantee that these
1    84 --              will print properly}
1    85 --      fIDSysPatterns  =    2;     {System Patterns, ie. LisaDraw}
1    86 --      fIDSysCursors   =    3;     {System Cursors}
1    87 --      fIDLT20Graphics =   23;     {LisaTerminal 20 Pitch VT100 graphics}
1    88 --      fIDLT12Graphics =   17;     {LisaTerminal 12 Pitch VT100 graphics}
1    89 --      fIDLT20Text     =   27;     {LisaTerminal 20 Pitch VT100 text}
1    90 --      fIDLT12Text     =   26;     {LisaTerminal 12 Pitch VT100 text}
1    91 --      fIDDeskIcons    =   22;     {Desktop Icon font}
1    92 --      fIDWM           =    1;     {Window Manager font}
1    93 --      fIDCalculator   =   18;     {Calculator font}
1    94 --      fIDIconName     =   21;     {Icon Name font}
1    95 --      fIDMarker       =   20;     {Marker Font}
1    96 --      fIDLisaGuide    =   24;     {LisaGuide Font}
1    97 --
1    98 --
1    99 --  TYPE
1   100 --
1   101 --      TFontIDArray = ARRAY[famMin..famMax, sizeMin..sizeMax] OF INTEGER;
1   102 --
1   103 --      TScaler =
1   104 --          RECORD   {scale-definition}
1   105 --          numerator:      point; {numerator.h DIV denominator.h is the scale factor in horiz direction}
1   106 --          denominator:    point; {numerator.v DIV denominator.v is the scale factor in the vert. direction}
1   107 --          END;
1   108 --
1   109 --      TRectCoords = ARRAY[FALSE..TRUE] OF Point; {TRectCoords(aRect)[FALSE] = aRect.topLeft; [TRUE] = botRight}
1   110 --
```

```
1  111 --      LPoint =
1  112 --              RECORD
1  113 --                  CASE INTEGER OF
1  114 --                      0: (v, h: LONGINT);
1  115 --                      1: (vh: ARRAY [VHSelect] OF LONGINT)
1  116 --              END;
1  117 --
1  118 --      LRect =
1  119 --              RECORD
1  120 --                  CASE INTEGER OF
1  121 --                      0: (top, left, bottom, right: LONGINT);
1  122 --                      1: (topLeft, botRight: LPoint)
1  123 --              END;
1  124 --
1  125 --      LPattern = PACKED ARRAY[0..7] OF 0..255;
1  126 --
1  127 --      TLRectCoords = ARRAY[FALSE..TRUE] OF LPoint; {TLRectCoords(anLRect)[FALSE] = anLRect.topLeft; etc.}
1  128 --
1  129 --      TEnumActions = (rErase, rFrame, rBackground, rDraw);
1  130 --      TActions = SET OF TEnumActions;
1  131 --
1  132 --      THighTransit = (hNone, hOffToDim, hOffToOn, hDimToOn, hDimToOff, hOnToOff, hOnToDim);
1  133 --      {Refresh assumes that the last four and only the last four start with already-highlighted stuff}
1  134 --
1  135 --      TEnumResizability = (userCanResizeIt, windowCanResizeIt);
1  136 --      TResizability = SET OF TEnumResizability;       {arg for TBranchArea.CREATE & TPanel.Divide}
1  137 --
1  138 --      TFontRecord =
1  139 --          PACKED RECORD
1  140 --          CASE BOOLEAN OF
1  141 --              FALSE:  (fontNum:       INTEGER);
1  142 --              TRUE:   (fontFamily:    Byte;
1  143 --                       fontSize:      Byte)
1  144 --          END;
1  145 --
1  146 --      TTypeStyle =
1  147 --          RECORD
1  148 --  {$IFC LibraryVersion <= 20}
1  149 --          onFaces:    TSetoface;
1  150 --  {$ELSEC}
1  151 --          onFaces:    Style;
1  152 --  {$ENDC}
1  153 --          font:       TFontRecord;
1  154 --          END;
1  155 --
1  156 --      TArea = SUBCLASS OF TObject
1  157 --
1  158 --        {Variables}
1  159 --          innerRect:      Rect;                   {window(usually)-relative bounds excluding borders}
1  160 --          outerRect:      Rect;                   {bounding box in ancestral coordinates}
1  161 --          parentBranch:   TBranchArea;            {only used for TPanels and TBranchAreas}
1  162 --
1  163 --        {Creation/Destruction}
1  164 --          FUNCTION  TArea.CREATE(object: TObject; heap: THeap; itsRect: Rect): TArea; ABSTRACT;
1  165 --
1  166 --        {Attributes}
1  167 --          FUNCTION  TArea.ChildWithPt(pt: Point; childList: TList; VAR nearestPt: Point): TArea;
1  168 --          PROCEDURE TArea.GetBorder(VAR border: Rect); DEFAULT;
1  169 --                                  {Return the deltas of the border bars, etc. (outer=inner+border)}
1  170 --                                  {windows, bands, panes: 1 all around;
1  171 --                                   panels: 1 on left/top, scroll bars on right/bottom}
1  172 --          PROCEDURE TArea.GetMinExtent(VAR minExtent: Point; windowIsResizingIt: BOOLEAN); ABSTRACT;
1  173 --          PROCEDURE TArea.SetOuterRect(newOuterRect: Rect);
1  174 --          PROCEDURE TArea.SetInnerRect(newInnerRect: Rect);
1  175 --
1  176 --        {Display}                   {Other methods assume grafPort, origin, & clipping were preset by Focus}
1  177 --          PROCEDURE TArea.Erase;
1  178 --                                  {Erase the interior}
1  179 --          PROCEDURE TArea.Focus; ABSTRACT;
1  180 --                                  {Set up the grafPort for this window or pad}
1  181 --          PROCEDURE TArea.Frame; DEFAULT;
1  182 --                                  {Draw outlines, scroll bars, etc. outside the bounding box}
1  183 --          PROCEDURE TArea.Refresh(rActions: TActions; highTransit: THighTransit); ABSTRACT;
1  184 --
1  185 --        {Buttoning}
1  186 --          FUNCTION  TArea.DownAt(mousePt: Point): BOOLEAN; ABSTRACT;
1  187 --
1  188 --        {Resizing}
1  189 --          PROCEDURE TArea.ResizeInside(newInnerRect: Rect); ABSTRACT;
1  190 --          PROCEDURE TArea.ResizeOutside(newOuterRect: Rect); ABSTRACT;
1  191 --
1  192 --          END;
1  193 --
1  194 --
1  195 --      TPad = SUBCLASS OF TArea
1  196 --
1  197 --        {Variables}
1  198 --          port:           GrafPtr;                {the GrafPort used by this pad}
1  199 --          viewedLRect:    LRect;                  {The portion of view that is displayed in innerRect}
1  200 --          visLRect:       LRect;                  {viewedLRect sect visRgn while focused}
1  201 --          availLRect:     LRect;                  {The larger part of view that fits in a 16-bit Rect}
1  202 --          scrollOffset:   LPoint;                 {The distance scrolled from the view topLeft}
1  203 --          origin:         Point;                  {What to set the grafport origin to when focused}
1  204 --          cdOffset:       LPoint;                 {What to subtract from coordinates to get port coords}
1  205 --          clippedRect:    rect;                   {additional clipping to apply at Focus time}
1  206 --
1  207 --          padRes:         Point;                  {spots/inch in the pad coordinate space}
1  208 --          viewedRes:      Point;                  {spots/inch in the 32-bit space being projected}
1  209 --
1  210 --          scaled:         BOOLEAN;                {the net scale factor, combining zooming}
1  211 --          scaleFactor:    TScaler;                {and aspect ratio, etc.}
1  212 --
1  213 --          zoomFactor:     TScaler;
1  214 --
1  215 --        {Creation/Destruction}
1  216 --          FUNCTION  TPad.CREATE(object: TObject; heap: THeap; itsInnerRect: Rect; itsViewedLRect: LRect;
1  217 --                                itsPadRes, itsViewRes: Point;
1  218 --                                itsPort: GrafPtr): TPad;
1  219 --
1  220 --          PROCEDURE TPad.Redefine(itsInnerRect: Rect; itsViewedLRect: LRect;
```

```
1  221 --                          itsPadRes, itsViewRes: Point;
1  222 --                      itsZoomFactor: TScaler; itsPort: GrafPtr);
1  223 --
1  224 --
1  225 --        {Coordinate Mapping -- grafPort to view}
1  226 --          PROCEDURE TPad.DistToLDist(distInPort: Point; VAR lDistInView: LPoint);
1  227 --          PROCEDURE TPad.PatToLPat(patInPort: Pattern; VAR lPatInView: LPattern);
1  228 --          PROCEDURE TPad.PtToLPt(ptInPort: Point; VAR lPtInView: LPoint);
1  229 --          PROCEDURE TPad.RectToLRect(rectInPort: Rect; VAR lRectInView: LRect);
1  230 --
1  231 --        {Coordinate Mapping -- view to grafPort}
1  232 --          PROCEDURE TPad.LDistToDist(lDistInView: LPoint; VAR distInPort: Point);
1  233 --          PROCEDURE TPad.LPatToPat(lPatInView: LPattern; VAR patInPort: Pattern);
1  234 --          PROCEDURE TPad.LPtToPt(lPtInView: LPoint; VAR ptInPort: Point);
1  235 --          PROCEDURE TPad.LRectToRect(lRectInView: LRect; VAR rectInPort: Rect);
1  236 --
1  237 --        {Scrolling}
1  238 --          PROCEDURE TPad.OffsetBy(deltaLPt: LPoint);    {offset viewedLRect -- no effect on display}
1  239 --          PROCEDURE TPad.SetScrollOffset(VAR newOffset: LPoint);
1  240 --                      {recalculates the origin and cdOffset fields; does not change arg}
1  241 --
1  242 --        {Display}
1  243 --          PROCEDURE TPad.ClipFurtherTo(rBand: rect);  {narrows down clip area at next Focus}
1  244 --          PROCEDURE TPad.Focus; OVERRIDE;
1  245 --          PROCEDURE TPad.InvalLRect(r: LRect);              {Force redraw of r at next update}
1  246 --          PROCEDURE TPad.InvalRect(r: Rect);                {Force redraw of r at next update}
1  247 --          PROCEDURE TPad.SetPen(pen: PenState);    {NB: We should later augment this so that it scales
1  248 --                                                        pensizes}
1  249 --          PROCEDURE TPad.SetPenToHighlight(highTransit: THighTransit); {SetPenState to highlight this way}
1  250 --
1  251 --          PROCEDURE TPad.SetZoomFactor(zoomNumerator, zoomDenominator: point); DEFAULT;
1  252 --
1  253 --        {Drawing}
1  254 --          PROCEDURE TPad.DrawLText(textBuf: Ptr; startByte, numBytes: INTEGER);
1  255 --          PROCEDURE TPad.DrawLLine(newLPt: LPoint);
1  256 --          PROCEDURE TPad.DrawLPicture(pic: PicHandle; r:LRect);
1  257 --          PROCEDURE TPad.DrawLRect(verb: GrafVerb; r: LRect);
1  258 --          PROCEDURE TPad.DrawLRRect(verb: GrafVerb; r: LRect; ovalWidth, ovalHeight: INTEGER);
1  259 --          PROCEDURE TPad.DrawLOval(verb: GrafVerb; r: LRect);
1  260 --          PROCEDURE TPad.DrawLArc(verb: GrafVerb; r: LRect; startAngle, arcAngle: INTEGER);
1  261 --          PROCEDURE TPad.DrawLBits(VAR srcBits: BitMap; VAR srcRect: Rect;
1  262 --                                   VAR dstLRect: LRect; mode: INTEGER; maskRgn: RgnHandle);
1  263 --
1  264 --        {Process termination and Debugging Assistance}
1  265 --          PROCEDURE TPad.Crash; ABSTRACT;
1  266 --          FUNCTION  TPad.BindHeap(activeVsClip, doBind: BOOLEAN): THeap; ABSTRACT;
1  267 --
1  268 --          END;
1  269 --
1  270 --      TBranchArea = SUBCLASS OF TArea
1  271 --
1  272 --        {Variables}
1  273 --          arrangement:    VHSelect;       {v means above one another}
1  274 --          elderFirst:     BOOLEAN;        {TRUE IFF elderChild is above or to the left of youngerChild}
1  275 --          resizability:   TResizability;
1  276 --          elderChild:     TArea;
1  277 --          youngerChild:   TArea;
1  278 --
1  279 --        {Creation/Destruction}
1  280 --          FUNCTION  TBranchArea.CREATE(object: TObject; heap: THeap; vhs: VHSelect; hasElderFirst: BOOLEAN;
1  281 --                                       whoCanResizeIt: TResizability;
1  282 --                                       itsElderChild, itsYoungerChild: TArea): TBranchArea;
1  283 --
1  284 --        {Attributes}
1  285 --          PROCEDURE TBranchArea.GetMinExtent(VAR minExtent: Point; windowIsResizingIt: BOOLEAN); OVERRIDE;
1  286 --          FUNCTION  TBranchArea.OtherChild(child: TArea): TArea;
1  287 --          PROCEDURE TBranchArea.ReplaceChild(child, newChild: TArea);
1  288 --          FUNCTION  TBranchArea.TopLeftChild: TArea;
1  289 --
1  290 --        {Resizing}
1  291 --          PROCEDURE TBranchArea.ResizeOutside(newOuterRect: Rect); OVERRIDE;
1  292 --          PROCEDURE TBranchArea.Redivide(newCd: INTEGER);
1  293 --
1  294 --          END;
1  295 --
1  296 --
1  297 --
1  298 --    VAR
1  299 --
1  300 --        amPrinting:         BOOLEAN;                    {Iff TRUE, we are currently printing rather than drawing}
1  301 --
1  302 --        zeroPt:             Point;                      {(0,0)}
1  303 --        zeroRect:           Rect;                       {(0,0)-(0,0)}
1  304 --        hugeRect:           Rect;                       {(0,0)-(MAXINT/2, MAXINT/2)}
1  305 --
1  306 --        zeroLPt:            LPoint;                     {(0,0)}
1  307 --        zeroLRect:          LRect;                      {(0,0)-(0,0)}
1  308 --        hugeLRect:          LRect;                      {(0,0)-(MAXLINT/2, MAXLINT/2)}
1  309 --
1  310 --        orthogonal:         ARRAY [v..h] OF VHSelect;   {Maps v to h and vice versa}
1  311 --
1  312 --        highPen:            ARRAY [THighTransit] OF PenState; {standard highlight-feedback transitions}
1  313 --
1  314 --        lPatWhite:          LPattern;                   {Maps to QuickDraw pattern white}
1  315 --        lPatBlack:          LPattern;                   {Maps to QuickDraw pattern black}
1  316 --        lPatGray:           LPattern;                   {Maps to QuickDraw pattern gray}
1  317 --        lPatLtGray:         LPattern;                   {Maps to QuickDraw pattern ltGray}
1  318 --        lPatDkGray:         LPattern;                   {Maps to QuickDraw pattern dkGray}
1  319 --
1  320 --        focusStack:         ARRAY [1..10] OF TArea;     {PushFocus pushes and PopFocus pops focusArea}
1  321 --        focusStkPtr:        INTEGER;                    {Index of last thing on focusStack}
1  322 --        focusArea:          TArea;                      {The currently focused area}
1  323 --        focusRgn:           RgnHandle;                  {either padRgn or visRgn}
1  324 --        padRgn:             RgnHandle;                  {intersection of pane and visRgn}
1  325 --        altVisRgn:          RgnHandle;                  {If useAltVisRgn, use this instead of visRgn in Focus}
1  326 --        useAltVisRgn:       BOOLEAN;                    {If TRUE, use altVisRgn instead of visRgn in Focus}
1  327 --        thePad:             TPad;                       {focusArea, if a TPad, else NIL}
1  328 --
1  329 --        noPad:              TPad;                       {maps every point to itself}
1  330 --        crashPad:           TPad;                       {an object willing to handle process termination}
```

```
1  331 --
1  332 --      screenRes:          Point;                      {screen resolution, pixels per inch}
1  333 --
1  334 --      sysTypeStyle:       TTypeStyle; {system font, normal face}
1  335 --
1  336 --      printerPseudoPort:  GrafPtr;
1  337 --
1  338 --      cArea:              TClass;
1  339 --      cPad:               TClass;
1  340 --      cBranchArea:        TClass;
1  341 --
1  342 --
1  343 --
1  344 --  {The next four are declared EXTERNAL in UOBJECT2}
1  345 --  PROCEDURE InitQDAM;
1  346 --  PROCEDURE TrmntExceptionHandler;
1  347 --  PROCEDURE InitErrorAbort(error: INTEGER);
1  348 --  {$IFC fDbgDraw}
1  349 --  FUNCTION  BindHeap(activeVsClip, doBind: BOOLEAN): THeap;
1  350 --  {$ENDC}
1  351 --
1  352 --  PROCEDURE Reduce(VAR numerator, denominator: INTEGER);   {reduce fraction to lowest terms}
1  353 --
1  354 --  FUNCTION  FPtPlusPt(operand1, operand2: Point): LONGINT;    {      p3 := Point(FPtPlusPt(p1, p2));      }
1  355 --  FUNCTION  FPtMinusPt(operand1, operand2: Point): LONGINT;   { F stands for FUNCTION }
1  356 --
1  357 --  FUNCTION  FPtMulInt(operand1: Point; operand2: INTEGER): LONGINT;
1  358 --  FUNCTION  FPtDivInt(operand1: Point; operand2: INTEGER): LONGINT;
1  359 --              {e.g.:        center := Point(FPtDivInt(POINT(FPtPlusPt(p1, p2)), 2);          }
1  360 --
1  361 --  FUNCTION  FPtMaxPt(operand1, operand2: Point): LONGINT;     { each coordinate is max'ed separately }
1  362 --  FUNCTION  FPtMinPt(operand1, operand2: Point): LONGINT;
1  363 --
1  364 --  FUNCTION  FDiagRect(operand1: Rect): LONGINT;               { FPtMinusPt(botRight-topLeft) }
1  365 --
1  366 --  PROCEDURE BoolToStr(bool: BOOLEAN; str: TPstring);
1  367 --
1  368 --  FUNCTION  LIntDivLInt(i, j: LONGINT): LONGINT;
1  369 --  FUNCTION  LIntDivInt(i: LONGINT; j: INTEGER): LONGINT;
1  370 --  FUNCTION  LIntMulInt(i: LONGINT; j: INTEGER): LONGINT;
1  371 --
1  372 --  FUNCTION  LIntOvrInt(i: LONGINT; j: INTEGER): LONGINT;
1  373 --      {This returns LIntDivInt(i+(j DIV 2), j) if i>0 and
1  374 --                    LIntDivInt(i-(j DIV 2), j) if i<0}
1  375 --
1  376 --  PROCEDURE PtPlusPt(operand1, operand2: Point; VAR result: Point);
1  377 --  PROCEDURE PtMinusPt(operand1, operand2: Point; VAR result: Point);
1  378 --  PROCEDURE PtMulInt(operand1: Point; operand2: INTEGER; VAR result: Point);
1  379 --  PROCEDURE PtDivInt(operand1: Point; operand2: INTEGER; VAR result: Point);
1  380 --  {$IFC LibraryVersion <= 20}
1  381 --  FUNCTION  EqualPt(operand1, operand2: Point): BOOLEAN;  {Will be in QuickDraw eventually}
1  382 --  {$ENDC}
1  383 --
1  384 --  PROCEDURE RectPlusRect(operand1, operand2: Rect; VAR result: Rect);
1  385 --  PROCEDURE RectMinusRect(operand1, operand2: Rect; VAR result: Rect);
1  386 --  {$IFC LibraryVersion <= 20}
1  387 --  FUNCTION  EqualRect(rectA, rectB: Rect): BOOLEAN;        {Will be in QuickDraw eventually}
1  388 --  FUNCTION  EmptyRect(r: Rect): BOOLEAN;                   {Will be in QuickDraw eventually}
1  389 --  {$ENDC}
1  390 --
1  391 --  PROCEDURE AlignRect(VAR dstRect: Rect; srcRect: Rect; vhs: VHSelect);
1  392 --  FUNCTION  LengthRect(r: Rect; vhs: VHSelect): INTEGER;
1  393 --  FUNCTION  RectHasPt(dstRect: Rect; pt: Point): BOOLEAN;
1  394 --  PROCEDURE RectHavePt(dstRect: Rect; VAR pt: Point);     {change pt so that topLeft <= pt <= botRight}
1  395 --  FUNCTION  RectsNest(outer, inner: Rect): BOOLEAN;
1  396 --  PROCEDURE RectifyRect(VAR dstRect: Rect);               {exchange coordinates until topLeft <= botRight}
1  397 --  FUNCTION  RectIsVisible(rectInPort: Rect): BOOLEAN;
1  398 --
1  399 --  PROCEDURE PointToStr(pt: Point; str: TPstring);    {Referenced as EXTERNAL by UABC2}
1  400 --  PROCEDURE RectToStr(r: Rect; str: TPstring);       {Referenced as EXTERNAL by UABC2}
1  401 --
1  402 --  PROCEDURE LPtPlusLPt(operand1, operand2: LPoint; VAR result: LPoint);
1  403 --  PROCEDURE LPtMinusLPt(operand1, operand2: LPoint; VAR result: LPoint);
1  404 --  PROCEDURE LPtMulInt(operand1: LPoint; operand2: INTEGER; VAR result: LPoint);
1  405 --  PROCEDURE LPtDivInt(operand1: LPoint; operand2: INTEGER; VAR result: LPoint);
1  406 --  FUNCTION  EqualLPt(operand1, operand2: LPoint): BOOLEAN;
1  407 --
1  408 --  PROCEDURE LRectPlusLRect(operand1, operand2: LRect; VAR result: LRect);
1  409 --  PROCEDURE LRectMinusLRect(operand1, operand2: LRect; VAR result: LRect);
1  410 --  FUNCTION  EqualLRect(rectA, rectB: LRect): BOOLEAN;
1  411 --  FUNCTION  EmptyLRect(r: LRect): BOOLEAN;
1  412 --
1  413 --  PROCEDURE AlignLRect(VAR destLRect: LRect; srcLRect: LRect; vhs: VHSelect);
1  414 --  FUNCTION  LengthLRect(r: LRect; vhs: VHSelect): LONGINT;
1  415 --  FUNCTION  LRectHasLPt(destLRect: LRect; pt: LPoint): BOOLEAN;
1  416 --  PROCEDURE LRectHaveLPt(destLRect: LRect; VAR pt: LPoint);    {change pt so that topLeft <= pt <= botRight}
1  417 --  FUNCTION  LRectsNest(outer, inner: LRect): BOOLEAN;
1  418 --  PROCEDURE RectifyLRect(VAR destLRect: LRect);           {exchange coordinates until topLeft <= botRight}
1  419 --  FUNCTION  LRectIsVisible(srcLRect: LRect): BOOLEAN;
1  420 --
1  421 --  PROCEDURE LPointToStr(pt: LPoint; str: TPstring);   {Referenced as EXTERNAL by UOBJECT2}
1  422 --  PROCEDURE LRectToStr(r: LRect; str: TPstring);      {Referenced as EXTERNAL by UOBJECT2}
1  423 --
1  424 --  PROCEDURE SetLPt(VAR destPt: LPoint; itsH, itsV: LONGINT);
1  425 --  PROCEDURE SetLRect(VAR dstRect: LRect; itsLeft, itsTop, itsRight, itsBottom: LONGINT);
1  426 --  PROCEDURE OffsetLRect(VAR dstRect: LRect; dh, dv: LONGINT);
1  427 --  PROCEDURE InsetLRect(VAR dstRect: LRect; dh, dv: LONGINT);
1  428 --  FUNCTION  SectLRect(srcRectA, srcRectB: LRect; VAR dstRect: LRect): BOOLEAN;
1  429 --  PROCEDURE UnionLRect(srcRectA, srcRectB: LRect; VAR dstRect: LRect);
1  430 --  FUNCTION  LPtInLRect(pt: LPoint; r: LRect): BOOLEAN;
1  431 --
1  432 --
1  433 --  FUNCTION  IsSmallPt(srcPt: LPoint): BOOLEAN;
1  434 --  FUNCTION  IsSmallRect(srcRect: LRect): BOOLEAN;
1  435 --
1  436 --
1  437 --  (*PROCEDURE ClipLRect(r: LRect);  {Not yet implementable}*)
1  438 --
1  439 --
1  440 --  {Drawing text}
```

```
1  441 --
1  442 --   PROCEDURE DrawLText(textBuf: Ptr; startByte, numBytes: INTEGER);
1  443 --
1  444 --   {Drawing lines, rectangles, and ovals}
1  445 --
1  446 --   PROCEDURE MoveToL(h, v: LONGINT);
1  447 --   PROCEDURE MoveL(dh, dv: LONGINT);
1  448 --   PROCEDURE LineToL(h, v: LONGINT);
1  449 --   PROCEDURE LineL(dh, dv: LONGINT);
1  450 --
1  451 --   PROCEDURE FrameLRect(r: LRect);
1  452 --   PROCEDURE PaintLRect(r: LRect);
1  453 --   PROCEDURE EraseLRect(r: LRect);
1  454 --   PROCEDURE InvrtLRect(r: LRect);
1  455 --   PROCEDURE FillLRect(r: LRect; lPat: LPattern);
1  456 --
1  457 --   PROCEDURE FrameLOval(r: LRect);
1  458 --   PROCEDURE PaintLOval(r: LRect);
1  459 --   PROCEDURE EraseLOval(r: LRect);
1  460 --   PROCEDURE InvrtLOval(r: LRect);
1  461 --   PROCEDURE FillLOval(r: LRect; lPat: LPattern);
1  462 --
1  463 --   PROCEDURE FrameLRRect(r: LRect; ovalWidth, ovalHeight: INTEGER);
1  464 --   PROCEDURE PaintLRRect(r: LRect; ovalWidth, ovalHeight: INTEGER);
1  465 --   PROCEDURE EraseLRRect(r: LRect; ovalWidth, ovalHeight: INTEGER);
1  466 --   PROCEDURE InvrtLRRect(r: LRect; ovalWidth, ovalHeight: INTEGER);
1  467 --   PROCEDURE FillLRRect(r: LRect; ovalWidth, ovalHeight: INTEGER; lPat: LPattern);
1  468 --
1  469 --   PROCEDURE FrameLArc(r: LRect; startAngle, arcAngle: INTEGER);
1  470 --   PROCEDURE PaintLArc(r: LRect; startAngle, arcAngle: INTEGER);
1  471 --   PROCEDURE EraseLArc(r: LRect; startAngle, arcAngle: INTEGER);
1  472 --   PROCEDURE InvrtLArc(r: LRect; startAngle, arcAngle: INTEGER);
1  473 --   PROCEDURE FillLArc(r: LRect; startAngle, arcAngle: INTEGER; lPat: LPattern);
1  474 --
1  475 --   FUNCTION  ClonePicture(pic: PicHandle; toHeap: THeap): PicHandle;
1  476 --
1  477 --   PROCEDURE ResizeFeedback(mousePt: Point; minPt, maxPt: Point; outerRect: Rect;
1  478 --                            tabHeight, sbWidth, sbHeight: INTEGER; VAR newPt: Point);
1  479 --
1  480 --   PROCEDURE PushFocus;              {Save old focusArea on focusStack}
1  481 --   PROCEDURE PopFocus;               {Restore old focusArea from focusStack and focus on it}
1  482 --
1  483 --   {$IFC LibraryVersion <= 20}
1  484 --   PROCEDURE MakeTypeStyle(itsFamily: INTEGER; itsSize: INTEGER;
1  485 --               itsFaces: TSetEFace;
1  486 --               VAR typeStyle: TTypeStyle);
1  487 --   {$ELSEC}
1  488 --   PROCEDURE MakeTypeStyle(itsFamily: INTEGER; itsSize: INTEGER;
1  489 --               itsFaces: Style;
1  490 --               VAR typeStyle: TTypeStyle);
1  491 --   {$ENDC}
1  492 --
1  493 --   FUNCTION  QDFontNumber(typeStyle: TTypeStyle): INTEGER;
1  494 --   PROCEDURE SetQDTypeStyle(typeStyle: TTypeStyle);
1  495 --
1  496 --   IMPLEMENTATION
1  497 --
1  498 --   {$I libtk/UDRAW2.TEXT}
2    1 --
2    2 --
1  499 --
1  500 --   END.
1  501 --
1  502 --
1  503 --
1  504 --
```

1. libtk/udraw.TEXT
2. libtk/UDRAW2.TEXT

-A-
AlignLRect 413•(1)
AlignRect 391•(1)
altVisRgn 325•(1)
amPrinting 300•(1)
arrangement 273 (1)
availLRect 281•(1)

-B-
BindHeap 266•(1) 349•(1)
BoolToStr 366•(1)
botRight 122•(1)
bottom 121•(1)
Byte 142 (1) 143 (1)

-C-
cArea 338•(1)
cBranchArea 340•(1)
cdOffset 204•(1)
ChildWithPt 167•(1)
ClipFurtherTo 243•(1)
clippedRect 205•(1)
ClonePicture 475•(1)
cPad 339•(1)
Crash 265•(1)
crashPad 330•(1)
CREATE 164•(1) 216•(1) 280•(1)

-D-
denominator 106•(1)
DistToLDist 226•(1)
DownAt 186•(1)
DrawLArc 260•(1)
DrawLBits 261•(1)
DrawLLine 255•(1)
DrawLOval 259•(1)
DrawLPicture 256•(1)
DrawLRect 257•(1)
DrawLRRect 258•(1)
DrawLText 254•(1) 442•(1)

-E-
elderChild 276•(1)
elderFirst 274•(1)
EmptyLRect 411•(1)
EmptyRect 388•(1)
EqualLPt 406•(1)
EqualLRect 410•(1)
EqualPt 381•(1)
EqualRect 387•(1)
Erase 177•(1)
EraseLArc 471•(1)
EraseLOval 459•(1)
EraseLRect 453•(1)
EraseLRRect 465•(1)
Events 30•(1)

-F-
famClassic 47•(1)
famMax 48•(1) 101 (1)
famMin 45•(1) 101 (1)
famModern 46•(1)
famSystem 43•(1)
FDiagRect 364•(1)
fID15Pitch 67•(1)
fID20Pitch 66•(1)
fIDc10Pitch 71•(1)
fIDc12Pitch 69•(1)
fIDc12Point 73•(1)
fIDc14Point 75•(1)
fIDc18Point 77•(1)
fIDc24Point 79•(1)
fIDCalculator 93•(1)
fIDDeskIcons 91•(1)
fIDIconName 94•(1)
fIDLisaGuide 96•(1)
fIDLT12Graphics 88•(1)
fIDLT12Text 90•(1)
fIDLT20Graphics 87•(1)
fIDLT20Text 89•(1)
fIDm10Pitch 70•(1)
fIDm12Pitch 68•(1)
fIDm12Point 72•(1)
fIDm14Point 74•(1)
fIDm18Point 76•(1)
fIDm24Point 78•(1)
fIDMarker 95•(1)
fIDRulers 81•(1)
fIDSysCursors 86•(1)
fIDSysPatterns 85•(1)
fIDSystem 64•(1)
fIDWH 92•(1)
FilerComm 32•(1)
FillLArc 473•(1)
FillLOval 461•(1)
FillLRect 455•(1)
FillLRRect 467•(1)
Focus 179•(1) 244•(1)
focusArea 322•(1)
focusRgn 323•(1)
focusStack 320•(1)
focusStkPtr 321•(1)
Folders 31•(1)
font 153•(1)
fontFamily 142•(1)
FontMgr 24•(1) 28•(1)
fontNum 141•(1)

fontSize 143•(1)
FPtDivInt 358•(1)
FPtMaxPt 361•(1)
FPtMinPt 362•(1)
FPtMinusPt 355•(1)
FPtMulInt 357•(1)
FPtPlusPt 354•(1)
Frame 181•(1)
FrameLArc 469•(1)
FrameLOval 457•(1)
FrameLRect 451•(1)
FrameLRRect 463•(1)

-G-
GetBorder 168•(1)
GetMinExtent 172•(1) 285•(1)
GrafPtr 198 (1) 336 (1)

-H-
h 114•(1) 310 (1)
hDimToOff 132•(1)
hDimToOn 132•(1)
highPen 312•(1)
hNone 132•(1)
hOffToDim 132•(1)
hOffToOn 132•(1)
hOnToDim 132•(1)
hOnToOff 132•(1)
hugeLRect 308•(1)
hugeRect 304•(1)

-I-
InitErrorAbort 347•(1)
InitQDWM 345•(1)
innerRect 159 (1)
InsetLRect 427•(1)
INTRINSIC 9•(1)
InvalLRect 245•(1)
InvalRect 246•(1)
InvrtLArc 472•(1)
InvrtLOval 460•(1)
InvrtLRect 454•(1)
InvrtLRRect 466•(1)
IsSmallPt 433•(1)
IsSmallRect 434•(1)

-L-
LDistToDist 232•(1)
left 121•(1)
LengthLRect 414•(1)
LengthRect 392•(1)
LineL 449•(1)
LineToL 448•(1)
LIntDivInt 369•(1)
LIntDivLInt 368•(1)
LIntMulInt 370•(1)
LIntOvrInt 372•(1)
lPatBlack 315•(1)
lPatDkGray 318•(1)
lPatGray 316•(1)
lPatLtGray 317•(1)
LPattern 125•(1) 314 (1) 315 (1) 316 (1) 317 (1) 318 (1)
LPatToPat 233•(1)
LPatWhite 314•(1)
LPoint 111•(1) 122 (1) 127 (1) 202 (1) 204 (1) 306 (1)
LPointToStr 421•(1)
LPtDivInt 405•(1)
LPtInLRect 430•(1)
LPtMinusLPt 403•(1)
LPtMulInt 404•(1)
LPtPlusLPt 402•(1)
LPtToPt 234•(1)
LRect 118•(1) 199 (1) 200 (1) 201 (1) 307 (1) 308 (1)
LRectHasLPt 415•(1)
LRectHaveLPt 416•(1)
LRectIsVisible 419•(1)
LRectMinusLRect 409•(1)
LRectPlusLRect 408•(1)
LRectsNest 417•(1)
LRectToRect 235•(1)
LRectToStr 422•(1)

-M-
MakeTypeStyle 484•(1) 488•(1)
MoveL 447•(1)
MoveToL 446•(1)

-N-
noPad 329•(1)
numerator 105•(1)

-O-
OffsetBy 238•(1)
OffsetLRect 426•(1)
onFaces 149•(1) 151•(1)
origin 203•(1)
orthogonal 310•(1)
OtherChild 286•(1)
outerRect 160•(1)

-P-
padRes 207•(1)
padRgn 324•(1)
PaintLArc 470•(1)
PaintLOval 458•(1)
PaintLRect 452•(1)
PaintLRRect 464•(1)
parentBranch 161•(1)
PaslibCall 20•(1)
PatToLPat 227•(1)

PenState 312 (1)
PicHandle 475 (1)
Point 109 (1) 203 (1) 207 (1) 208 (1) 302 (1) 332 (1)
point 105 (1) 106 (1)
PointToStr 399•(1)
PopFocus 481•(1)
port 198 (1)
PPasLibC 21•(1)
printerPseudoPor 336•(1)
PtDivInt 379•(1)
PtMinusPt 377•(1)
PtMulInt 378•(1)
PtPlusPt 376•(1)
PtToLPt 228•(1)
PushFocus 480•(1)

-Q-
QDFontNumber 493•(1)
QuickDraw 26•(1)

-R-
rBackground 129•(1)
rDraw 129•(1)
Rect 159 (1) 160 (1) 303 (1) 304 (1)
rect 205 (1)
RectHasPt 393•(1)
RectHavePt 394•(1)
RectifyLRect 418•(1)
RectifyRect 396•(1)
RectIsVisible 397•(1)
RectMinusRect 385•(1)
RectPlusRect 384•(1)
RectsNest 395•(1)
RectToLRect 229•(1)
RectToStr 400•(1)
Redefine 220•(1)
Redivide 292•(1)
Reduce 352•(1)
Refresh 183•(1)
ReplaceChild 287•(1)
rErase 129•(1)
resizability 275•(1)
ResizeFeedback 477•(1)
ResizeInside 189•(1)
ResizeOutside 190•(1) 291•(1)
rFrame 129•(1)
RgnHandle 323 (1) 324 (1) 325 (1)
right 121•(1)

-S-
scaled 210•(1)
scaleFactor 211•(1)
screenRes 332•(1)
scrollOffset 202•(1)
SectLRect 428•(1)
SetInnerRect 174•(1)
SetLPt 424•(1)
SetLRect 425•(1)
SetOuterRect 173•(1)
SetPen 247•(1)
SetPenToHighligh 249•(1)
SetQDTypeStyle 494•(1)
SetScrollOffset 239•(1)
SetZoomFactor 251•(1)
size10Pitch 56•(1)
size12Pitch 55•(1)
size12Point 57•(1)
size14Point 58•(1)
size15Pitch 54•(1)
size18Point 59•(1)
size20Pitch 53•(1)
size24Point 60•(1)
sizeMax 61•(1) 101 (1)
sizeMin 52•(1) 101 (1)
Style 151 (1)
SysCall 18•(1)
sysTypeStyle 334•(1)

-T-
TActions 130•(1)
TArea 156•(1) 164 (1) 167 (1) 195 (1) 270 (1) 276 (1) 277 (1) 286 (1) 288 (1) 320 (1)
322 (1)
TBranchArea 161 (1) 270•(1) 282 (1)
TClass 338 (1) 339 (1) 340 (1)
TEnumActions 129•(1) 130 (1)
TEnumResizabilit 135•(1) 136 (1)
TFontIDArray 101•(1)
TFontRecord 138•(1) 153 (1)
THeap 266 (1) 349 (1)
thePad 327•(1)
THighTransit 132•(1) 312 (1)
TLRectCoords 127•(1)
TObject 156 (1)
top 121•(1)
topLeft 122•(1)
TopLeftChild 288•(1)
TPad 195•(1) 218 (1) 327 (1) 329 (1) 330 (1)
TRectCoords 109•(1)
TResizability 136•(1) 275 (1)
TrmntExceptionHa 346•(1)
TScaler 103•(1) 211 (1) 213 (1)
TSeteface 149 (1)
TTypeStyle 146•(1) 334 (1)

-U-
UDraw 1•(1)
UnionLRect 429•(1)
UnitHz 16•(1)
UnitStd 15•(1)
UObject 17•(1)

```
  useAltVisRgn          326*( 1)
  userCanResizeIt       135*( 1)

-V-
  v                     114*( 1)  310 ( 1)
  vh                    115*( 1)
  VHSelect              115 ( 1)  273 ( 1)  310 ( 1)
  viewedLRect           199*( 1)
  viewedRes             208*( 1)
  visLRect              200*( 1)

-W-
  windowCanResizeI      135*( 1)

-Y-
  youngerChild          277*( 1)

-Z-
  zeroLPt               306*( 1)
  zeroLRect             307*( 1)
  zeroPt                302*( 1)
  zeroRect              303*( 1)
  zoomFactor            213*( 1)

*** End Xref: 304 id's   377 references      [422336 bytes/4695 id's/43402 refs]
```

```
1     1 --  {UNIT UABC}
1     2 --  {Copyright 1983, 1984, Apple Computer, Inc.}
1     3 --
1     4 --                        { *** METHODS NEED TO BE GROUPED INTO RIGHT CATEGORIES *** }
1     5 --                        { *** ADD reserve IN ALMOST EVERY CLASS ***}
1     6 --
1     7 --  { UABC2.TEXT  TProcess-TDocDirectory-TDocManager-TClipboard-TCommand-TCutCopyCommand-TPasteCommand}
1     8 --  { UABC3.TEXT  TImage-TView-TPaginatedView-TPageView-TPrintManager-THeading-TSelection}
1     9 --  { UABC4.TEXT  TWindow-TDialogBox-TMenuBar-TFont}
1    10 --  { UABC5.TEXT  TPanel-TBand-TPane-TMarginPad-TBodyPad-TScroller-TScrollBar}
1    11 --
1    12 --
1    13 --  UNIT UABC;
1    14 --
1    15 --  {$SETC IsIntrinsic := TRUE }
1    16 --
1    17 --  {$IFC IsIntrinsic}
1    18 --  INTRINSIC;
1    19 --  {$ENDC}
1    20 --
1    21 --  INTERFACE
1    22 --
1    23 --
1    24 --  USES
1    25 --      {$U UnitStd      } UnitStd,       {Client should not USE UnitStd}
1    26 --      {$U UnitHz       } UnitHz,        {Client should not USE UnitHz and MUST NOT USE Storage}
1    27 --      {$U libtk/UObject      } UObject,       {Client must USE UObject}
1    28 --      {$U LIBOS/SysCall} SysCall,      {Client may USE SysCall}
1    29 --  {$IFC LibraryVersion > 20}
1    30 --      {$U LIBTK/Passwd} Passwd,
1    31 --  {$ENDC}
1    32 --  {$IFC LibraryVersion <= 20}
1    33 --      {$U FontMgr      } FontMgr,       {Client should USE UFont instead of FontMgr before QuickDraw}
1    34 --  {$ENDC}
1    35 --      {$U QuickDraw    } QuickDraw,     {Client must USE QuickDraw (unless we provide a type-stub for it)}
1    36 --  {$IFC LibraryVersion > 20}
1    37 --      {$U FontMgr      } FontMgr,       {Client should USE UFont instead of FontMgr after QuickDraw}
1    38 --  {$ENDC}
1    39 --      {$U libtk/UDraw        } UDraw,         {Client must USE UDraw}
1    40 --
1    41 --                                       {Client need not USE anything below this line}
1    42 --      {$U PMDecl       } PMDecl,
1    43 --  {$IFC LibraryVersion <= 20}        { P E P S I }
1    44 --      {$U PrStd        } PrStd,
1    45 --  {$ENDC}
1    46 --      {$U WM.Events    } Events,
1    47 --      {$U WM.Folders   } Folders,
1    48 --      {$U WM.Menus     } Menus,
1    49 --      {$U AlertMgr     } AlertMgr,
1    50 --  {$IFC LibraryVersion <= 20}
1    51 --      {$U PrProcs      } PrProcs,
1    52 --  {$ENDC}
1    53 --      {$U WMLstd       } WMLstd,
1    54 --      {$U WMLCrs       } WMLCrs,
1    55 --      {$U WMLSb        } WMLSb,
1    56 --      {$U WMLGrow      } WMLGrow,
1    57 --      {$U Scrap        } Scrap,
1    58 --  {$IFC LibraryVersion <= 20}
1    59 --      {$U PrMgrUtil    } PrMgrUtil,
1    60 --      {$U PrMgr        } PrMgr,
1    61 --  {$ELSEC}                           { S P R I N G }
1    62 --      {$U PrStdInfo}     PrStdInfo,
1    63 --      {$U PrPublic}      PrPublic,
1    64 --  {$ENDC}
1    65 --      {$U FilerComm    } FilerComm;
1    66 --
1    67 --
1    68 --  {$SETC fDbgABC     := fDbgOK}{FALSE}
1    69 --  {$SETC fRngABC     := fDbgOK}{FALSE}
1    70 --  {$SETC fSymABC     := fSymOK}{FALSE}
1    71 --
1    72 --  {$SETC fDebugMethods := fDbgABC} {if VAR also true, trace entries and/or exits}
1    73 --
1    74 --
1    75 --  CONST
1    76 --
1    77 --      maxMenus         = 31;   {unfortunate, but menus must be in non-relocatable storage, & this is easiest}
1    78 --      maxFonts         = 11;
1    79 --      maxSegments      = 6;
1    80 --      maxSegSize       = $20000; {128K}
1    81 --      abortChunkSize   = 32768;  {32k}
1    82 --
1    83 --      iconNameSeparator = '<';{character separating parts of an icon name}
1    84 --
1    85 --      stdHHysteresis   = 9;    {amount the mouse must move from anchor before drag starts, unless}
1    86 --      stdVHysteresis   = 6;    {           TSelection.GetHysteresis is overridden}
1    87 --
1    88 --      noCursor         = -2;   {                    used when you do not set the cursor}
1    89 --      hiddenCursor     = -1;   {icrsHidden          Hides the cursor entirely}
1    90 --      arrowCursor      =  1;   {icrsInactive        Standard arrow cursor}
1    91 --      crossCursor      =  9;   {icrsLCCross         LisaCalc cross}
1    92 --      textCursor       = 10;   {icrsXIBeam          Standard text I-Beam}
1    93 --      checkCursor      = 12;   {icrsCheck           Checkmark}
1    94 --      smCrossCursor    = 13;   {icrsGECross         LisaDraw cross (smaller than crossCursor)}
1    95 --      fingerCursor     = 14;   {icrsLFinger         LisaDraw left-pointing finger}
1    96 --
1    97 --      firstUserCursor = 100;   { this is the smallest user-defined cursor }
1    98 --
1    99 --      nothingKind      = 0;
1   100 --
1   101 --      noCmdNumber      = 0;
1   102 --
1   103 --      docLdsn          = 3;        {ldsn for the first document data segment}
1   104 --      docDsBytes       = 5120;     {default heap size for a document data segment}
1   105 --      docExcess        = 2048;     {the virtual data segment may be this much larger than needed for the heap}
1   106 --
1   107 --      printLdsn        = 2;        {ldsn to hand to LisaPrint}
1   108 --      ascArwDown       = $1F;
1   109 --      ascArwLeft       = $1C;
1   110 --      ascArwRight      = $1D;
```

```
1  111 --       ascArrUp        = $1E;
1  112 --       ascBackspace    = $08;
1  113 --       ascClear        = $1B;
1  114 --       ascEnter        = $03;
1  115 --       ascReturn       = $0D;
1  116 --       ascTab          = $09;
1  117 --
1  118 --     {alert phrase codes must be between 9 and 899}
1  119 --
1  120 --       phWordDelimiters= 9;
1  121 --
1  122 --       phTrouble       = 10;   {The tool is having trouble}
1  123 --       phUnknown       = 11;   {Phrase(error) is undefined for this error}
1  124 --       phNoText        = 21;
1  125 --       phNoSel         = 22;
1  126 --       phNoInsPt       = 23;
1  127 --       phRevert        = 24;
1  128 --       phRevBlank      = 25;
1  129 --       phUnkCmd        = 26;
1  130 --       phSelCant       = 27;
1  131 --       phUnchanged     = 28;
1  132 --       phSaving        = 29;
1  133 --       phTerminated    = 30;
1  134 --       phEditClip      = 31;
1  135 --       phNoClip        = 32;
1  136 --       phUnkClip       = 33;
1  137 --       phDialogUp      = 34;
1  138 --       phCantUndo      = 35;
1  139 --       phNoCommand     = 36;
1  140 --       phOlderVersion  = 37;
1  141 --       phNewerVersion  = 38;
1  142 --       phConverting    = 39;
1  143 --       phAborting      = 40;
1  144 --
1  145 --       phPage          = 41;  {+SW+}
1  146 --       phTitle         = 42;  {+SW+}
1  147 --
1  148 --       phCantSave      = 43;
1  149 --       phCantRevert    = 44;
1  150 --
1  151 --       phCountry       = 45;
1  152 --
1  153 --
1  154 --     {command, selection, and phrase indices used by Dialog Building Block}
1  155 --       uCreateLayoutBox = 701;     {Command numbers}
1  156 --       uMoveLayoutBoxes  = 702;
1  157 --       uCmdLaunchHeading = 703;
1  158 --       uCmdInstallMargins = 704;
1  159 --
1  160 --       layPickKind = 119;          {Selection kinds}
1  161 --       layEditLegendKind = 133;
1  162 --       frameKind = 161;
1  163 --
1  164 --       phTooManyChars = 101;       {Phrases}
1  165 --       phOddEven = 102;
1  166 --       phOddOnly = 103;
1  167 --       phEvenOnly  = 104;
1  168 --       phOddOrEven = 105;
1  169 --       phMinPage = 106;
1  170 --       phMaxPage = 107;
1  171 --       phPageAlignment = 108;
1  172 --       phAlignment = 109;
1  173 --       phTopLeft = 110;
1  174 --       phTopCenter = 111;
1  175 --       phTopRight = 112;
1  176 --       phBotLeft = 113;
1  177 --       phBotCenter = 114;
1  178 --       phBotRight = 115;
1  179 --       phLaunchHeading = 116;
1  180 --       phPageMargins = 117;
1  181 --       phUnits = 118;
1  182 --       phInches = 119;
1  183 --       phCentimeters = 120;
1  184 --       phLeft = 121;
1  185 --       phLeftCluster = 122;
1  186 --       phTop = 123;
1  187 --       phTopCluster = 124;
1  188 --       phRight = 125;
1  189 --       phRightCluster = 126;
1  190 --       phBottom = 127;
1  191 --       phBotCluster = 128;
1  192 --       phInstallMargins = 129;
1  193 --       phInchTitle= 130;
1  194 --       phCmTitle = 131;
1  195 --       phNewHeading = 132;  {+SW+}
1  196 --
1  197 --       phOK = 142;         {client should NOT lightly change the phrases for these, since they are used for}
1  198 --       phCancel = 143;     {determining text and locations of OK/Cancel buttons for built-in ToolKit dialogs}
1  199 --
1  200 --       stdBoxWidth  = 17;  {dimensions for default checkboxes}
1  201 --       stdBoxHeight = 11;
1  202 --       stdBoxSpacing = 20;
1  203 --
1  204 --       stdCurvH  = 18;   {for Buttons}
1  205 --       stdCurvV  = 14;
1  206 --       stdBtnHeight = 22;
1  207 --
1  208 --       noIDNumber = -2;
1  209 --       noId = '';
1  210 --
1  211 --       IDLength = 9; {the significant length of id strings}
1  212 --
1  213 --       stdTitleHeight = 10;      {for layout boxes}
1  214 --       stdSlimTitleHeight = 6;
1  215 --       stdLeftRightBorder = 3;
1  216 --       stdBottomBorder = 2;
1  217 --
1  218 --     {errcodes of other libraries}
1  219 --
1  220 --       erAborted       = 4033; {user typed Apple-.; Desktop Manager}
```

```
1  221 --       erDuplicateName =  890; {OS & Desktop Manager}
1  222 --       erInvalidName   =  971; {OS & Desktop Manager}
1  223 --       erNameNotFound  =  972; {OS & Desktop Manager}
1  224 --
1  225 --     {ToolKit errCodes must be between 4201 and 4499}
1  226 --
1  227 --       erPassword      = 4201;
1  228 --       erVersion       = 4202;
1  229 --       erBadData       = 4203;
1  230 --       erCantRead      = 4304;
1  231 --       erCantWrite     = 4305;
1  232 --       erDirtyDoc      = 4306;
1  233 --       erNoMoreDocs    = 4307;
1  234 --       erNoMemory      = 4308;
1  235 --       erNoDiskSpace   = 4309;
1  236 --       erWrongPassword = 4310;
1  237 --       erMaxToolKit    = 4499;
1  238 --
1  239 --     {command codes must be between 101 and 999}
1  240 --
1  241 --       uSetAllAside    = 101;
1  242 --       uSetAside       = 102;
1  243 --       uPutAway        = 103;
1  244 --       uPrFmt          = 104;
1  245 --       uPrintAsIs      = 111;
1  246 --       uPrint          = 105;
1  247 --       uPrMonitor      = 106;
1  248 --       uSaveVersion    = 107;
1  249 --       uRevertVersion  = 108;
1  250 --       utSetAside      = 109;  {Set Aside ¶Document¶}
1  251 --       uSetClipAside   = 110;
1  252 --
1  253 --       {Typing Buzzword}
1  254 --       uTyping         = 150;
1  255 --
1  256 --       {The toolkit uses the following only as arguments to selection.CantDoCmd}
1  257 --       uBackspace      = 151;
1  258 --       uEnter          = 152;
1  259 --       uForwardSpace   = 153;
1  260 --       uReturn         = 154;
1  261 --       uTab            = 155;
1  262 --
1  263 --       {The toolkit uses the following only as arguments to process.RememberCommand}
1  264 --       uSomeCommand    = 156;
1  265 --       uScrolling      = 157;
1  266 --       uSplitting      = 158;
1  267 --       uResizeWindow   = 159;
1  268 --       uResizePanel    = 160;
1  269 --       UMousePress     = 161;
1  270 --       uThumbing       = 162;
1  271 --       uMoveWindow     = 163;
1  272 --       uKeyDown        = 164; {could be made the same as uTyping}
1  273 --
1  274 --       uCopy           = 201;
1  275 --       uCut            = 202;
1  276 --       uPaste          = 203;
1  277 --       uSelAll         = 204;
1  278 --       uUndoLast       = 205;
1  279 --       utUndoLast      = 206;  {Undo ¶Last Change¶}
1  280 --       utRedoLast      = 207;  {Redo ¶Last Change¶}
1  281 --       uClear          = 208;
1  282 --
1  283 --     {$IFC LibraryVersion <= 20}
1  284 --       uFnt0           = 300;
1  285 --       uFnt1           = 301;
1  286 --       uFnt2           = 302;
1  287 --       uFnt3           = 303;
1  288 --       uFnt4           = 304;
1  289 --       uFnt5           = 305;
1  290 --       uFnt6           = 306;
1  291 --       uFnt7           = 307;
1  292 --       uFnt8           = 308;
1  293 --       uFnt9           = 309;
1  294 --       uFnt10          = 310;
1  295 --       uFnt11          = 311;
1  296 --     {$ENDC}
1  297 --       uModern         = 320 + famModern - famMin;     {should result in 320}
1  298 --       uClassic        = 320 + famClassic - famMin;
1  299 --
1  300 --       u20Pitch        = 330 + size20Pitch - sizeMin;  {should result in 330}
1  301 --       u15Pitch        = 330 + size15Pitch - sizeMin;
1  302 --       u12Pitch        = 330 + size12Pitch - sizeMin;
1  303 --       u10Pitch        = 330 + size10Pitch - sizeMin;
1  304 --       u12Point        = 330 + size12Point - sizeMin;
1  305 --       u14Point        = 330 + size14Point - sizeMin;
1  306 --       u18Point        = 330 + size18Point - sizeMin;
1  307 --       u24Point        = 330 + size24Point - sizeMin;
1  308 --
1  309 --       uPlain          = 351;
1  310 --       uBold           = 352;
1  311 --       uItalic         = 353;
1  312 --       uUnderline      = 354;
1  313 --       uShadow         = 355;
1  314 --       uOutline        = 356;
1  315 --       uSuperscript    = 357;
1  316 --       uSubscript      = 358;
1  317 --
1  318 --       uPrvuMargins    = 401;
1  319 --       uPrvuBreaks     = 402;
1  320 --       uPrvuOff        = 403;
1  321 --       uDesignPages    = 405;
1  322 --
1  323 --       uShouFullSize   = 406;
1  324 --       uReduce70Pct    = 407;
1  325 --       uReduceToFit    = 408;
1  326 --
1  327 --       uSetHorzBreak   = 411;
1  328 --       uSetVertBreak   = 412;
1  329 --       uClearBreaks    = 413;
1  330 --
```

```
1  331 --       uRiseVertically   = 421;
1  332 --       uRiseHorizontally = 422;
1  333 --
1  334 --       uAddColumnStrip   = 431;
1  335 --       uAddRouStrip      = 432;
1  336 --
1  337 --       uReportEvents     = 501;
1  338 --
1  339 --       uCountHeap        = 506;
1  340 --
1  341 --       uCheckIndices     = 509;
1  342 --       uDumpGlobals      = 510;
1  343 --       uDumpPrelude      = 511;
1  344 --       uExperimenting    = 512;
1  345 --       uReptGarbage      = 513;
1  346 --       uFreeGarbage      = 514;
1  347 --
1  348 --       uMainScramble     = 515;
1  349 --       uDocScramble      = 516;
1  350 --
1  351 --       uEditDialog       = 521;
1  352 --       uStopEditDialog   = 522;
1  353 --
1  354 --     { the standard WantMenu will return FALSE for any menus with menuID >= mBuzzword;
1  355 --       buzzword menus should be assigned IDs >= 100;
1  356 --       debug menus should be assigned IDs 90-99 }
1  357 --
1  358 --       {$IFC fDbgABC}
1  359 --       mBuzzword         = 100;
1  360 --       {$ELSEC}
1  361 --       mBuzzword         =  90;
1  362 --       {$ENDC}
1  363 --
1  364 --       mnuClipFilePrint = 1000;     {special menuID for Clipboard File/Print}
1  365 --
1  366 --       firstPrivateEvent = 100; {first event type that you can use in TProcess.SendEvent}
1  367 --
1  368 --       {$IFC NOT fDbgABC}
1  369 --       fExperimenting =   FALSE;  { not experimenting if debug code if off }
1  370 --       {$ENDC}
1  371 --
1  372 --   TYPE
1  373 --
1  374 --       TPrinterMetrics = RECORD
1  375 --           paperRect:      Rect;      {the physical rectangle}
1  376 --           printRect:      Rect;      {the printable rectangle}
1  377 --           res:            Point;     {resolution, spots/inch}
1  378 --           reserve:        ARRAY[0..7] OF BYTE;
1  379 --           END;
1  380 --
1  381 --       TPrevieuMode = (mPrvuMargins, mPrvuBreaks, mPrvuOff);
1  382 --
1  383 --       TDiResponse = (diAccept, diDismissDialogBox, diGiveToMainWindou, diRefuse);
1  384 --
1  385 --       TEnumAbilities = (aBar, aScroll, aSplit);
1  386 --       TAbilities = SET OF TEnumAbilities;                    {for TPanel.Divide/CREATE argument}
1  387 --
1  388 --       TUnitsFromEdge = (pixelsFromEdge, percentFromEdge);    {for TPanel.Divide argument}
1  389 --
1  390 --       TAlertArg = 1..5;
1  391 --       TAlertCounter = 7..9;
1  392 --
1  393 --       TAlignment = (aLeft, aRight, aCenter, aJustify);
1  394 --       TPageAlignment = (aTopLeft, aTopCenter, aTopRight, aBottomLeft, aBottomCenter, aBottomRight);
1  395 --
1  396 --       TClickState =  RECORD
1  397 --                      where: Point;
1  398 --                      when: LONGINT;
1  399 --                      clickCount: INTEGER;
1  400 --                      fShift, fOption, fApple: BOOLEAN;
1  401 --                      END;
1  402 --
1  403 --       TCmdNumber = INTEGER;        {the unique identifier of a command in a menu (or elseuhere)}
1  404 --
1  405 --       TCmdPhase = (doPhase, undoPhase, redoPhase);{doPhase first time, then undoPhase & redoPhase alternately}
1  406 --
1  407 --       TCursorNumber = INTEGER;
1  408 --
1  409 --       TEnumIcons = (iSkeuer, iScrollBack, iFlipBack, iGrayA, iThumb, iGrayB, iFlipFud, iScrollFud); {TIcon}
1  410 --
1  411 --       TMousePhase = (mPress, mMove, mRelease);
1  412 --
1  413 --       TRevelation = (revealNone, revealSome, revealAll);
1  414 --
1  415 --       TPrReserve = ARRAY [0..127] OF Byte;        {lengthened}
1  416 --       TPrelude =
1  417 --           RECORD
1  418 --               password:       {2} INTEGER;
1  419 --               version:        {2} INTEGER;        {*** Should also do ABC version protection***}
1  420 --               country:        {2} INTEGER;
1  421 --               language:       {2} INTEGER;
1  422 --               preludeSize:    {2} INTEGER;        {SIZEOF(TPrelude), uhich precedes the heap}
1  423 --               unused:         {6} ARRAY [0..5] OF Byte;
1  424 --              {The above fields should occupy 16 bytes to meet the Lisa standard}
1  425 --               printPref:    {128} TPrReserve;
1  426 --               docSize:        {4} LONGINT;        {sum of the sizes of the consecutive data segments}
1  427 --               numSegments:    {2} INTEGER;        {no. of segments; all but the last are maxSegSize bytes}
1  428 --               docDirectory:   {4} TDocDirectory;  {uhence one finds the class table and the uindou}
1  429 --              {Other fields may be added later}
1  430 --               END;
1  431 --
1  432 --       TPPrelude = ^TPrelude;
1  433 --
1  434 --       TSBoxID = LONGINT; {THSb alias}
1  435 --
1  436 --       TWindouID = LONGINT; {WindouPtr alias}
1  437 --
1  438 --       TWmgrCmd =
1  439 --           RECORD
1  440 --               cmdNumber:   INTEGER;               {the command number}
```

```
1  441 --              menuIndex:   Byte;                  {the ordinal number of the menu in its menu bar (or file)}
1  442 --              itemIndex:   Byte;                  {the ordinal number of the item in its menu}
1  443 --          END;
1  444 --
1  445 --
1  446 --      TProcess = SUBCLASS OF TObject {only one instance exists (process)}
1  447 --
1  448 --        {Variables}
1  449 --
1  450 --        {Creation/Destruction}
1  451 --          FUNCTION  {TProcess.}CREATE(object: TObject; heap: THeap): TProcess;
1  452 --
1  453 --        {Debugging}
1  454 --          {$IFC fDebugMethods}
1  455 --          PROCEDURE {TProcess.}DontDebug;     {Turn off all debug flags when last document is closed}
1  456 --          {$ENDC}
1  457 --          {$IFC fDbgABC}
1  458 --          PROCEDURE {TProcess.}DumpGlobals;   {Print most global variables on alternate screen}
1  459 --          {$ENDC}
1  460 --
1  461 --
1  462 --        {Cursor Tracking}
1  463 --          PROCEDURE {TProcess.}ChangeCursor(cursorNumber: TCursorNumber);
1  464 --                    { applications call ChangeCursor if they want to change the cursor shape }
1  465 --          PROCEDURE {TProcess.}DoCursorChange(cursorNumber: TCursorNumber);
1  466 --                    { applications implement DoCursorChange to test cursorNumber for one of their
1  467 --                        cursor shapes; if found, it calls QuickDraw's SetCursor routine, otherwise
1  468 --                        it calls the generic TProcess.DoCursorChange }
1  469 --          PROCEDURE {TProcess.}TrackCursor;
1  470 --
1  471 --        {Error Reporting}
1  472 --          PROCEDURE {TProcess.}ArgAlert(whichArg: TAlertArg; argText: S255); {whichArg = 1 to 5}
1  473 --          FUNCTION  {TProcess.}Ask(phraseNumber: INTEGER): INTEGER;
1  474 --          PROCEDURE {TProcess.}BeginWait(phraseNumber: INTEGER);
1  475 --          FUNCTION  {TProcess.}Caution(phraseNumber: INTEGER): BOOLEAN;
1  476 --          PROCEDURE {TProcess.}CountAlert(whichCtr: TAlertCounter; counter: INTEGER);
1  477 --          PROCEDURE {TProcess.}DrawAlert(phraseNumber: INTEGER; marginLRect: LRect);
1  478 --          PROCEDURE {TProcess.}EndWait;
1  479 --          PROCEDURE {TProcess.}GetAlert(phraseNumber: INTEGER; VAR theText: S255);
1  480 --          PROCEDURE {TProcess.}Note(phraseNumber: INTEGER);
1  481 --          PROCEDURE {TProcess.}RememberCommand(cmdNumber: TCmdNumber);     { for ⌘C and ⌘K in alerts }
1  482 --          FUNCTION  {TProcess.}Phrase(error: INTEGER): INTEGER;
1  483 --          PROCEDURE {TProcess.}Stop(phraseNumber: INTEGER);
1  484 --
1  485 --        {Initiate/Terminate}
1  486 --          PROCEDURE {TProcess.}Commence(phraseVersion: INTEGER); {process init after the process object exists}
1  487 --          PROCEDURE {TProcess.}Complete(allIsWell: BOOLEAN);
1  488 --
1  489 --        {Abort Handling}
1  490 --          FUNCTION  {TProcess.}AbortRequest: BOOLEAN;
1  491 --          PROCEDURE {TProcess.}AbortXferSequential(whichWay: xReadWrite; pFirst: Ptr;
1  492 --                                                   numBytes, chunkSize: LONGINT; fs: TFileScanner);
1  493 --
1  494 --        {Main Loop}
1  495 --          PROCEDURE {TProcess.}ObeyEvents(FUNCTION StopCondition: BOOLEAN);
1  496 --                    {This will return IF: (1) amDying is TRUE (application terminated)
1  497 --                        or (2) StopCondition returns TRUE (StopCondition is checked
1  498 --                        only when no events are available, before starting to idle.)}
1  499 --
1  500 --          PROCEDURE {TProcess.}ObeyFilerEvent;
1  501 --          PROCEDURE {TProcess.}ObeyTheEvent;
1  502 --          PROCEDURE {TProcess.}Run;
1  503 --
1  504 --        {Private Events (Inter-process communication)}
1  505 --          PROCEDURE {TProcess.}HandlePrivateEvent(typeOfEvent: INTEGER; fromProcess: LONGINT;
1  506 --                                                  when: LONGINT; otherData: LONGINT); DEFAULT;
1  507 --          PROCEDURE {TProcess.}SendEvent(typeOfEvent: INTEGER; targetProcess: LONGINT; otherData: LONGINT);
1  508 --
1  509 --        {Memory Management}
1  510 --          PROCEDURE {TProcess.}BindCurrentDocument;
1  511 --
1  512 --        {Open/Close Window/Document}
1  513 --          FUNCTION  {TProcess.}NewDocManager(volumePrefix: TFilePath; openAsTool: BOOLEAN)
1  514 --                                          : TDocManager; DEFAULT;
1  515 --
1  516 --        {External Document Support}
1  517 --          PROCEDURE {TProcess.}CopyExternalDoc(VAR error: INTEGER;
1  518 --                                externalName, volumePrefix: TFilePath); DEFAULT;
1  519 --              {This is called if the application puts icons into the clipboard and the user
1  520 --                  then pastes them into a folder or disk.}
1  521 --
1  522 --          END;
1  523 --
1  524 --
1  525 --      TDocDirectory = SUBCLASS OF TObject
1  526 --
1  527 --        {Variables}
1  528 --          window:          TWindow;
1  529 --          classWorld:      TClassWorld;
1  530 --
1  531 --        {Creation/Destruction}
1  532 --          FUNCTION  {TDocDirectory.}CREATE(object: TObject; heap: THeap; itsWindow: TWindow;
1  533 --                                           itsClassWorld: TClassWorld): TDocDirectory;
1  534 --
1  535 --        {Version Conversion}
1  536 --          PROCEDURE {TDocDirectory.}Adopt;
1  537 --
1  538 --          END;
1  539 --
1  540 --
1  541 --      TDocManager = SUBCLASS OF TObject
1  542 --
1  543 --        {Variables}
1  544 --          files:
1  545 --              RECORD
1  546 --              volumePrefix:    TFilePath;          {Desktop Manager volume and prefix of OS files}
1  547 --              volume:          TFilePath;          {Desktop Manager volume of OS files; -volname-}
1  548 --  {$IFC LibraryVersion > 20}
1  549 --              password:        TPassword;          {The password for this document}
1  550 --  {$ENDC}
```

```
1  551 --              saveExists:      BOOLEAN;                {whether Save file is known to exist and seem readable}
1  552 --              shouldSuspend:   BOOLEAN;                {should we create suspend files?}
1  553 --              shouldToolSave:  BOOLEAN;                {should we create save files if opened as a tool?}
1  554 --              END;
1  555 --            dataSegment:
1  556 --              RECORD
1  557 --                refnum:        ARRAY [1..maxSegments] OF INTEGER;    {refnums of its data segments}
1  558 --                preludePtr:    TPPrelude;              {a pointer to the prelude of the data segment}
1  559 --                changes:       LONGINT;                {How many changes since the last checkpoint}
1  560 --              END;
1  561 --            docHeap:           THeap;                  {the heap starts after the prelude}
1  562 --            window:            TWindow;                {the document's window (it is in the data segment)}
1  563 --            pendingNote:       INTEGER;                {If <> 0, NOTE alert that was requested while inactive}
1  564 --            openedAsTool:      BOOLEAN;
1  565 --
1  566 --          {Creation/Destruction}
1  567 --            FUNCTION  {TDocManager.}CREATE(object: TObject; heap: THeap; itsPathPrefix: TFilePath): TDocManager;
1  568 --
1  569 --          {Debugging}
1  570 --            {$IFC fDbgABC}
1  571 --            PROCEDURE {TDocManager.}DumpPrelude;        {Print most of prelude on alternate screen}
1  572 --            {$ENDC}
1  573 --
1  574 --          {Attributes}
1  575 --            FUNCTION  {TDocManager.}WindowWithId(wmgrID: TWindowID): TWindow;
1  576 --
1  577 --          {Process Termination}
1  578 --            PROCEDURE {TDocManager.}Complete(allIsWell: BOOLEAN);
1  579 --
1  580 --          {Open/Close Window}
1  581 --            FUNCTION  {TDocManager.}NewWindow(heap: THeap; wmgrID: TWindowID): TWindow; DEFAULT;
1  582 --
1  583 --          {Files}
1  584 --            PROCEDURE {TDocManager.}Close(afterSuspend: BOOLEAN);
1  585 --              { CloseFiles is for the application to override if it has any of its own files that must be
1  586 --                closed }
1  587 --            PROCEDURE {TDocManager.}CloseFiles;
1  588 --            PROCEDURE {TDocManager.}Open(VAR error: INTEGER; wmgrID: TWindowID; VAR OpenedSuspended: Boolean);
1  589 --            PROCEDURE {TDocManager.}OpenBlank(VAR error: INTEGER; wmgrID: TWindowID);
1  590 --            PROCEDURE {TDocManager.}OpenSaved(VAR error: INTEGER; wmgrID: TWindowID);
1  591 --            PROCEDURE {TDocManager.}OpenSuspended(VAR error: INTEGER; wmgrID: TWindowID);
1  592 --            PROCEDURE {TDocManager.}RevertVersion(VAR error: INTEGER; wmgrID: TWindowID);
1  593 --            PROCEDURE {TDocManager.}SaveVersion(VAR error: INTEGER; volumePrefix: TFilePath;
1  594 --                                                andContinue: BOOLEAN);
1  595 --            PROCEDURE {TDocManager.}Suspend(VAR error: INTEGER);
1  596 --
1  597 --          {Data Segment}
1  598 --            PROCEDURE {TDocManager.}Assimilate(VAR error: INTEGER);
1  599 --            PROCEDURE {TDocManager.}Bind; DEFAULT;
1  600 --            PROCEDURE {TDocManager.}ConserveMemory(maxExcess: LONGINT; fGC: BOOLEAN);
1  601 --                                    {if fGC is TRUE also do a garbage collect -- on debugging versions,
1  602 --                                      we just report garbage, on non-debugging versions we free it
1  603 --                                      also.}
1  604 --            PROCEDURE {TDocManager.}Deactivate;
1  605 --            FUNCTION  {TDocManager.}DfltHeapSize: LONGINT;
1  606 --            PROCEDURE {TDocManager.}ExpandMemory(bytesNeeded: LONGINT);
1  607 --            PROCEDURE {TDocManager.}KillSegments(first, last: INTEGER);
1  608 --            PROCEDURE {TDocManager.}MakeSegments(VAR error: INTEGER; oldSegments: INTEGER; newDocSize: LONGINT);
1  609 --            PROCEDURE {TDocManager.}ResumeAfterOpen(VAR error: INTEGER; wmgrID: TWindowID);
1  610 --            PROCEDURE {TDocManager.}SetSegSize(VAR error: INTEGER; minSize, maxExcess: LONGINT);
1  611 --            PROCEDURE {TDocManager.}Unbind; DEFAULT;
1  612 --            END;
1  613 --
1  614 --        TClipboard = SUBCLASS OF TDocManager
1  615 --
1  616 --          {Variables}
1  617 --            hasView:             BOOLEAN;          {FALSE if no tool-kit-specific representation available}
1  618 --            hasPicture:          BOOLEAN;          {FALSE if no universal picture available}
1  619 --            hasUniversalText:    BOOLEAN;          {FALSE if no universal text available}
1  620 --            hasIcon:             BOOLEAN;          {TRUE if there is an icon reference available}
1  621 --          {****NOTE: The only way into or out of Universal Text is via the Universal Text Building Block****}
1  622 --            cuttingTool:         LONGINT;          {The tool number of the tool that loaded the Clipboard, or 0}
1  623 --            cuttingProcessID:    LONGINT;          {The OS process ID of the tool that loaded the Clipboard, or 0}
1  624 --            clipCopy:            TFileScanner;     {IF <> NIL a scanner on the file containing a copy of the
1  625 --                                                     clipboard before conversion.}
1  626 --
1  627 --          {Creation/Destruction}
1  628 --            FUNCTION  {TClipboard.}CREATE(object: TObject; heap: THeap): TClipboard;
1  629 --
1  630 --          {Editing}
1  631 --            PROCEDURE {TClipboard.}AboutToCut;      {whether or not data will actually be put in the data seg}
1  632 --            PROCEDURE {TClipboard.}BeginCut;
1  633 --            PROCEDURE {TClipboard.}EndCut;
1  634 --
1  635 --          {Undo}
1  636 --            PROCEDURE {TClipboard.}CommitCut;
1  637 --            FUNCTION  {TClipboard.}UndoCut: BOOLEAN; {return TRUE if succeeds}
1  638 --
1  639 --          {Identification}
1  640 --            PROCEDURE {TClipboard.}Inspect;
1  641 --            PROCEDURE {TClipboard.}Publicize;
1  642 --
1  643 --          {Data Segment}
1  644 --            {PROCEDURE  TClipboard.  Bind;}
1  645 --            {PROCEDURE  TClipboard.  Unbind;}
1  646 --
1  647 --            END;
1  648 --
1  649 --
1  650 --        TCommand = SUBCLASS OF TObject
1  651 --
1  652 --          {Variables}
1  653 --            cmdNumber:           TCmdNumber;       {the command number of the menu item that describes the command;
1  654 --                                                     usually the same one the user chose, but not necessarily}
1  655 --            image:               TImage;           {If non-NIL, affects filtering by image.EachVirtualPart}
1  656 --            undoable:            BOOLEAN;          {TRUE iff this command is undoable}
1  657 --            doing:               BOOLEAN;          {TRUE if Performing or just did doPhase or redoPhase}
1  658 --            revelation:          TRevelation;      {revealNone/Some/All of selection before performing command}
1  659 --            unHiliteBefore:      ARRAY [TCmdPhase] OF BOOLEAN; {TRUE -> Toolkit unhilites all selections before
1  660 --                                                                perform}
```

```
1  661 --          hiliteAfter:     ARRAY [TCmdPhase] OF BOOLEAN; {TRUE -> Toolkit hilites all selections after perform}
1  662 --
1  663 --        {Creation/Destruction}
1  664 --          FUNCTION  {TCommand.}CREATE(object: TObject; heap: THeap; itsCmdNumber: TCmdNumber;
1  665 --                                      itsImage: TImage; isUndoable: BOOLEAN; itsRevelation: TRevelation): TCommand;
1  666 --
1  667 --        {Filtering}
1  668 --          PROCEDURE {TCommand.}EachVirtualPart(PROCEDURE DoToObject(filteredObj: TObject));
1  669 --          PROCEDURE {TCommand.}FilterAndDo(actualObj: TObject; PROCEDURE DoToObject(filteredObj: TObject));
1  670 --
1  671 --        {Command Execution}
1  672 --          PROCEDURE {TCommand.}Commit; DEFAULT;                              {commit a command}
1  673 --          PROCEDURE {TCommand.}Perform(cmdPhase: TCmdPhase); DEFAULT;  {do, undo, or redo a command}
1  674 --
1  675 --          END;
1  676 --
1  677 --
1  678 --      TCutCopyCommand = SUBCLASS OF TCommand
1  679 --
1  680 --        {Variables}
1  681 --          isCut: BOOLEAN;          {TRUE iff this was a cut; FALSE iff a copy}
1  682 --
1  683 --        {Creation/Destruction}
1  684 --          FUNCTION  {TCutCopyCommand.}CREATE(object: TObject; heap: THeap; itsCmdNumber: TCmdNumber;
1  685 --                                             itsImage: TImage; isCutCmd: BOOLEAN): TCutCopyCommand;
1  686 --
1  687 --        {Command Execution}
1  688 --          {PROCEDURE  TCutCopyCommand.  Commit;}
1  689 --          PROCEDURE {TCutCopyCommand.}DoCutCopy(clipSelection: TSelection; deleteOriginal: BOOLEAN;
1  690 --                                                cmdPhase: TCmdPhase); DEFAULT;
1  691 --                          {the clipboard is already set up; you only have to load data into it in doPhase}
1  692 --          {PROCEDURE  TCutCopyCommand.  Perform(cmdPhase: TCmdPhase);}
1  693 --
1  694 --          END;
1  695 --
1  696 --
1  697 --      TPasteCommand = SUBCLASS OF TCommand
1  698 --
1  699 --        {Creation/Destruction}
1  700 --          FUNCTION  {TPasteCommand.}CREATE(object: TObject; heap: THeap; itsCmdNumber: TCmdNumber;
1  701 --                                           itsImage: TImage): TPasteCommand;
1  702 --
1  703 --        {Command Execution}
1  704 --          PROCEDURE {TPasteCommand.}DoPaste(clipSelection: TSelection; pic: PicHandle;
1  705 --                                            cmdPhase: TCmdPhase); DEFAULT;
1  706 --                          {the clipboard is already set up, except in undoPhase sel & pic are NIL}
1  707 --          {PROCEDURE  TPasteCommand.  Perform(cmdPhase: TCmdPhase);}
1  708 --
1  709 --          END;
1  710 --
1  711 --
1  712 --      TImage = SUBCLASS OF TObject
1  713 --
1  714 --        {Variables}
1  715 --          extentLRect:          LRect;          {the bounding box for updates; also for default hit-testing}
1  716 --          view:                 TView;
1  717 --          allowMouseOutside:    BOOLEAN;        {If TRUE, TImage.MouseTrack will NOT force the mouse point
1  718 --                                                    to lie within the extentLRect; TImage.CREATE sets this FALSE}
1  719 --
1  720 --      {methods}
1  721 --          FUNCTION  {TImage.}CREATE(object: TObject; heap: THeap; itsExtent: LRect; itsView: TView): TImage;
1  722 --
1  723 --          FUNCTION  {TImage.}CursorAt(mouseLPt: lPoint): TCursorNumber; DEFAULT;
1  724 --          PROCEDURE {TImage.}Draw; DEFAULT;
1  725 --          PROCEDURE {TImage.}EachActualPart(PROCEDURE DoToObject(filteredObj: TObject)); DEFAULT;
1  726 --          PROCEDURE {TImage.}EachVirtualPart(PROCEDURE DoToObject(filteredObj: TObject)); DEFAULT;
1  727 --          PROCEDURE {TImage.}FilterAndDo(actualObj: TObject; PROCEDURE DoToObject(filteredObj: TObject));
1  728 --          PROCEDURE {TImage.}HaveView(view: TView); DEFAULT;
1  729 --          FUNCTION  {TImage.}Hit(mouseLPt: lPoint): BOOLEAN; DEFAULT;
1  730 --          PROCEDURE {TImage.}Invalidate;  {does NOT do it on all pads}
1  731 --          FUNCTION  {TImage.}LaunchLayoutBox(view: TView): TImage; DEFAULT;
1  732 --          PROCEDURE {TImage.}OffSetBy(deltaLPt: LPoint); DEFAULT;
1  733 --          PROCEDURE {TImage.}OffSetTo(newTopLeft: LPoint);
1  734 --          PROCEDURE {TImage.}MouseMove(mouseLPt: lPoint); DEFAULT;
1  735 --          PROCEDURE {TImage.}MousePress(mouseLPt: lPoint); DEFAULT;
1  736 --          PROCEDURE {TImage.}MouseRelease; DEFAULT;
1  737 --          PROCEDURE {TImage.}MouseTrack(mPhase: THousePhase; mouseLPt: LPoint); DEFAULT;
1  738 --          PROCEDURE {TImage.}ReactToPrinterChange; DEFAULT;
1  739 --          PROCEDURE {TImage.}RecalcExtent; DEFAULT;
1  740 --          PROCEDURE {TImage.}Resize(newExtent: LRect); DEFAULT;
1  741 --          FUNCTION  {TImage.}SeesSameAs(image: TImage): BOOLEAN; DEFAULT; {$}
1  742 --
1  743 --          END;
1  744 --
1  745 --
1  746 --      TView = SUBCLASS OF TImage
1  747 --
1  748 --        {Variables}
1  749 --          panel:                TPanel;                    {The panel in which it is viewed}
1  750 --          clickLPt:             LPoint;                    {The last place the user clicked the mouse button}
1  751 --          printManager:         TPrintManager;             {NIL if view not printable}
1  752 --          res:                  Point;                     {resolution, spots/inch}
1  753 --
1  754 --          screenPad:            TPad;       {like noPad, but scales from view coords to screen coords if view
1  755 --                                             resolution and screen resolution differ
1  756 --                                             *** CAUTION -- Only for mapping coordinates-- DO NOT try to
1  757 --                                             Focus this pad or do Invals, etc ***}
1  758 --
1  759 --          fitPagesPerfectly:    BOOLEAN;  {whether view size should fluctuate automatically so that one always
1  760 --                                            ends up with an even number of pages}
1  761 --
1  762 --          isPrintable:          BOOLEAN;                   {Whether this view can be printed}
1  763 --          isMainView:           BOOLEAN;     {FALSE if an auxiliary view, such as page view or paginated view}
1  764 --          stdScroll:            LPoint;
1  765 --          scrollPastEnd:        Point;                {Amount we should scroll past the end of the view}
1  766 --
1  767 --
1  768 --        {Creation/Destruction}
1  769 --          FUNCTION  {TView.}CREATE(object: TObject; heap: THeap; itsPanel: TPanel; itsExtent: LRect;
1  770 --              itsPrintManager: TPrintManager; itsDfltMargins: LRect; itsFitPagesPerfectly: BOOLEAN;
```

```
1  771 --              itsRes: Point; isMainView: BOOLEAN): TView;
1  772 --           {PROCEDURE  TView.  Free;}
1  773 --
1  774 --         {Attributes}
1  775 --           PROCEDURE {TView.}BeInPanel(panel: TPanel);
1  776 --           PROCEDURE {TView.}GetStdScroll(VAR deltaLStd: LPoint);
1  777 --           FUNCTION  {TView.}MaxPageToPrint: LONGINT;
1  778 --
1  779 --         {Pagination}
1  780 --           PROCEDURE {TView.}AddStripOfPages(vhs: VHSelect); DEFAULT;
1  781 --           FUNCTION  {TView.}ForceBreakAt(vhs: VHSelect; precedingLocation: LONGINT;
1  782 --               proposedLocation: LONGINT): LONGINT;
1  783 --           PROCEDURE {TView.}RedoBreaks; DEFAULT;
1  784 --           PROCEDURE {TView.}RemapManualBreaks(
1  785 --                FUNCTION NewBreakLocation(vhs: VHSelect; oldBreak: LONGINT): LONGINT);
1  786 --
1  787 --         {Cross-Panel Drag}
1  788 --           FUNCTION  {TView.}DoReceive(selection: TSelection; lPtInView: LPoint): BOOLEAN;
1  789 --
1  790 --         {Direct Display Permission -- per panel}
1  791 --           FUNCTION  {TView.}OKToDrawIn(lRectInView: LRect): BOOLEAN;  {Default is FALSE; app can override}
1  792 --
1  793 --         {Cursor tracking - per pane}
1  794 --           {FUNCTION  TView.  CursorAt(mouseLPt: LPoint): TCursorNumber;}
1  795 --
1  796 --         {Resizing}
1  797 --           {PROCEDURE  TView.  Resize(newExtent: LRect);}
1  798 --           PROCEDURE {TView.}SetMinViewSize(VAR minLRect: LRect);
1  799 --
1  800 --         {Clipboard Setup}
1  801 --           PROCEDURE {TView.}CreateUniversalText;
1  802 --
1  803 --         {Variables embedded in text}
1  804 --           PROCEDURE {TView.}SetFunctionValue(keyword: S255; VAR itsValue: S255);
1  805 --
1  806 --         {Selecting}
1  807 --           FUNCTION  {TView.}NoSelection: TSelection;
1  808 --           END;
1  809 --
1  810 --
1  811 --       TPaginatedView = SUBCLASS OF TView
1  812 --
1  813 --         {Variables}
1  814 --           unpaginatedView:     TView;              {the unpaginated view from whence this derives}
1  815 --
1  816 --           pageSize:    ARRAY[VHSelect] OF LONGINT;  {width/height of a page's representation on the screen,
1  817 --                                                      in the same metrics as the regular view -- could still
1  818 --                                                      differ from actual screen space a/c screen horiz/vertical
1  819 --                                                      resolution}
1  820 --
1  821 --           workingInMargins:    BOOLEAN;
1  822 --
1  823 --         {Creation/Destruction}
1  824 --           FUNCTION  {TPaginatedView.}CREATE(object: TObject; heap: THeap;
1  825 --                                             itsUnpaginatedView: TView): TPaginatedView;
1  826 --
1  827 --          {PROCEDURE  TPaginatedView.  AddStripOfPages(vhs: VHSelect);}
1  828 --           PROCEDURE {TPaginatedView.}AdornPageOnScreen;
1  829 --          {FUNCTION   TPaginatedView.  CursorAt(mouseLPt: LPoint): TCursorNumber;}
1  830 --           PROCEDURE {TPaginatedView.}DepagifyLPoint(pagLPt: LPoint; VAR unPagLPt: LPoint);
1  831 --           PROCEDURE {TPaginatedView.}DoOnPages(focusOnInterior: BOOLEAN;  PROCEDURE DoOnAPage);
1  832 --          {PROCEDURE  TPaginatedView.  Draw;}
1  833 --          {PROCEDURE  TPaginatedView.  MouseTrack(mPhase: TPhase; mouseLPt: LPoint);}
1  834 --           PROCEDURE {TPaginatedView.}PagifyLPoint(unPagLPt: LPoint; VAR pagLPt: LPoint);
1  835 --          {PROCEDURE  TPaginatedView.  ReactToPrinterChange;}
1  836 --          {PROCEDURE  TPaginatedView.  RedoBreaks;}
1  837 --
1  838 --         END;
1  839 --
1  840 --
1  841 --       TPageView = SUBCLASS OF TView
1  842 --
1  843 --           FUNCTION  {TPageView.}CREATE(object: TObject; heap: THeap;
1  844 --                                        itsPrintManager: TPrintManager): TPageView;
1  845 --          {PROCEDURE  TPageView.  Draw;}
1  846 --           END;
1  847 --
1  848 --
1  849 --       THeading = SUBCLASS OF TImage      {a header/footer image}
1  850 --
1  851 --           printManager:            TPrintManager;
1  852 --           pageAlignment:           TPageAlignment;
1  853 --           offsetFromAlignment:     LPoint;
1  854 --
1  855 --           oddOnly:         BOOLEAN;   {to restrict printing only to odd-numbered pages}
1  856 --           evenOnly:        BOOLEAN;   { ditto even }
1  857 --           minPage:         LONGINT;   {minimum page number to want this heading}
1  858 --           maxPage:         LONGINT;   {maximum page number to want it}
1  859 --
1  860 --         {Creation/Destruction}
1  861 --           FUNCTION  {THeading.}CREATE(object: TObject; heap: THeap; itsPrintManager: TPrintManager;
1  862 --                                       itsExtentLRect: LRect; itsPageAlignment: TPageAlignment;
1  863 --                                       itsOffsetFromAlignment: LPoint): THeading;
1  864 --
1  865 --         {Attributes}
1  866 --           PROCEDURE {THeading.}ChangePageAlignment(newPageAlignment: TPageAlignment);
1  867 --
1  868 --         {Selective Use}
1  869 --           FUNCTION  {THeading.}ShouldDraw(pageNumber: LONGINT): BOOLEAN;
1  870 --           FUNCTION  {THeading.}ShouldFrame: BOOLEAN; DEFAULT;
1  871 --
1  872 --         {Display}
1  873 --           PROCEDURE {THeading.}AdjustForPage(pageNumber: LONGINT; editing: BOOLEAN); DEFAULT;
1  874 --           PROCEDURE {THeading.}LocateOnPage(editing: BOOLEAN);
1  875 --          {PROCEDURE  THeading.  Draw;}
1  876 --           END;
1  877 --
1  878 --
1  879 --       TPrintManager = SUBCLASS OF TObject
1  880 --           view:                TView;
```

```
1  881 --          pageView:           TView;
1  882 --
1  883 --          breaks:             ARRAY[VHSelect] OF TArray; {of LONGINT}
1  884 --                                {pagebreak representation: absolute value gives location; negative
1  885 --                                signifies manual break; nonnegative signifies automatic pagebreak}
1  886 --
1  887 --          pageMargins:        LRect;      {in view resolution; top and left are > 0 , bot & right < 0}
1  888 --
1  889 --          headings:           TList; {OF THeading}
1  890 --
1  891 --          canEditPages:       BOOLEAN;
1  892 --          layoutDialogBox:    TDialogBox;
1  893 --
1  894 --          frameBody:          BOOLEAN;
1  895 --          paperLRect:         LRect;
1  896 --          printableLRect:     LRect;
1  897 --
1  898 --          contentLRect:       LRect;  {the inner rectangle into which chunks of view are stuffed}
1  899 --
1  900 --          printerMetrics:     TPrinterMetrics;    {physical properties of the printer}
1  901 --
1  902 --          pageRiseDirection:  VHSelect;
1  903 --               {if 'h', it means that page numbers rise from left to right fastest;
1  904 --                if 'v', it means that page numbers rise from top to bottom fastest;
1  905 --                default value is 'h'}
1  906 --
1  907 --          FUNCTION  {TPrintManager.}CREATE(object: TObject; heap: THeap): TPrintManager;
1  908 --          PROCEDURE {TPrintManager.}Init(itsMainView: TView; itsDfltMargins: LRect);
1  909 --        {PROCEDURE  TPrintmanager. Free;}
1  910 --
1  911 --          PROCEDURE {TPrintManager.}AddStripOfPages(vhs: VHSelect);
1  912 --          PROCEDURE {TPrintManager.}ChangeMargins(margins: LRect);
1  913 --          PROCEDURE {TPrintManager.}ClearPageBreaks(automatic: BOOLEAN);
1  914 --          PROCEDURE {TPrintManager.}DrawBreaks(manualOnly:  BOOLEAN);
1  915 --          PROCEDURE {TPrintManager.}DrawOneBreak(pageBreak: LONGINT; vhs: vhSelect);
1  916 --          PROCEDURE {TPrintManager.}DrawPage;
1  917 --          PROCEDURE {TPrintManager.}EnterPageEditing;
1  918 --          PROCEDURE {TPrintManager.}GetPageLimits(pageNumber: LONGINT; VAR viewLRect: LRect);
1  919 --          FUNCTION  {TPrintManager.}NewPaginatedView(object: TObject): TPaginatedView;
1  920 --          FUNCTION  {TPrintManager.}NewPageView(object: TObject): TView;
1  921 --          FUNCTION  {TPrintManager.}PageWith(VAR lPtInView: LPoint; VAR strip: Point): LONGINT;
1  922 --          PROCEDURE {TPrintManager.}Print(printPref: TPrReserve);
1  923 --          PROCEDURE {TPrintManager.}ReactToPrinterChange;
1  924 --          PROCEDURE {TPrintManager.}RedoBreaks;
1  925 --          PROCEDURE {TPrintManager.}SetBreak(vhs: VHSelect; where: LONGINT; isAutomatic: BOOLEAN);
1  926 --          PROCEDURE {TPrintManager.}SetDfltHeadings; DEFAULT;
1  927 --          PROCEDURE {TPrintManager.}SkipPage(pageNumber: LONGINT);
1  928 --
1  929 --      END; {TPrintManager definition}
1  930 --
1  931 --
1  932 --      TSelection = SUBCLASS OF TObject
1  933 --
1  934 --        {Variables}
1  935 --          window:          TWindow;           {the window in which it was made}
1  936 --          panel:           TPanel;            {the panel in which it was made}
1  937 --          view:            TView;             {the view or subview of panel in which it was made}
1  938 --          kind:            INTEGER;           {0 means no selection, rest of codes are defined by view}
1  939 --          anchorLPt:       LPoint;            {the place the mouse went down (view-relative)}
1  940 --          currLPt:         LPoint;            {the place the mouse was last tracked}
1  941 --          boundLRect:      LRect;             {bounding box of the selection} {+++LSR+++}
1  942 --          coSelection:     TSelection;        {if non-NIL, a selection to forward unimplemented methods to}
1  943 --          canCrossPanels:  BOOLEAN;           {:=TRUE in MousePress/FALSE in MouseRelease for cross-panel drag}
1  944 --
1  945 --        {Creation/Destruction}
1  946 --          FUNCTION  {TSelection.}CREATE(object: TObject; heap: THeap; itsView: TView; itsKind: INTEGER;
1  947 --                                        itsAnchorLPt: LPoint): TSelection;
1  948 --         {FUNCTION   TSelection. Clone(heap: THeap): TObject;}    {clones coSelection}
1  949 --          FUNCTION  {TSelection.}FreedAndReplacedBy(selection: TSelection): TSelection;
1  950 --
1  951 --        {Attributes}
1  952 --          PROCEDURE {TSelection.}GetHysteresis(VAR hysterPt: Point); DEFAULT; {rtns a delta from orig panel pt}
1  953 --          PROCEDURE {TSelection.}HaveView(view: TView);
1  954 --
1  955 --        {Files}
1  956 --          PROCEDURE {TSelection.}MarkChanged; DEFAULT;                        {Increment change counters}
1  957 --
1  958 --        {Command Dispatch}
1  959 --          FUNCTION  {TSelection.}CanDoCommand(cmdNumber: TCmdNumber; VAR checkIt: BOOLEAN): BOOLEAN; DEFAULT;
1  960 --          PROCEDURE {TSelection.}CantDoCmd(cmdNumber: TCmdNumber); DEFAULT;
1  961 --          PROCEDURE {TSelection.}CantDoIt; DEFAULT;
1  962 --          PROCEDURE {TSelection.}DoKey(ascii: CHAR; keycap: Byte; shiftKey, appleKey, optionKey: BOOLEAN);
1  963 --          FUNCTION  {TSelection.}NewCommand(cmdNumber: TCmdNumber): TCommand; DEFAULT;
1  964 --          PROCEDURE {TSelection.}PerformCommand(command: TCommand; cmdPhase: TCmdPhase); DEFAULT;
1  965 --
1  966 --        {Idle}
1  967 --          PROCEDURE {TSelection.}IdleBegin(centiSeconds: LONGINT); DEFAULT;
1  968 --          PROCEDURE {TSelection.}IdleContinue(centiSeconds: LONGINT); DEFAULT;
1  969 --          PROCEDURE {TSelection.}IdleEnd(centiSeconds: LONGINT); DEFAULT;
1  970 --
1  971 --        {Editing -- to be overridden by applications}
1  972 --          PROCEDURE {TSelection.}KeyBack(fWord: BOOLEAN); DEFAULT;
1  973 --          PROCEDURE {TSelection.}KeyChar(ch: CHAR); DEFAULT;
1  974 --          PROCEDURE {TSelection.}KeyClear; DEFAULT;
1  975 --          PROCEDURE {TSelection.}KeyEnter(dh, dv: INTEGER); DEFAULT;
1  976 --          PROCEDURE {TSelection.}KeyForward(fWord: BOOLEAN); DEFAULT;
1  977 --          PROCEDURE {TSelection.}KeyPause; DEFAULT; {Pause in typing}
1  978 --          PROCEDURE {TSelection.}KeyReturn; DEFAULT;
1  979 --          PROCEDURE {TSelection.}KeyTab(fBackward: BOOLEAN); DEFAULT;
1  980 --          PROCEDURE {TSelection.}SelectParagraphs;
1  981 --
1  982 --        {Drawing -- per pane}
1  983 --          PROCEDURE {TSelection.}Highlight(highTransit: THighTransit); DEFAULT;
1  984 --
1  985 --        {Selecting}
1  986 --          PROCEDURE {TSelection.}DeSelect; DEFAULT;
1  987 --          PROCEDURE {TSelection.}DrawGhost; DEFAULT;
1  988 --          PROCEDURE {TSelection.}MousePress(mouseLPt: LPoint); DEFAULT;
1  989 --          PROCEDURE {TSelection.}MouseMove(mouseLPt: LPoint); DEFAULT;
1  990 --          PROCEDURE {TSelection.}MouseRelease; DEFAULT;
```

```
1  991 --          PROCEDURE {TSelection.}MoveBackToAnchor; DEFAULT;   {called when cross-panel drag has been refused}
1  992 --
1  993 --        {Undo Maintenance}
1  994 --          PROCEDURE {TSelection.}Restore; DEFAULT;
1  995 --          PROCEDURE {TSelection.}Save; DEFAULT;
1  996 --
1  997 --        {Scroll into view}
1  998 --          PROCEDURE {TSelection.}Reveal(asMuchAsPossible: BOOLEAN); DEFAULT;
1  999 --
1 1000 --          END;
1 1001 --
1 1002 --
1 1003 --      TWindow = SUBCLASS OF TArea
1 1004 --
1 1005 --        {Variables}
1 1006 --          panels:          TList {OF TPanel};          {The panels in the window (at least one)}
1 1007 --          panelTree:       TArea;                      {no panels: NIL, one panel: that; else a TBranchArea}
1 1008 --          dialogBox:       TDialogBox;                 {NIL if SELF IS a dialog box window}
1 1009 --          selectPanel:     TPanel;                     {The panel with the active selection}
1 1010 --          undoSelPanel:    TPanel;                     {The selectPanel during the last command}
1 1011 --          clickPanel:      TPanel;                     {The panel in which the user last clicked in a pane}
1 1012 --          undoClickPanel:  TPanel;                     {The clickPanel during the last command}
1 1013 --          selectWindow:    TWindow;                    {The window with the active selection -- either
1 1014 --                                                         SELF or its Dialogbox }
1 1015 --          undoSelWindow:   TWindow;                    {the selectWindow during the last command}
1 1016 --          wmgrID:          TWindowID;                  {ORD(Pointer to the Window Manager's GrafPort)}
1 1017 --          isResizable:     BOOLEAN;                    {Is there a Resize Box}
1 1018 --          believeWmgr:     BOOLEAN;                    {TRUE iff the Toolkit should believe the window
1 1019 --                                                           manager's idea of the size of the window;
1 1020 --                                                           this will be FALSE (for example) if we create
1 1021 --                                                           the window object before the window is put on
1 1022 --                                                           the screen.}
1 1023 --          maxInnerSize:    Point;                      {The window size the user explicitly set with grow
1 1024 --                                                           icon}
1 1025 --          changes:         LONGINT;                    {How many changes since the last save}
1 1026 --          lastCmd:         TCommand;                   {last undoable command object}
1 1027 --          printerMetrics:  TPrinterMetrics;            {Properties of the printer currently formatted for}
1 1028 --          pgSzOK:          BOOLEAN;                    {Whether to allow user-defined page-sizes in Fmt For
1 1029 --                                                           Printer dialog}
1 1030 --          pgRgOK:          BOOLEAN;                    {Whether page-range dialog should be enabled in PRINT...
1 1031 --                                                           dialog -- normally TRUE}
1 1032 --          panelToPrint:    TPanel;                     {NB: IF >1 printable panel in window, choice should be
1 1033 --                                                           made by providing separate menu items}
1 1034 --          objectToFree:    TObject;                    {used to stash a reference to an object which should be
1 1035 --                                                           freed at end of event loop}
1 1036 --        {Creation/Destruction}
1 1037 --          FUNCTION  {TWindow.}CREATE(object: TObject; heap: THeap; itsWmgrID: TWindowID; itsResizability
1 1038 --                                  : BOOLEAN): TWindow;
1 1039 --         {PROCEDURE  TWindow.  Free;}
1 1040 --
1 1041 --        {$IFC fDbgABC}
1 1042 --        {Debugging}
1 1043 --          PROCEDURE {TWindow.}ToggleFlag(VAR flag: BOOLEAN); DEFAULT; {Toggle a debug flag in a menu}
1 1044 --         {$ENDC}
1 1045 --
1 1046 --        {Attributes}
1 1047 --         {PROCEDURE  TWindow.  GetMinExtent(VAR minExtent: Point; windowIsResizingIt: BOOLEAN);}
1 1048 --          PROCEDURE {TWindow.}GetTitle(VAR title: S255); {Get the window title}
1 1049 --          FUNCTION  {TWindow.}IsActive: BOOLEAN;
1 1050 --          FUNCTION  {TWindow.}IsVisible: BOOLEAN;
1 1051 --          PROCEDURE {TWindow.}SetWmgrId(itsWmgrId: TWindowID); {Also sets port fields of panes}
1 1052 --
1 1053 --        {Buttoning}
1 1054 --          PROCEDURE {TWindow.}DownEventAt(mousePt: Point); DEFAULT;
1 1055 --         {FUNCTION   TWindow.  DownAt(mousePt: Point): BOOLEAN;}
1 1056 --
1 1057 --        {Dialog Box affairs}
1 1058 --          PROCEDURE {TWindow.}PutUpDialogBox(dialogBox: TDialogBox); DEFAULT;
1 1059 --          PROCEDURE {TWindow.}TakeDownDialogBox; DEFAULT;
1 1060 --
1 1061 --        {Display}
1 1062 --         {PROCEDURE  TWindow.  Focus;}
1 1063 --         {PROCEDURE  TWindow.  Frame;}
1 1064 --          PROCEDURE {TWindow.}Highlight(highTransit: THighTransit); DEFAULT;
1 1065 --         {PROCEDURE  TWindow.  Refresh(rActions: TActions; highTransit: THighTransit);}
1 1066 --          PROCEDURE {TWindow.}Update(doHilite: BOOLEAN); DEFAULT;
1 1067 --
1 1068 --        {Resizing}
1 1069 --          PROCEDURE {TWindow.}DownInSizeBox(mousePt: Point); DEFAULT;
1 1070 --          PROCEDURE {TWindow.}Resize(moving: BOOLEAN); DEFAULT; {Reset size from portRect size (w. adjustments)}
1 1071 --          PROCEDURE {TWindow.}ResizeTo(newSize: Point); DEFAULT; {callable from application}
1 1072 --
1 1073 --        {Command Dispatch and Menus}
1 1074 --          FUNCTION  {TWindow.}CanDoCommand(cmdNumber: TCmdNumber; VAR checkIt: BOOLEAN): BOOLEAN; DEFAULT;
1 1075 --          FUNCTION  {TWindow.}CanDoStdCommand(cmdNumber: TCmdNumber; VAR checkIt: BOOLEAN): BOOLEAN; DEFAULT;
1 1076 --          PROCEDURE {TWindow.}CommitLast; DEFAULT;
1 1077 --          PROCEDURE {TWindow.}DoCommand(cmdNumber: TCmdNumber); DEFAULT;
1 1078 --          PROCEDURE {TWindow.}LoadMenuBar; DEFAULT;
1 1079 --          PROCEDURE {TWindow.}MenuEventAt(mousePt: Point); DEFAULT;
1 1080 --          FUNCTION  {TWindow.}NewCommand(cmdNumber: TCmdNumber): TCommand; DEFAULT;
1 1081 --          FUNCTION  {TWindow.}NewStdCommand(cmdNumber: TCmdNumber): TCommand;
1 1082 --          PROCEDURE {TWindow.}PerformCommand(newCommand: TCommand);
1 1083 --          PROCEDURE {TWindow.}PerformLast(cmdPhase: TCmdPhase);
1 1084 --          PROCEDURE {TWindow.}SaveCommand(command: TCommand); {NOTE: do not use the arg after calling this;
1 1085 --                                                                use window.lastCmd instead}
1 1086 --          PROCEDURE {TWindow.}SetupMenus;
1 1087 --          PROCEDURE {TWindow.}UndoLast;
1 1088 --          FUNCTION  {TWindow.}WantMenu(menuID: INTEGER; inClipboard: BOOLEAN): BOOLEAN;
1 1089 --
1 1090 --        {Miscellaneous}
1 1091 --          PROCEDURE {TWindow.}AbortEvent; {only QuickPort should override this}
1 1092 --
1 1093 --        {Selection Maintenance during commands}
1 1094 --          PROCEDURE {TWindow.}RestoreSelection;
1 1095 --          PROCEDURE {TWindow.}RevealSelection(asMuchAsPossible, doHilite: BOOLEAN);
1 1096 --          PROCEDURE {TWindow.}SaveSelection;
1 1097 --
1 1098 --        {Desktop}
1 1099 --             {The following 2 methods assume that we are focused on the window before they are called}
1 1100 --          PROCEDURE {TWindow.}Activate;
```

```
1 1101 --           PROCEDURE {TWindow.}Deactivate;
1 1102 --           PROCEDURE {TWindow.}BlankStationery; DEFAULT;
1 1103 --           PROCEDURE {TWindow.}StashPicture(highTransit: THighTransit);
1 1104 --
1 1105 --     {$IFC LibraryVersion > 20}
1 1106 --         {Desktop Manager Communication}
1 1107 --           PROCEDURE {TWindow.}NameToPrefix(VAR error, offset: INTEGER; VAR name, prefix: TFilePath);
1 1108 --           PROCEDURE {TWindow.}PrefixToName(VAR error, offset: INTEGER; VAR prefix, name: TFilePath);
1 1109 --
1 1110 --                 (*Convert between OS prefix (ie., '-volname-{DxxxTyyy}' and an icon pathname (ie.,
1 1111 --                     '<diskname<foldername1<foldername2<...<iconname').  If an error is returned,
1 1112 --                     offset will point just beyond the part of the name that caused the error, e.g.
1 1113 --                     if '<office<forms<expenses' returns erDuplicateName and the offset is 14
1 1114 --                     (pointing to the third '<') then there is more than one 'forms' folder on the
1 1115 --                     office disk.  Error constants are defined above.
1 1116 --
1 1117 --                   NOTE: these methods will likely take a while to execute, since the Desktop Manager
1 1118 --                   must be swapped in to process the request.  Therefore, you should try to minimize the
1 1119 --                   number of times these are called.*)
1 1120 --     {$ENDC}
1 1121 --
1 1122 --         {Foci of Attention}
1 1123 --           FUNCTION  {TWindow.}CursorFeedback: TCursorNumber;
1 1124 --           PROCEDURE {TWindow.}PickStdCursor;
1 1125 --
1 1126 --         {Printing}
1 1127 --           PROCEDURE {TWindow.}AcceptNewPrintingInfo(document: TDocManager; prReserve: TPrReserve);
1 1128 --           PROCEDURE {TWindow.}ChkPrMismatch;
1 1129 --           PROCEDURE {TWindow.}GetPrinterMetrics;
1 1130 --           PROCEDURE {TWindow.}Print(panel: TPanel; nixPgRange: BOOLEAN; nixWholeDialog: BOOLEAN);
1 1131 --
1 1132 --         {Filtering}
1 1133 --           PROCEDURE {TWindow.}EachActualPart(PROCEDURE DoToObject(filteredObj: TObject)); {For app to implement}
1 1134 --           PROCEDURE {TWindow.}EachVirtualPart(PROCEDURE DoToObject(filteredObj: TObject));
1 1135 --           PROCEDURE {TWindow.}FilterAndDo(actualObj: TObject; PROCEDURE DoToObject(filteredObj: TObject));
1 1136 --           PROCEDURE {TWindow.}FilterDispatch(actualObj: TObject; image: TImage;
1 1137 --                                              PROCEDURE DoToObject(filteredObj: TObject));
1 1138 --
1 1139 --         {Idle}
1 1140 --           PROCEDURE {TWindow.}IdleBegin(centiSeconds: LONGINT);
1 1141 --           PROCEDURE {TWindow.}IdleContinue(centiSeconds: LONGINT);
1 1142 --           PROCEDURE {TWindow.}IdleEnd(centiSeconds: LONGINT);
1 1143 --
1 1144 --           END;
1 1145 --
1 1146 --
1 1147 --       TDialogBox = SUBCLASS OF TWindow
1 1148 --
1 1149 --         {Variables}
1 1150 --           keyResponse:                TDiResponse;
1 1151 --           menuResponse:               TDiResponse;
1 1152 --           downInMainWindowResponse:   TDiResponse;
1 1153 --
1 1154 --           freeOnDismissal:            BOOLEAN;
1 1155 --
1 1156 --         {Creation/Destruction}
1 1157 --           FUNCTION  {TDialogBox.}CREATE(object: TObject; heap: THeap; itsResizability: BOOLEAN;
1 1158 --                                         itsHeight: INTEGER;  itsKeyResponse, itsMenuResponse,
1 1159 --                                         itsDownInMainWindowResponse: TDiResponse): TDialogBox;
1 1160 --
1 1161 --         {Attributes}
1 1162 --           {PROCEDURE  TDialogBox.  GetMinExtent(VAR minExtent: Point; windowIsResizingIt: BOOLEAN);}
1 1163 --
1 1164 --         {Display}
1 1165 --           PROCEDURE {TDialogBox.}Appear;
1 1166 --           PROCEDURE {TDialogBox.}BeDismissed;  DEFAULT;
1 1167 --           PROCEDURE {TDialogBox.}Disappear;  DEFAULT;
1 1168 --
1 1169 --           END;
1 1170 --
1 1171 --
1 1172 --       TBand = SUBCLASS OF TArea
1 1173 --
1 1174 --         {Variables}
1 1175 --           window:          TWindow;
1 1176 --           panes:           TList {OF TPane};
1 1177 --           panel:           TPanel;
1 1178 --           scroller:        TScroller;                  {the scroll box}
1 1179 --           scrollDir:       VHSelect;                   {v if a row of panes with a vertical bar,
1 1180 --                                                         h if a column of panes with a horizontal bar}
1 1181 --
1 1182 --         {Creation/Destruction}
1 1183 --           FUNCTION  {TBand.}CREATE(object: TObject; heap: THeap; itsPanel: TPanel; itsInnerRect: Rect;
1 1184 --                                    itsScroller: TScroller; itsDir: VHSelect): TBand;
1 1185 --           {PROCEDURE  TBand.  Free;}
1 1186 --
1 1187 --         {Attributes}
1 1188 --           FUNCTION  {TBand.}ViewLCd: LONGINT;
1 1189 --
1 1190 --         {Scrolling}
1 1191 --           PROCEDURE {TBand.}OffsetPanes(deltaLPt: LPoint);
1 1192 --           PROCEDURE {TBand.}ScrollBy(deltaLCd: LONGINT);
1 1193 --                     {A TBand can only scroll in one direction; this also moves the thumb}
1 1194 --           PROCEDURE {TBand.}ScrollStep(icon: TEnumIcons; deltaLStd: LONGINT);
1 1195 --           PROCEDURE {TBand.}ScrollTo(viewLCd: LONGINT);
1 1196 --           FUNCTION  {TBand.}ThumbPos: INTEGER;
1 1197 --           PROCEDURE {TBand.}ThumbTo(newThumbPos: INTEGER);
1 1198 --
1 1199 --
1 1200 --         {Resizing}
1 1201 --           {PROCEDURE  TBand.  ResizeOutside(newOuterRect: Rect);}
1 1202 --           PROCEDURE {TBand.}ResizePanes(newViewLCd: LONGINT);
1 1203 --
1 1204 --           END;
1 1205 --
1 1206 --
1 1207 --       TSideBand = SUBCLASS OF TBand
1 1208 --         {Fields}
1 1209 --           topOrLeft:  BOOLEAN;
1 1210 --           {NOTE: SELF.scroller is NIL}
```

```
1 1211 --
1 1212 --          FUNCTION  {TSideBand.}CREATE(object: TObject; heap: THeap; itsPanel: TPanel; itsInnerRect: Rect;
1 1213 --                                itsDir: VHSelect; itsTopOrLeft: BOOLEAN;
1 1214 --                                itsViewLCd: LONGINT): TSideBand;
1 1215 --
1 1216 --        {Attributes}
1 1217 --          FUNCTION  {TSideBand.}CoBand: TBand;
1 1218 --              {returns the band adjacent to SELF}
1 1219 --          END;
1 1220 --
1 1221 --
1 1222 --      TPanel = SUBCLASS OF TArea
1 1223 --
1 1224 --        {Variables}                                   {panes are listed row-wise in the panes list}
1 1225 --          window:             TWindow;
1 1226 --          panes:              TList {OF TPane};
1 1227 --          currentView:        TView;                  {The view seen through SELF: normal or paginated}
1 1228 --          view:               TView;                  {The unpaginated view seen through SELF }
1 1229 --          paginatedView:      TPaginatedView;         {NIL if not previewing margins}
1 1230 --          selection:          TSelection;             {the current selection}
1 1231 --          undoSelection:      TSelection;             {the selection to be restored for an undo/redo}
1 1232 --          bands:              ARRAY[VHSelect] OF TList;   {redundant... bands[v].at(1) = top row of panes}
1 1233 --          scrollBars:         ARRAY[VHSelect] OF TScrollBar;  {scrollBars[(v,h)]--the (vert,horiz) scroll bars}
1 1234 --          abilities:          ARRAY[VHSelect] OF TAbilities;  {[aBar, aScroll, aSplit, aResize]}
1 1235 --          minInnerDiagonal:   Point;
1 1236 --          resizeBranch:       TBranchArea;            {the branch that my botRight resizes, or NIL}
1 1237 --          zoomed:             BOOLEAN;
1 1238 --          zoomFactor:         TScaler;
1 1239 --          previewMode:        TPreviewMode;
1 1240 --          lastClick:          RECORD                  {describes the pane the user last clicked}
1 1241 --                              CASE gotPane: BOOLEAN OF
1 1242 --                                  TRUE: (clickPane: TPane);   {the last pane the user clicked on}
1 1243 --                                  FALSE: (clickPt: Point);    {the innerRect.topLeft of lastClick.pane,
1 1244 --                                                               in the case where the lastClick.pane was
1 1245 --                                                               deleted}
1 1246 --                              END;
1 1247 --          contentRect:        Rect;                   {part of the innerRect not containing side bands}
1 1248 --          tlSideBandSize:     Point;  {size of topLeft side bands}
1 1249 --          brSideBandSize:     Point;
1 1250 --              {NOTE: The sideband sizes refer to the size of the innerRect of the side band;
1 1251 --                     therefore a size of -1 means there is no side band on that side}
1 1252 --          deletedSplits:      TArray; {If NIL, don't remember splits that go away because the panel
1 1253 --                                       shrinks.  Otherwise, this should be a TArray with recordBytes
1 1254 --                                       2.  This is initialized to NIL in TPanel.CREATE; clients can
1 1255 --                                       allocate an array and change the field if they desire.}
1 1256 --
1 1257 --        {Creation/Destruction}
1 1258 --          FUNCTION  {TPanel.}CREATE(object: TObject; heap: THeap; itsWindow: TWindow;
1 1259 --                                    minHeight, minWidth: INTEGER; itsVAbilities, itsHAbilities: TAbilities): TPanel;
1 1260 --         {PROCEDURE  TPanel.  Free;}
1 1261 --          PROCEDURE {TPanel.}HaveView(view: TView);
1 1262 --          FUNCTION  {TPanel.}NewView(object: TObject; itsExtent: LRect; itsPrintManager: TPrintManager;
1 1263 --                                     itsDfltMargins: LRect; itsFitPerfectlyOnPages: BOOLEAN): TView;
1 1264 --          FUNCTION  {TPanel.}NewStatusView(object: TObject; itsExtent: LRect): TView;
1 1265 --
1 1266 --        {Attributes}
1 1267 --          PROCEDURE {TPanel.}ComputeContentRect;
1 1268 --          PROCEDURE {TPanel.}DecideAboutBars(newOuterRect: Rect); {Decide if to have scroll bars & resize icon}
1 1269 --         {PROCEDURE  TPanel.  GetMinExtent(VAR minExtent: Point; windowIsResizingIt: BOOLEAN);}
1 1270 --         {PROCEDURE  TPanel.  GetBorder(VAR border: Rect);}
1 1271 --          FUNCTION  {TPanel.}FindBranchThatIsResized: TBranchArea;
1 1272 --          FUNCTION  {TPanel.}PaneShowing(anLRect: LRect): TPane;  {Returns first pane showing an part
1 1273 --                                                                   of anLRect, else NIL}
1 1274 --          PROCEDURE {TPanel.}SetInnerRect(newInnerRect: Rect); OVERRIDE;
1 1275 --          PROCEDURE {TPanel.}SetOuterRect(newOuterRect: Rect); OVERRIDE;
1 1276 --
1 1277 --        {Paneling the window}
1 1278 --          FUNCTION  {TPanel.}Divide(vhs: VHSelect;
1 1279 --                                    fromEdgeOfPanel: INTEGER; units: TUnitsFromEdge;
1 1280 --                                    whoCanResizeIt: TResizability;
1 1281 --                                    minSize: INTEGER; itsVAbilities, itsHAbilities: TAbilities): TPanel;
1 1282 --          PROCEDURE {TPanel.}Insert(panel: TPanel; vhs: VHSelect;
1 1283 --                                    fromEdgeOfPanel: INTEGER; units: TUnitsFromEdge;
1 1284 --                                    whoCanResizeIt: TResizability);          {Resizes both to share my space}
1 1285 --          PROCEDURE {TPanel.}Remove;                {Does not Free SELF; Expands sibling to fill my space}
1 1286 --          PROCEDURE {TPanel.}Replace(panel: TPanel);  {Does not Free SELF; Resizes panel to fit my old space}
1 1287 --
1 1288 --        {Buttoning}
1 1289 --         {FUNCTION   TPanel.  DownAt(mousePt: Point): BOOLEAN;}
1 1290 --          PROCEDURE {TPanel.}DownInSizeBox(mousePt: Point);
1 1291 --          PROCEDURE {TPanel.}HitScroller(vhs: VHSelect; mousePt: Point; scroller: TScroller; icon: TEnumIcons);
1 1292 --
1 1293 --        {Selecting}
1 1294 --          PROCEDURE {TPanel.}BeginSelection;
1 1295 --          PROCEDURE {TPanel.}BeSelectPanel(inSelectWindow: BOOLEAN);
1 1296 --    (*    FUNCTION  {TPanel.}NoSelection: TSelection; *)
1 1297 --
1 1298 --        {Cursor tracking}
1 1299 --          FUNCTION  {TPanel.}CursorAt(mousePt: Point): TCursorNumber;
1 1300 --
1 1301 --        {Display}
1 1302 --         {PROCEDURE  TPanel.  Frame;}
1 1303 --          PROCEDURE {TPanel.}Highlight(selection: TSelection; highTransit: THighTransit);
1 1304 --                      { this highlights the selection on all pads }
1 1305 --          PROCEDURE {TPanel.}Invalidate;
1 1306 --                      { this invalidates the whole panel }
1 1307 --          PROCEDURE {TPanel.}InvalLRect(lRectInView: LRect);
1 1308 --                      { this invalidates the given LRect on all pads }
1 1309 --          FUNCTION  {TPanel.}OKToDrawIn(lRectInView: LRect): BOOLEAN;
1 1310 --                      { If this returns FALSE, commands must InvalLRect or XOR, not Draw or Erase }
1 1311 --          PROCEDURE {TPanel.}OnAllPadsDo(PROCEDURE DoOnThePad);
1 1312 --         {PROCEDURE  TPanel.  Refresh(rActions: TActions; highTransit: THighTransit);}
1 1313 --          PROCEDURE {TPanel.}Rescroll;
1 1314 --          PROCEDURE {TPanel.}SetZoomFactor(zoomNumerator, zoomDenominator: Point);
1 1315 --
1 1316 --        {Page-Previewing}
1 1317 --          PROCEDURE {TPanel.}Preview(newMode: TPreviewMode);
1 1318 --
1 1319 --        {Printing}
1 1320 --          PROCEDURE {TPanel.}PrintView(printPref: TPrReserve);
```

```
1 1321 --
1 1322 --          {Scrolling}
1 1323 --            PROCEDURE {TPanel.}AutoScroll(mousePt: Point);
1 1324 --            PROCEDURE {TPanel.}DoScrolling(inArea: TArea; itsPane: TPane;
1 1325 --                                hOk, vOk: BOOLEAN; VAR deltaLPt: LPoint);
1 1326 --                          {inArea must be a TBand or a TPane; if a TPane then inArea=itsPane;
1 1327 --                            if a TBand then itsPane is any one of the band's panes}
1 1328 --            FUNCTION  {TPanel.}PaneToScroll(VAR anLRect: LRect; hMinToSee, vMinToSee: INTEGER): TPane;
1 1329 --                          {Returns the pane to scroll for showing the minimum desired part ofLRect;
1 1330 --                            if that part is already showing, it returns NIL;
1 1331 --                           NOTE: anLRect is NOT changed}
1 1332 --            PROCEDURE {TPanel.}RevealLRect(VAR anLRect: LRect; hMinToSee, vMinToSee: INTEGER);
1 1333 --                           {Show at least the desired part of the LRect in the pane returned by PaneToShow;
1 1334 --                           NOTE: anLRect is NOT changed}
1 1335 --
1 1336 --          {Splitting}
1 1337 --            PROCEDURE {TPanel.}CleanUpPanes(deleteList: TList);
1 1338 --            PROCEDURE {TPanel.}MakeBand(vhs: VHSelect; scroller, prevScroller: TScroller);
1 1339 --            PROCEDURE {TPanel.}MoveSplitBefore(scroller: TScroller; newSkurCd: INTEGER);
1 1340 --            FUNCTION  {TPanel.}NewBand(heap: THeap; myInnerRect: Rect;
1 1341 --                                scroller: TScroller; vhs: VHSelect): TBand;
1 1342 --            FUNCTION  {TPanel.}NewPane(heap: THeap; innerRect: Rect; viewedLRect: LRect): TPane;
1 1343 --            PROCEDURE {TPanel.}RemakePanes;
1 1344 --            PROCEDURE {TPanel.}RememberSplit(vhs: VHSelect; atCd: INTEGER);
1 1345 --            PROCEDURE {TPanel.}RepaneOrthogonalBands(vhs: VHSelect);
1 1346 --            PROCEDURE {TPanel.}RestoreSplits;
1 1347 --
1 1348 --        {Side Bands}
1 1349 --            PROCEDURE {TPanel.}ShowSideBand(vhs: VHSelect; topOrLeft: BOOLEAN; size: INTEGER; viewLCd: LONGINT);
1 1350 --            PROCEDURE {TPanel.}SideBandRect(vhs: VHSelect; topOrLeft: BOOLEAN; VAR bandRect: Rect);
1 1351 --                    {returns the innerRect of the side band, given SELF.contentRect}
1 1352 --
1 1353 --          {Resizing}
1 1354 --            PROCEDURE {TPanel.}ResizeBand(vhs: VHSelect; band: TBand; newViewLCd: LONGINT;
1 1355 --                                fInvalidate: BOOLEAN);
1 1356 --           {PROCEDURE  TPanel. ResizeInside(newInnerRect: Rect);}
1 1357 --           {PROCEDURE  TPanel. ResizeOutside(newOuterRect: Rect);}
1 1358 --
1 1359 --            END;
1 1360 --
1 1361 --
1 1362 --        TPane = SUBCLASS OF TPad
1 1363 --
1 1364 --          {Variables}
1 1365 --            currentView:     TView;                    {The view that is currently}
1 1366 --            panel:           TPanel;                   {The containing panel}
1 1367 --
1 1368 --          {Creation/Destruction}
1 1369 --            FUNCTION  {TPane.}CREATE(object: TObject; heap: THeap; itsPanel: TPanel; itsInnerRect: Rect;
1 1370 --                                itsViewedLRect: LRect): TPane;
1 1371 --            PROCEDURE {TPane.}HaveView(view: TView);
1 1372 --
1 1373 --          {Attributes}
1 1374 --            PROCEDURE {TPane.}GetScrollLimits(VAR viewedLRect, scrollableLRect: LRect);
1 1375 --           {PROCEDURE  TPane. SetZoomFactor(zoomNumerator, zoomDenominator: Point);}
1 1376 --
1 1377 --          {Selecting}
1 1378 --            PROCEDURE {TPane.}MouseTrack(mPhase: TMousePhase; mousePt: Point);
1 1379 --                          {assumes mousePt is in the pane's innerRect}
1 1380 --
1 1381 --          {Cursor tracking}
1 1382 --            FUNCTION  {TPane.}CursorAt(mousePt: Point): TCursorNumber;
1 1383 --
1 1384 --          {Display}
1 1385 --           {PROCEDURE  TPane. Refresh(rActions: TActions; highTransit: THighTransit);}
1 1386 --
1 1387 --          {Resizing}
1 1388 --            PROCEDURE {TPane.}Resize(newInnerRect: Rect; vhs: VHSelect);
1 1389 --
1 1390 --          {Scrolling}
1 1391 --            PROCEDURE {TPane.}ScrollBy(VAR deltaLPt: LPoint);
1 1392 --                          {NOTE: deltaLPt is NOT changed; also moves the thumb(s)}
1 1393 --            PROCEDURE {TPane.}ScrollToReveal(VAR anLRect: LRect; hMinToSee, vMinToSee: INTEGER);
1 1394 --                          {NOTE: anLRect is NOT changed}
1 1395 --            END;
1 1396 --
1 1397 --
1 1398 --        TMarginPad = SUBCLASS OF TPad
1 1399 --
1 1400 --          {Variables}
1 1401 --            view:               TView;     {The view seen on the BODY of this page}
1 1402 --            pageNumber:         LONGINT;
1 1403 --            bodyPad:            TBodyPad;
1 1404 --
1 1405 --          {Creation/Destruction}
1 1406 --            FUNCTION {TMarginPad.}CREATE(object: TObject; heap: THeap): TMarginPad;
1 1407 --
1 1408 --            PROCEDURE {TMarginPad.}Rework(itsView: TView; itsOrigin: Point; itsRes: Point;
1 1409 --                          itsPageNumber: LONGINT; itsZoomFactor: TScaler; itsPort: GrafPtr);
1 1410 --            PROCEDURE {TMarginPad.}SetForPage(itsPageNumber: LONGINT; itsOrigin: Point);
1 1411 --
1 1412 --          {Display}
1 1413 --           {PROCEDURE  TMarginPad. Focus;}
1 1414 --
1 1415 --          {Process termination and Debugging Assistance}
1 1416 --           {PROCEDURE TMarginPad. Crash;}
1 1417 --           {FUNCTION  TMarginPad. BindHeap(activeVsClip, doBind: BOOLEAN): THeap;}
1 1418 --
1 1419 --            END;
1 1420 --
1 1421 --
1 1422 --        TBodyPad = SUBCLASS OF TPad
1 1423 --
1 1424 --          {Variables}
1 1425 --            marginPad:    TMarginPad; {the page shell whose body I am}
1 1426 --            nonNullBody:   Rect;  {the portion of the pad in the range of the mapped view;
1 1427 --              BodyPad.innerRect = nonNullBody unless manual pagebreak or end-of-view forces
1 1428 --              a shortage of view to map into entire inner rect} {someday make this comment comprehensible}
1 1429 --
1 1430 --          {Creation/Destruction}
```

```
1 1431 --           FUNCTION  {TBodyPad.}CREATE(object: TObject; heap: THeap; itsMarginPad: TMarginPad): TBodyPad;
1 1432 --           PROCEDURE {TBodyPad.}Recompute;
1 1433 --           PROCEDURE {TBodyPad.}SetForPage(itsPageNumber: LONGINT);
1 1434 --
1 1435 --         {Display}
1 1436 --          {PROCEDURE  TBodyPad. Focus;}
1 1437 --
1 1438 --         END;
1 1439 --
1 1440 --
1 1441 --       TScroller = SUBCLASS OF TObject
1 1442 --
1 1443 --         {Variables}
1 1444 --           scrollBar:      TScrollBar;                    {the scroll bar of which it is part}
1 1445 --           band:           TBand;                         {the object that can respond to scroll events}
1 1446 --           sBoxID:         TSBoxID;                       {the scroll-bar-library representation}
1 1447 --
1 1448 --         {Creation/Destruction}
1 1449 --           FUNCTION  {TScroller.}CREATE(object: TObject; heap: THeap; itsScrollBar: TScrollBar; itsId: TSBoxID)
1 1450 --                                 : TScroller;
1 1451 --          {PROCEDURE  TScroller. Free;}
1 1452 --
1 1453 --         {Attributes}
1 1454 --           PROCEDURE {TScroller.}GetSize(VAR boxRect: Rect);
1 1455 --           FUNCTION  {TScroller.}ScrollDir: VHSelect;
1 1456 --           PROCEDURE {TScroller.}SetSize(ownerRect: Rect);
1 1457 --           FUNCTION  {TScroller.}ThumbRange: INTEGER;
1 1458 --
1 1459 --         {Buttoning}
1 1460 --           PROCEDURE {TScroller.}TrackSkewer(mousePt: Point; VAR newSkwrCd: INTEGER;
1 1461 --                                 VAR scroller, prevScroller: TScroller);
1 1462 --           PROCEDURE {TScroller.}TrackThumb(mousePt: Point; VAR oldThumbPos, newThumbPos: INTEGER);
1 1463 --
1 1464 --         {Display}
1 1465 --           PROCEDURE {TScroller.}FillIcon(icon: TEnumIcons; fBlack: BOOLEAN);
1 1466 --           PROCEDURE {TScroller.}MoveThumb(newThumbPos: INTEGER);
1 1467 --
1 1468 --         {Splitting}
1 1469 --           PROCEDURE {TScroller.}ResplitAt(newSkwrCd: INTEGER; prevScroller: TScroller);
1 1470 --           PROCEDURE {TScroller.}SplitAt(newSkwrCd: INTEGER; VAR nextScroller: TScroller);
1 1471 --
1 1472 --           END;
1 1473 --
1 1474 --
1 1475 --       TScrollBar = SUBCLASS OF TObject
1 1476 --
1 1477 --         {Variables}
1 1478 --           firstBox:    TScroller;                        {the rest are found via the SB Library}
1 1479 --           isVisible:   BOOLEAN;                          {TRUE iff this scroll bar should be drawn}
1 1480 --
1 1481 --         {Creation/Destruction}
1 1482 --           FUNCTION  {TScrollBar.}CREATE(object: TObject; heap: THeap; vhs: VHSelect; outerRect: Rect;
1 1483 --                                 itsVisibility: BOOLEAN): TScrollBar;
1 1484 --           PROCEDURE {TScrollBar.}ChangeVisibility(needsBothBars: BOOLEAN;
1 1485 --                                               bandOuterRect: Rect; itsAbilities: TAbilities);
1 1486 --
1 1487 --         {Buttoning}
1 1488 --           FUNCTION  {TScrollBar.}DownAt(mousePt: Point; VAR scroller: TScroller; VAR icon: TEnumIcons): BOOLEAN;
1 1489 --
1 1490 --         {Display}
1 1491 --           PROCEDURE {TScrollBar.}Draw;
1 1492 --           PROCEDURE {TScrollBar.}Erase;
1 1493 --
1 1494 --           END;
1 1495 --
1 1496 --
1 1497 --       TMenuBar = SUBCLASS OF TObject {only one instance exists (menuBar)}
1 1498 --
1 1499 --         {Variables}
1 1500 --           isLoaded:       ARRAY [1..maxMenus] OF BOOLEAN;    {TRUE iff the i'th menu has been inserted}
1 1501 --           mapping:        TArray {OF TWmgrCmd};              {maps command number to menu & item indices}
1 1502 --           numMenus:       INTEGER;                           {how many menus}
1 1503 --           numCommands:    INTEGER;                           {how many commands in all menus together}
1 1504 --
1 1505 --         {Creation/Destruction}
1 1506 --           FUNCTION  {TMenuBar.}CREATE(object: TObject; heap: THeap; itsScanner: TFileScanner): TMenuBar;
1 1507 --
1 1508 --         {Attributes}
1 1509 --           PROCEDURE {TMenuBar.}Check(cmdNumber: TCmdNumber; checked: BOOLEAN);
1 1510 --           PROCEDURE {TMenuBar.}Enable(cmdNumber: TCmdNumber; canBeChosen: BOOLEAN);
1 1511 --           PROCEDURE {TMenuBar.}BuildCmdName(destCmd, templateCmd: TCmdNumber; param: TPString);
1 1512 --                   {if param is NIL, use the default}
1 1513 --           FUNCTION  {TMenuBar.}GetCmdName(cmdNumber: TCmdNumber; pName: TPString): BOOLEAN;
1 1514 --                   {returns TRUE iff cmdNumber is found (pName will be empty);
1 1515 --                       pName can be NIL, which will save the overhead of returning the
1 1516 --                       menu item, for case where you just want to see if it exists}
1 1517 --           PROCEDURE {TMenuBar.}PutCmdName(cmdNumber: TCmdNumber; pName: TPString);
1 1518 --
1 1519 --         {Buttoning}
1 1520 --           FUNCTION  {TMenuBar.}CmdKey(ch: CHAR): TCmdNumber;
1 1521 --           FUNCTION  {TMenuBar.}DownAt(mousePt: Point): TCmdNumber;
1 1522 --
1 1523 --         {Display}
1 1524 --           PROCEDURE {TMenuBar.}Draw;
1 1525 --           PROCEDURE {TMenuBar.}EndCmd;
1 1526 --           PROCEDURE {TMenuBar.}HighlightMenu(withCmd: TCmdNumber);
1 1527 --                   {call this when the user presses the CLEAR key for example, to highlight
1 1528 --                       the appropriate menu title; you should then call window.DoCommand with
1 1529 --                       an apropriate command number.}
1 1530 --
1 1531 --         {Loading}
1 1532 --           PROCEDURE {TMenuBar.}Delete(menuID: INTEGER);
1 1533 --           PROCEDURE {TMenuBar.}Insert(menuID, beforeId: INTEGER);
1 1534 --           PROCEDURE {TMenuBar.}Unload;
1 1535 --
1 1536 --         {For Future Use}
1 1537 --           FUNCTION  {TMenuBar.}MenuWithID(menuID: INTEGER): Ptr;
1 1538 --           END;
1 1539 --
1 1540 --
```

```
1 1541 --   {$IFC LibraryVersion <= 20 AND FALSE} {do it this way in case we need it back for Pepsi version}
1 1542 --       TFont = SUBCLASS OF TObject
1 1543 --
1 1544 --         {Variables}
1 1545 --           family:          INTEGER;                          {Font Manager TFam}
1 1546 --
1 1547 --         {Creation/Destruction}
1 1548 --           FUNCTION  {TFont.}CREATE(object: TObject; heap: THeap; itsFamily: INTEGER): TFont;
1 1549 --
1 1550 --           END;
1 1551 --   {$ENDC}
1 1552 --
1 1553 --       { GLOBAL VARIABLES -- EFFECTIVELY, FIELDS OF CLASS TProcess }
1 1554 --
1 1555 --   VAR
1 1556 --
1 1557 --       activeWindowID:      TWindowID;                        {The wmgrID field of the active document, or 0}
1 1558 --       allowAbort:          BOOLEAN;                          {Iff TRUE, allow aborts}
1 1559 --       autoBreakPen:        PenState;                         {pen to use to draw automatic page breaks}
1 1560 --       blinkOffCentiSecs:   LONGINT;                          {Centiseconds to hide the insertion point}
1 1561 --       blinkOnCentiSecs:    LONGINT;                          {Centiseconds to display the insertion point}
1 1562 --       boundClipboard:      TClipboard;                       {The clipboard whose data segment is bound, or NIL}
1 1563 --       boundDocument:       TDocManager;                      {The document whose data segment is bound, or NIL}
1 1564 --       cancelString:        STRING[20];                       {The word "Cancel" for use in buttons}
1 1565 --       clickState:          TClickState;                      {Shifts and repeats of the last mouse click}
1 1566 --       clipboard:           TClipboard;                       {The Clipboard document manager}
1 1567 --       clipPrintPref:       TPrReserve;                       {the print-preference for the clipboard}
1 1568 --       closedBySuspend:     BOOLEAN;                          {Iff TRUE, closedDocument was just suspended}
1 1569 --       closedDocument:      TDocManager;                      {If not NIL, this document was just put away}
1 1570 --       cornerNumberStyle:   TTypeStyle;                       {TypeStyle used for page-numbers in page-preview}
1 1571 --       countryCode:         INTEGER;                          {The country code as read from phrase file}
1 1572 --       currentDocument:     TDocManager;                      {The active document OR if running in background, the
1 1573 --                                                                  document to use; otherwise NIL}
1 1574 --       currentWindow:       TWindow;                          {currentDocument.window, OR NIL}
1 1575 --       cursorShape:         TCursorNumber;                    {The cursor shape as recorded by TProcess.ChangeCursor}
1 1576 --       deferUpdate:         BOOLEAN;                          {set TRUE by app to defer updating while typing}
1 1577 --       dfltNewHeading:      STRING[20]; {+SW+}                {Default value for newly-created headings}
1 1578 --       docList:             TList {OF TDocManager};           {The documents that are open}
1 1579 --       eventTime:           LONGINT;                          {The time of the most recent WM event}
1 1580 --       eventType:           INTEGER;                          {The type number of the most recent WM event}
1 1581 --   {$IFC fDbgABC}
1 1582 --       fExperimenting:      BOOLEAN;                          {IF TRUE, enable zoom experimentation etc.}
1 1583 --       fCountHeap:          BOOLEAN;                          {Iff TRUE and IFC fCheckHeap, count objects once per cmd}
1 1584 --   {$ENDC}
1 1585 --   {$IFC LibraryVersion <= 20 AND FALSE} {do it this way in case we need it back for the Pepsi version}
1 1586 --       fonts:               ARRAY [0..maxFonts] OF TFont;
1 1587 --   {$ENDC}
1 1588 --       genClipPic:          BOOLEAN;                          {Iff TRUE, we are generating the Clipboard picture}
1 1589 --       highLevel:           ARRAY [BOOLEAN] OF THighTransit;     {TRUE=>hOffToOn, FALSE=>hOffToDim}
1 1590 --       highToggle:          ARRAY [BOOLEAN] OF THighTransit;     {TRUE=>hOffToOn, FALSE=>hOnToOff}
1 1591 --       idleTime:            LONGINT;                          {The time we finished processing the last user input}
1 1592 --       inBackground:        BOOLEAN;                          {Iff TRUE, currently running in background}
1 1593 --       limboPen:            PenState;                         {pen to use to fill limbo area in paginated view}
1 1594 --       manualBreakPen:      PenState;                         {pen to use to draw manual page breaks}
1 1595 --       marginPattern:       LPattern;                         {pattern to use to fill margins in paginated view}
1 1596 --       menuBar:             TMenuBar;                         {The menus of the application and the Clipboard}
1 1597 --       myProcessID:         LONGINT;                          {The OS ID of this process}
1 1598 --       myTool:              LONGINT;                          {The tool number of this tool}
1 1599 --       normalPen:           PenState;                         {pen state resulting from PenNormal}
1 1600 --       okString:            STRING[20];                       {The word "OK" for use in buttons}
1 1601 --       phraseFile:          TFileScanner;                     {The Main Phrase File TFileScanner}
1 1602 --       process:             TProcess;                         {The process object of this process}
1 1603 --
1 1604 --       screenRightEdge:     INTEGER;                          {720 for Lisa 1.0 screen}
1 1605 --       scrollRgn:           RgnHandle;                        {what needs to be refreshed because of scroll}
1 1606 --       stdMargins:          LRect;                            {standard page-margins, in screen pixels}
1 1607 --       suspendSuffix:       ARRAY [1..maxSegments] OF STRING[3];
1 1608 --       theBodyPad:          TBodyPad;                         {current BodyPad being written to}
1 1609 --       theMarginPad:        TMarginPad;                       {current MarginPad being written to}
1 1610 --       toolName:            STRING[67];                       {The name of the tool}
1 1611 --       toolPrefix:          TFilePath;                        {*The prefix '{Tnn}' of the OS path name of the tool*}
1 1612 --       toolVolume:          TFilePath;                        {The volume '-name-' on which the tool resides}
1 1613 --       varPage:             STRING[20]; {+SW+}                {The string 'PAGE', for use in heading variables}
1 1614 --       varTitle:            STRING[20]; {+SW+}                {The string 'TITLE' for use in heading variables}
1 1615 --       wordDelimiters:      STRING[67];                       {The delimiters of a Lisa "word" in this language}
1 1616 --
1 1617 --
1 1618 --   PROCEDURE GetPrefixPart(wholeName: S255; VAR filePart: TFilePath); (*'{prefix}'*)
1 1619 --   FUNCTION  ToolOfFile(wholeName: S255): LONGINT;
1 1620 --   FUNCTION  ToolOfProcess(processId: LONGINT): LONGINT;
1 1621 --
1 1622 --   { Used to insert comments into the Universal Graph of Clipboard, so LisaDraw can understand it;
1 1623 --       These procedures only insert comment when we are generating the Universal Graph }
1 1624 --                                  { beginning of a series of text drawing ops that should be grouped }
1 1625 --   PROCEDURE PicTextBegin(alignment: TAlignment);
1 1626 --   PROCEDURE PicTextEnd;          { end of series }
1 1627 --   PROCEDURE PicGrpBegin;         { beginning of a series of grouped objects }
1 1628 --   PROCEDURE PicGrpEnd;           { end of series }
1 1629 --
1 1630 --   PROCEDURE InitProcess;
1 1631 --
1 1632 --   FUNCTION GetTime: LONGINT;
1 1633 --       {This function returns the same "time" as is used in events (see global variable eventTime),
1 1634 --               and in the idle loop}
1 1635 --
1 1636 --
1 1637 --   IMPLEMENTATION
1 1638 --
2    1 --
2    2 --
1 1639 --   {$I LIBTK/UABC2.TEXT}      {TProcess-TDocDirectory-TDocManager-TClipboard-TCommand-TCutCopyCommand-
1 1640 --                               TPasteCommand}
3    1 --
3    2 --
1 1641 --   {$I LIBTK/UABC3.TEXT}      {TImage-TView-TPaginatedView-TPageView-TPrintManager-THeading-TSelection}
4    1 --
4    2 --
1 1642 --   {$I LIBTK/UABC4.TEXT}      {TWindow-TDialogBox-TMenuBar-TFont}
5    1 --
5    2 --
```

```
1 1643 --   {$I LIBTK/UABC5.TEXT}     {TPanel-TBand-TPane-TMarginPad-TBodyPad-TScroller-TScrollBar}
1 1644 --
1 1645 --   (**********
1 1646 --   {$I UABC2.TEXT}     {TProcess-TDocDirectory-TDocManager-TClipboard-TCommand-TCutCopyCommand-TPasteCommand}
1 1647 --   {$I UABC3.TEXT}     {TImage-TView-TPaginatedView-TPageView-TPrintManager-THeading-TSelection}
1 1648 --   {$I UABC4.TEXT}     {TWindow-TDialogBox-TMenuBar-TFont}
1 1649 --   {$I UABC5.TEXT}     {TPanel-TBand-TPane-TMarginPad-TBodyPad-TScroller-TScrollBar}
1 1650 --   **********)
1 1651 --
1 1652 --   END.
1 1653 --
```

1. libtk/uabc.TEXT
2. LIBTK/UABC2.TEXT
3. LIBTK/UABC3.TEXT
4. LIBTK/UABC4.TEXT
5. LIBTK/UABC5.TEXT

-A-
aBar 385•(1)
abilities 1234•(1)
abortChunkSize 81•(1)
AbortEvent 1091•(1)
AbortRequest 490•(1)
AbortXferSequent 491•(1)
aBottomCenter 394•(1)
aBottomLeft 394•(1)
aBottomRight 394•(1)
AboutToCut 631•(1)
AcceptNewPrintIn 1127•(1)
aCenter 393•(1)
Activate 1100•(1)
activeWindowID 1557•(1)
AddStripOfPages 780•(1) 911•(1)
AdjustForPage 873•(1)
Adopt 536•(1)
AdornPageOnScree 828•(1)
aJustify 393•(1)
aLeft 393•(1)
AlertMgr 49•(1)
allowAbort 1558•(1)
allowMouseOutsid 717•(1)
anchorLPt 939•(1)
Appear 1165•(1)
ArgAlert 472•(1)
aRight 393•(1)
arrowCursor 90•(1)
ascArrowDown 108•(1)
ascArrowLeft 109•(1)
ascArrowRight 110•(1)
ascArrowUp 111•(1)
ascBackspace 112•(1)
ascClear 113•(1)
ascEnter 114•(1)
ascReturn 115•(1)
aScroll 385•(1)
ascTab 116•(1)
Ask 473•(1)
aSplit 385•(1)
Assimilate 598•(1)
aTopCenter 394•(1)
aTopLeft 394•(1)
aTopRight 394•(1)
autoBreakPen 1559•(1)
AutoScroll 1323•(1)

-B-
band 1445•(1)
bands 1232•(1)
BeDismissed 1166•(1)
BeginCut 632•(1)
BeginSelection 1294•(1)
BeginWait 474•(1)
BeInPanel 775•(1)
believeWMgr 1018•(1)
BeSelectPanel 1295•(1)
Bind 599•(1)
BindCurrentDocum 510•(1)
BlankStationery 1102•(1)
blinkOffCentiSec 1560•(1)
blinkOnCentiSecs 1561•(1)
bodyPad 1403•(1)
boundClipboard 1562•(1)
boundDocument 1563•(1)
boundLRect 941•(1)
breaks 883•(1)
brSideBandSize 1249•(1)
BuildCmdName 1511•(1)
BYTE 378 (1)
Byte 415 (1) 423 (1) 441 (1) 442 (1)

-C-
cancelString 1564•(1)
canCrossPanels 943•(1)
CanDoCommand 959•(1) 1074•(1)
CanDoStdCommand 1075•(1)
canEditPages 891•(1)
CantDoCmd 960•(1)
CantDoIt 961•(1)
Caution 475•(1)
ChangeCursor 463•(1)
ChangeMargins 912•(1)
ChangePageAlignm 866•(1)
changes 559•(1) 1025•(1)
ChangeVisibility 1484•(1)
Check 1509•(1)
checkCursor 93•(1)
ChkPrMismatch 1128•(1)
classWorld 529•(1)
CleanUpPanes 1337•(1)
ClearPageBreaks 913•(1)
clickCount 399•(1)
clickLPt 750•(1)
clickPane 1242•(1)
clickPanel 1011•(1)
clickPt 1243•(1)
clickState 1565•(1)
clipboard 1566•(1)
clipCopy 624•(1)
clipPrintPref 1567•(1)
Close 584•(1)
closedBySuspend 1568•(1)

```
  closedDocument    1569*( 1)
  CloseFiles         587*( 1)
  CmdKey            1520*( 1)
  cmdNumber          440*( 1)  653 ( 1)
  CoBand            1217*( 1)
  Commence           486*( 1)
  Commit             672*( 1)
  CommitCut          636*( 1)
  CommitLast        1076*( 1)
  Complete           487*( 1)  578*( 1)
  ComputeContentRe  1267*( 1)
  ConserveMemory     600*( 1)
  contentLRect       898*( 1)
  contentRect       1247*( 1)
  CopyExternalDoc    517*( 1)
  cornerNumberStyl  1570*( 1)
  coSelection        942*( 1)
  CountAlert         476*( 1)
  country            420*( 1)
  countryCode       1571*( 1)
  CREATE             451 ( 1)  532*( 1)  567*( 1)  628*( 1)  664*( 1)  684*( 1)  700 ( 1)  721*( 1)  769*( 1)  824*( 1)
                     843 ( 1)  861*( 1)  907*( 1)  946*( 1) 1037*( 1) 1157*( 1) 1183*( 1) 1212*( 1) 1258*( 1) 1369*( 1)
                    1406*( 1) 1431*( 1) 1449*( 1) 1482*( 1) 1506*( 1) 1548*( 1)
  CreateUniversalT   801*( 1)
  crossCursor         91*( 1)
  currentDocument   1572*( 1)
  currentView       1227*( 1) 1365 ( 1)
  currentWindow     1574*( 1)
  currLPt            940*( 1)
  CursorAt           723*( 1) 1299*( 1) 1382*( 1)
  CursorFeedback    1123*( 1)
  cursorShape       1575*( 1)
  cuttingProcessID   623*( 1)
  cuttingTool        622*( 1)

-D-
  dataSegment        555*( 1)
  Deactivate         604*( 1) 1101*( 1)
  DecideAboutBars   1268*( 1)
  deferUpdate       1576*( 1)
  Delete            1532*( 1)
  deletedSplits     1252*( 1)
  DepagifyLPoint     830*( 1)
  DeSelect           986*( 1)
  DfltHeapSize       605*( 1)
  dfltNewHeading    1577*( 1)
  diAccept           383*( 1)
  dialogBox         1008*( 1)
  diDismissDialogB   383*( 1)
  diGiveToMainWind   383*( 1)
  diRefuse           383*( 1)
  Disappear         1167*( 1)
  Divide            1278*( 1)
  docDirectory       428*( 1)
  docDsBytes         104*( 1)
  docExcess          105*( 1)
  docHeap            561*( 1)
  docLdsn            103*( 1)
  docList           1578*( 1)
  DoCommand         1077*( 1)
  docSize            426*( 1)
  DoCursorChange     465*( 1)
  DoCutCopy          689*( 1)
  doing              657*( 1)
  DoKey              962*( 1)
  DontDebug          455*( 1)
  DoOnPages          831*( 1)
  DoPaste            704*( 1)
  doPhase            405*( 1)
  DoReceive          788*( 1)
  DoScrolling       1324*( 1)
  DownAt            1488*( 1) 1521*( 1)
  DownEventAt       1054*( 1)
  downInMainWindow  1152*( 1)
  DownInSizeBox     1069*( 1) 1290*( 1)
  Draw               724*( 1) 1491*( 1) 1524*( 1)
  DrawAlert          477*( 1)
  DrawBreaks         914*( 1)
  DrawGhost          987*( 1)
  DrawOneBreak       915*( 1)
  DrawPage           916*( 1)
  DumpGlobals        458*( 1)
  DumpPrelude        571*( 1)

-E-
  EachActualPart     725*( 1) 1133*( 1)
  EachVirtualPart    668*( 1)  726*( 1) 1134*( 1)
  Enable            1510*( 1)
  EndCmd            1525*( 1)
  EndCut             633*( 1)
  EndWait            478*( 1)
  EnterPageEditing   917*( 1)
  erAborted          220*( 1)
  Erase             1492*( 1)
  erBadData          229*( 1)
  erCantRead         230*( 1)
  erCantWrite        231*( 1)
  erDirtyDoc         232*( 1)
  erDuplicateName    221*( 1)
  erInvalidName      222*( 1)
  erMaxToolKit       237*( 1)
  erNameNotFound     223*( 1)
  erNoDiskSpace      235*( 1)
  erNoMemory         234*( 1)
  erNoMoreDocs       233*( 1)
  erPassword         227*( 1)
  erVersion          228*( 1)
  erWrongPassword    236*( 1)
  evenOnly           856*( 1)
  Events              46*( 1)
```

eventTime 1579•(1)
eventType 1580•(1)
ExpandMemory 606•(1)
extentLRect 715 (1)

-F-
famClassic 298 (1)
family 1545 (1)
famMin 297 (1) 298 (1)
famModern 297 (1)
fApple 400•(1)
fCountHeap 1583•(1)
fExperimentFng 369•(1) 1582•(1)
FilerComm 65•(1)
files 544 (1)
FillIcon 1465•(1)
FilterAndDo 669•(1) 727•(1) 1135•(1)
FilterDispatch 1136•(1)
FindBranchThatIs 1271•(1)
fingerCursor 95•(1)
firstBox 1478 (1)
firstPrivateEven 366•(1)
firstUserCursor 97•(1)
fitPagesPerfectl 759•(1)
Folders 47•(1)
FontMgr 33•(1) 37•(1)
fonts 1586•(1)
fOption 400•(1)
ForceBreakAt 781•(1)
frameBody 894•(1)
frameKind 162•(1)
FreedAndReplaced 949•(1)
freeOnDismissal 1154•(1)
fShift 400•(1)

-G-
genClipPic 1588•(1)
GetAlert 479•(1)
GetCmdName 1513•(1)
GetHysteresis 952•(1)
GetPageLimits 918•(1)
GetPrefixPart 1618•(1)
GetPrinterMetric 1129•(1)
GetScrollLimits 1374•(1)
GetSize 1454•(1)
GetStdScroll 776•(1)
GetTime 1632•(1)
GetTitle 1048•(1)
gotPane 1241•(1)

-H-
HandlePrivateEve 505•(1)
hasIcon 620•(1)
hasPicture 618•(1)
hasUniversalText 619•(1)
hasView 617 (1)
HaveView 728•(1) 953•(1) 1261•(1) 1371•(1)
headings 889•(1)
hiddenCursor 89•(1)
highLevel 1589•(1)
Highlight 983•(1) 1064•(1) 1303•(1)
HighlightMenu 1526•(1)
highToggle 1590•(1)
hiliteAfter 661•(1)
Hit 729•(1)
HitScroller 1291•(1)

-I-
iconNameSeparato 83•(1)
IdleBegin 967•(1) 1140•(1)
IdleContinue 968•(1) 1141•(1)
IdleEnd 969•(1) 1142•(1)
IDLength 211•(1)
idleTime 1591•(1)
iFlipBack 409•(1)
iFlipFwd 409•(1)
iGrayA 409•(1)
iGrayB 409•(1)
image 655•(1)
inBackground 1592•(1)
Init 908•(1)
InitProcess 1630•(1)
Insert 1282•(1) 1533•(1)
Inspect 640•(1)
INTRINSIC 18•(1)
Invalidate 730•(1) 1305•(1)
InvalLRect 1307•(1)
IsActive 1049•(1)
iScrollBack 409•(1)
iScrollFwd 409•(1)
isCut 681 (1)
iSkewer 409•(1)
isLoaded 1500 (1)
isMainView 763•(1)
isPrintable 762•(1)
isResizable 1017•(1)
IsVisible 1050•(1)
isVisible 1479•(1)
itemIndex 442•(1)
iThumb 409•(1)

-K-
KeyBack 972•(1)
KeyChar 973•(1)
KeyClear 974•(1)
KeyEnter 975•(1)
KeyForward 976•(1)
KeyPause 977•(1)
keyResponse 1150 (1)
KeyReturn 978•(1)

KeyTab 979•(1)
KillSegments 607•(1)
kind 938•(1)

-L-
language 421•(1)
lastClick 1240•(1)
lastCmd 1026•(1)
LaunchLayoutBox 731•(1)
layEditLegendKin 161•(1)
layoutDialogBox 892•(1)
layPickKind 160•(1)
limboPen 1595•(1)
LoadMenuBar 1078•(1)
LocateOnPage 874•(1)
LPattern 1595 (1)
LPoint 750 (1) 764 (1) 853 (1) 939 (1) 940 (1)
LRect 715 (1) 887 (1) 895 (1) 896 (1) 898 (1) 941 (1) 1606 (1)

-M-
MakeBand 1338•(1)
MakeSegments 608•(1)
manualBreakPen 1594•(1)
mapping 1501•(1)
marginPad 1425 (1)
marginPattern 1595•(1)
MarkChanged 956•(1)
maxFonts 78•(1) 1586 (1)
maxInnerSize 1023•(1)
maxMenus 77•(1) 1500 (1)
maxPage 858•(1)
MaxPageToPrint 777•(1)
maxSegments 79•(1) 557 (1) 1607 (1)
maxSegSize 80•(1)
mBuzzword 359•(1) 361•(1)
menuBar 1596•(1)
MenuEventAt 1079•(1)
menuIndex 441•(1)
menuResponse 1151•(1)
Menus 48•(1)
MenuWithID 1537•(1)
minInnerDiagonal 1235•(1)
minPage 857•(1)
mMove 411•(1)
mnuClipFilePrint 364•(1)
MouseMove 734•(1) 989•(1)
MousePress 735•(1) 988•(1)
MouseRelease 736•(1) 990•(1)
MouseTrack 737•(1) 1378•(1)
MoveBackToAnchor 991•(1)
MoveSplitBefore 1339•(1)
MoveThumb 1466•(1)
mPress 411•(1)
mPrvuBreaks 381•(1)
mPrvuMargins 381•(1)
mPrvuOff 381•(1)
mRelease 411•(1)
myProcessID 1597•(1)
myTool 1598•(1)

-N-
NameToPrefix 1107•(1)
NewBand 1340•(1)
NewCommand 963•(1) 1080•(1)
NewDocManager 513•(1)
NewPageView 920•(1)
NewPaginatedView 919•(1)
NewPane 1342•(1)
NewStatusView 1264•(1)
NewStdCommand 1081•(1)
NewView 1262•(1)
NewWindow 581•(1)
noCmdNumber 101•(1)
noCursor 88•(1)
noId 209•(1)
noIDNumber 208•(1)
nonNullBody 1426•(1)
normalPen 1599•(1)
NoSelection 807•(1)
Note 480•(1)
nothingKind 99•(1)
numCommands 1503•(1)
numMenus 1502•(1)
numSegments 427•(1)

-O-
ObeyEvents 495•(1)
ObeyFilerEvent 500•(1)
ObeyTheEvent 501•(1)
objectToFree 1034•(1)
oddOnly 855•(1)
OffSetBy 732•(1)
offsetFromAlignm 853•(1)
OffsetPanes 1191•(1)
OffSetTo 733•(1)
okString 1600•(1)
OKToDrawIn 791•(1) 1309•(1)
OnAllPadsDo 1311•(1)
Open 588•(1)
OpenBlank 589•(1)
openedAsTool 564•(1)
OpenSaved 590•(1)
OpenSuspended 591•(1)

-P-
pageAlignment 852•(1)
pageMargins 887•(1)
pageNumber 1402•(1)
pageRiseDirectio 902•(1)
pageSize 816•(1)

pageView 881•(1)
PageWith 921•(1)
PagifyLPoint 834•(1)
paginatedView 1229•(1)
panel 749 (1) 936•(1) 1177•(1) 1366•(1)
panels 1006 (1)
panelToPrint 1032•(1)
panelTree 1007•(1)
panes 1176•(1) 1226•(1)
PaneShowing 1272•(1)
PaneToScroll 1328•(1)
paperLRect 895•(1)
paperRect 375•(1)
Passwd 30•(1)
password 418•(1) 549•(1)
pendingNote 563•(1)
PenState 1559 (1) 1593 (1) 1594 (1) 1599 (1)
percentFromEdge 388•(1)
Perform 673•(1)
PerformCommand 964•(1) 1082•(1)
PerformLast 1083•(1)
pgRgOK 1030•(1)
pgSzOK 1028•(1)
phAborting 143•(1)
phAlignment 172•(1)
phBotCenter 177•(1)
phBotCluster 191•(1)
phBotLeft 176•(1)
phBotRight 178•(1)
phBottom 190•(1)
phCancel 198•(1)
phCantRevert 149•(1)
phCantSave 148•(1)
phCantUndo 138•(1)
phCentimeters 183•(1)
phCmTitle 194•(1)
phConverting 142•(1)
phCountry 151•(1)
phDialogUp 137•(1)
phEditClip 134•(1)
phEvenOnly 167•(1)
phInches 182•(1)
phInchTitle 193•(1)
phInstallMargins 192•(1)
phLaunchHeading 179•(1)
phLeft 184•(1)
phLeftCluster 185•(1)
phMaxPage 170•(1)
phMinPage 169•(1)
phNewerVersion 141•(1)
phNewHeading 195•(1)
phNoClip 135•(1)
phNoCommand 139•(1)
phNoInsPt 126•(1)
phNoSel 125•(1)
phNoText 124•(1)
phOddEven 165•(1)
phOddOnly 166•(1)
phOddOrEven 168•(1)
phOK 197•(1)
phOlderVersion 140•(1)
phPage 145•(1)
phPageAlignment 171•(1)
phPageMargins 180•(1)
Phrase 482•(1)
phraseFile 1601•(1)
phRevBlank 128•(1)
phRevert 127•(1)
phRight 188•(1)
phRightCluster 189•(1)
phSaving 132•(1)
phSelCant 130•(1)
phTerminated 133•(1)
phTitle 146•(1)
phTooManyChars 164•(1)
phTop 186•(1)
phTopCenter 174•(1)
phTopCluster 187•(1)
phTopLeft 173•(1)
phTopRight 175•(1)
phTrouble 122•(1)
phUnchanged 131•(1)
phUnits 181•(1)
phUnkClip 136•(1)
phUnkCmd 129•(1)
phUnknown 123•(1)
phWordDelimiters 120•(1)
PicGrpBegin 1627•(1)
PicGrpEnd 1628•(1)
PickStdCursor 1124•(1)
PicTextBegin 1625•(1)
PicTextEnd 1626•(1)
pixelsFromEdge 388•(1)
PMDecl 42•(1)
Point 377 (1) 397 (1) 752 (1) 765 (1) 1023 (1) 1235 (1) 1243 (1) 1248 (1) 1249 (1)
PrefixToName 1108•(1)
preludePtr 558•(1)
preludeSize 422•(1)
Preview 1317•(1)
previewMode 1239•(1)
Print 922•(1) 1130•(1)
printableLRect 896•(1)
printerMetrics 900•(1) 1027•(1)
printLdsn 107•(1)
printManager 751•(1) 851 (1)
printPref 425•(1)
printRect 376•(1)
PrintView 1320•(1)
PrMgr 60•(1)
PrMgrUtil 59•(1)

process 1602•(1)
PrProcs 51•(1)
PrPublic 63•(1)
PrStd 44•(1)
PrStdInfo 62•(1)
Ptr 1537 (1)
Publicize 641•(1)
PutCmdName 1517•(1)
PutUpDialogBox 1058•(1)

-Q-
QuickDraw 35•(1)

-R-
ReactToPrinterCh 738•(1) 923•(1)
RecalcExtent 739•(1)
Recompute 1432•(1)
Rect 375 (1) 376 (1) 1247 (1) 1426 (1)
RedoBreaks 783•(1) 924•(1)
redoPhase 405•(1)
refnum 557•(1)
RemakePanes 1343•(1)
RemapManualBreak 784•(1)
RememberCommand 481•(1)
RememberSplit 1344•(1)
Remove 1285•(1)
RepaneOrthogonal 1345•(1)
Replace 1286•(1)
res 377•(1) 752•(1)
Rescroll 1313•(1)
reserve 378•(1)
Resize 740•(1) 1070•(1) 1388•(1)
ResizeBand 1354•(1)
resizeBranch 1236•(1)
ResizePanes 1202•(1)
ResizeTo 1071•(1)
ResplitAt 1469•(1)
Restore 994•(1)
RestoreSelection 1094•(1)
RestoreSplits 1346•(1)
ResumeAfterOpen 609•(1)
Reveal 998•(1)
revealAll 413•(1)
RevealLRect 1332•(1)
revealNone 413•(1)
RevealSelection 1095•(1)
revealSome 413•(1)
revelation 658•(1)
RevertVersion 592•(1)
Rework 1408•(1)
RgnHandle 1605 (1)
Run 502•(1)

-S-
Save 995•(1)
SaveCommand 1084•(1)
saveExists 551•(1)
SaveSelection 1096•(1)
SaveVersion 593•(1)
sBoxID 1446•(1)
Scrap 57•(1)
screenPad 754•(1)
screenRightEdge 1604•(1)
scrollBar 1444 (1)
scrollBars 1233•(1)
ScrollBy 1192•(1) 1391•(1)
ScrollDir 1455•(1)
scrollDir 1179•(1)
scroller 1178•(1)
scrollPastEnd 765•(1)
scrollRgn 1605•(1)
ScrollStep 1194•(1)
ScrollTo 1195•(1)
ScrollToReveal 1393•(1)
SeesSameAs 741•(1)
selection 1230•(1)
selectPanel 1009•(1)
SelectParagraphs 980•(1)
selectWindow 1013•(1)
SendEvent 507•(1)
SetBreak 925•(1)
SetDfltHeadings 926•(1)
SetForPage 1410•(1) 1433•(1)
SetFunctionValue 804•(1)
SetInnerRect 1274•(1)
SetMinVieuSize 798•(1)
SetOuterRect 1275•(1)
SetSegSize 610•(1)
SetSize 1456•(1)
SetupMenus 1086•(1)
SetWmgrId 1051•(1)
SetZoomFactor 1314•(1)
ShouldDraw 869•(1)
ShouldFrame 870•(1)
shouldSuspend 552•(1)
shouldToolSave 553•(1)
ShouSideBand 1349•(1)
SideBandRect 1350•(1)
size10Pitch 303 (1)
size12Pitch 302 (1)
size12Point 304 (1)
size14Point 305 (1)
size15Pitch 301 (1)
size18Point 306 (1)
size20Pitch 300 (1)
size24Point 307 (1)
sizeMin 300 (1) 301 (1) 302 (1) 303 (1) 304 (1) 305 (1) 306 (1) 307 (1)
SkipPage 927•(1)
smCrossCursor 94•(1)
SplitAt 1470•(1)

StashPicture 1103*(1)
stdBottomBorder 216*(1)
stdBoxHeight 201*(1)
stdBoxSpacing 202*(1)
stdBoxWidth 200*(1)
stdBtnHeight 206*(1)
stdCurvH 204*(1)
stdCurvV 205*(1)
stdHHysteresis 85*(1)
stdLeftRightBord 215*(1)
stdMargins 1606*(1)
stdScroll 764*(1)
stdSlimTitleHeig 214*(1)
stdTitleHeight 213*(1)
stdVHysteresis 86*(1)
Stop 483*(1)
STRING 1564 (1) 1577 (1) 1600 (1) 1607 (1) 1610 (1) 1613 (1) 1614 (1) 1615 (1)
Suspend 595*(1)
suspendSuffix 1607*(1)
SysCall 28*(1)

-T-
TAbilities 386*(1) 1234 (1)
TakeDownDialogBo 1059*(1)
TAlertArg 390*(1)
TAlertCounter 391*(1)
TAlignment 393*(1)
TArea 1003 (1) 1007 (1) 1172 (1) 1222 (1)
TArray 883 (1) 1252 (1) 1501 (1)
TBand 1172*(1) 1184 (1) 1207 (1) 1217 (1) 1341 (1) 1445 (1)
TBodyPad 1403 (1) 1422*(1) 1431 (1) 1608 (1)
TBranchArea 1236 (1) 1271 (1)
TClassWorld 529 (1)
TClickState 396*(1) 1565 (1)
TClipboard 614*(1) 628 (1) 1562 (1) 1566 (1)
TCmdNumber 403*(1) 653 (1) 1520 (1) 1521 (1)
TCmdPhase 405*(1) 659 (1) 661 (1)
TCommand 650*(1) 665 (1) 678 (1) 697 (1) 963 (1) 1026 (1) 1080 (1) 1081 (1)
TCursorNumber 407*(1) 723 (1) 1123 (1) 1299 (1) 1382 (1) 1575 (1)
TCutCopyCommand 678*(1) 685 (1)
TDialogBox 892 (1) 1008 (1) 1147*(1) 1159 (1)
TDiResponse 383*(1) 1150 (1) 1151 (1) 1152 (1)
TDocDirectory 428 (1) 525*(1) 533 (1)
TDocManager 514 (1) 541*(1) 567 (1) 614 (1) 1563 (1) 1569 (1) 1572 (1)
TEnumAbilities 385*(1) 386 (1)
TEnumIcons 409*(1)
textCursor 92*(1)
TFilePath 546 (1) 547 (1) 1611 (1) 1612 (1)
TFileScanner 624 (1) 1601 (1)
TFont 1542*(1) 1548 (1) 1586 (1)
THeading 849*(1) 863 (1)
THeap 561 (1)
theBodyPad 1608*(1)
theMarginPad 1609*(1)
THighTransit 1589 (1) 1590 (1)
ThumbPos 1196*(1)
ThumbRange 1457*(1)
ThumbTo 1197*(1)
TImage 655 (1) 712*(1) 721 (1) 731 (1) 746 (1) 849 (1)
TList 889 (1) 1006 (1) 1176 (1) 1226 (1) 1232 (1) 1578 (1)
tlSideBandSize 1248*(1)
TMarginPad 1398*(1) 1406 (1) 1425 (1) 1609 (1)
TMenuBar 1497*(1) 1506 (1) 1596 (1)
TMousePhase 411*(1)
TObject 446 (1) 525 (1) 541 (1) 650 (1) 712 (1) 879 (1) 932 (1) 1034 (1) 1441 (1) 1475 (1)
1497 (1) 1542 (1)
ToggleFlag 1043*(1)
toolName 1610*(1)
ToolOfFile 1619*(1)
ToolOfProcess 1620*(1)
toolPrefix 1611*(1)
toolVolume 1612*(1)
topOrLeft 1209 (1)
TPad 754 (1) 1362 (1) 1398 (1) 1422 (1)
TPageAlignment 394*(1) 852 (1)
TPageView 841*(1) 844 (1)
TPaginatedView 811*(1) 825 (1) 919 (1) 1229 (1)
TPane 1242 (1) 1272 (1) 1328 (1) 1342 (1) 1362*(1) 1370 (1)
TPanel 749 (1) 936 (1) 1009 (1) 1010 (1) 1011 (1) 1012 (1) 1032 (1) 1177 (1) 1222*(1) 1259 (1)
1281 (1) 1366 (1)
TPassword 549 (1)
TPasteCommand 697*(1) 701 (1)
TPPrelude 432*(1) 558 (1)
TPrelude 416*(1) 432 (1)
TPreviewMode 381*(1) 1239 (1)
TPrinterMetrics 374*(1) 900 (1) 1027 (1)
TPrintManager 751 (1) 851 (1) 879*(1) 907 (1)
TProcess 446*(1) 451 (1) 1602 (1)
TPrReserve 415*(1) 425 (1) 1567 (1)
TrackCursor 469*(1)
TrackSkewer 1460*(1)
TrackThumb 1462*(1)
TRevelation 413*(1) 658 (1)
TSBoxID 434*(1) 1446 (1)
TScaler 1238 (1)
TScrollBar 1233 (1) 1444 (1) 1475*(1) 1483 (1)
TScroller 1178 (1) 1441*(1) 1450 (1) 1478 (1)
TSelection 807 (1) 932*(1) 942 (1) 947 (1) 949 (1) 1230 (1) 1231 (1)
TSideBand 1207*(1) 1214 (1)
TTypeStyle 1570 (1)
TUnitsFromEdge 388*(1)
TView 716 (1) 746*(1) 771 (1) 811 (1) 814 (1) 841 (1) 880 (1) 881 (1) 920 (1) 937 (1)
1227 (1) 1228 (1) 1263 (1) 1264 (1) 1365 (1) 1401 (1)
TWindow 528 (1) 562 (1) 575 (1) 581 (1) 935 (1) 1003*(1) 1013 (1) 1015 (1) 1038 (1) 1147 (1)
1175 (1) 1225 (1) 1574 (1)
TWindowID 436*(1) 1016 (1) 1557 (1)
TWmgrCmd 438*(1)

-U-
ulOPitch 303*(1)

```
u12Pitch          302•( 1)
u12Point          304•( 1)
u14Point          305•( 1)
u15Pitch          301•( 1)
u18Point          306•( 1)
u20Pitch          300•( 1)
u24Point          307•( 1)
UABC               13•( 1)
uAddColumnStrip   334•( 1)
uAddRowStrip      335•( 1)
uBackspace        257•( 1)
uBold             310•( 1)
uCheckIndices     341•( 1)
uClassic          298•( 1)
uClear            281•( 1)
uClearBreaks      329•( 1)
uCmdInstallMargi  158•( 1)
uCmdLaunchHeadIn  157•( 1)
uCopy             274•( 1)
uCountHeap        339•( 1)
uCreateLayoutBox  155•( 1)
uCut              275•( 1)
uDesignPages      321•( 1)
uDocScramble      349•( 1)
UDraw              39•( 1)
uDumpGlobals      342•( 1)
uDumpPrelude      343•( 1)
uEditDialog       351•( 1)
uEnter            258•( 1)
uExperimenting    344•( 1)
uFnt0             284•( 1)
uFnt1             285•( 1)
uFnt10            294•( 1)
uFnt11            295•( 1)
uFnt2             286•( 1)
uFnt3             287•( 1)
uFnt4             288•( 1)
uFnt5             289•( 1)
uFnt6             290•( 1)
uFnt7             291•( 1)
uFnt8             292•( 1)
uFnt9             293•( 1)
uForwardSpace     259•( 1)
uFreeGarbage      346•( 1)
uItalic           311•( 1)
uKeyDown          272•( 1)
uMainScramble     348•( 1)
uModern           297•( 1)
UMousePress       269•( 1)
uMoveLayoutBoxes  156•( 1)
uMoveWindow       271•( 1)
Unbind            611•( 1)
undoable          656•( 1)
undoClickPanel   1012•( 1)
UndoCut           637•( 1)
UndoLast         1087•( 1)
undoPhase         405•( 1)
undoSelection    1231•( 1)
undoSelPanel     1010•( 1)
undoSelWindow    1015•( 1)
unHiliteBefore    659•( 1)
UnitHz             26•( 1)
UnitStd            25•( 1)
Unload           1534•( 1)
unpaginatedView   814 ( 1)
unused            423•( 1)
UObject            27•( 1)
uOutline          314•( 1)
uPaste            276•( 1)
Update           1066•( 1)
uPlain            309•( 1)
uPrFmt            244•( 1)
uPrint            246•( 1)
uPrintAsIs        245•( 1)
uPrMonitor        247•( 1)
uPrvuBreaks       319•( 1)
uPrvuMargins      318•( 1)
uPrvuOff          320•( 1)
uPutAway          243•( 1)
uReduce70Pct      324•( 1)
uReduceToFit      325•( 1)
uReportEvents     337•( 1)
uReptGarbage      345•( 1)
uResizePanel      268•( 1)
uResizeWindow     267•( 1)
uReturn           260•( 1)
uRevertVersion    249•( 1)
uRiseHorizontall  332•( 1)
uRiseVertically   331•( 1)
uSaveVersion      248•( 1)
uScrolling        265•( 1)
uSelAll           277•( 1)
uSetAllAside      241•( 1)
uSetAside         242•( 1)
uSetClipAside     251•( 1)
uSetHorzBreak     327•( 1)
uSetVertBreak     328•( 1)
uShadow           313•( 1)
uShowFullSize     323•( 1)
uSomeCommand      264•( 1)
uSplitting        266•( 1)
uStopEditDialog   352•( 1)
uSubscript        316•( 1)
uSuperscript      315•( 1)
uTab              261•( 1)
uThumbing         270•( 1)
utRedoLast        280•( 1)
utSetAside        250•( 1)
utUndoLast        279•( 1)
uTyping           254•( 1)
```

```
  uUnderline          312*( 1)
  uUndoLast           278*( 1)

-V-
  varPage            1613*( 1)
  varTitle           1614*( 1)
  version             419*( 1)
  VMSelect            816 ( 1)  883 ( 1)  902 ( 1) 1179 ( 1) 1232 ( 1) 1233 ( 1) 1234 ( 1) 1455 ( 1)
  view                716*( 1)  880 ( 1)  937*( 1) 1228*( 1) 1401 ( 1)
  ViewLCd            1188*( 1)
  volume              547*( 1)
  volumePrefix        546*( 1)

-W-
  WantMenu           1088*( 1)
  when                398*( 1)
  where               397*( 1)
  window              528 ( 1)  562*( 1)  935 ( 1) 1175 ( 1) 1225 ( 1)
  WindowWithId        575*( 1)
  wmgrID             1016*( 1)
  WMLCrs               54*( 1)
  WMLGrow              56*( 1)
  WMLSb                55*( 1)
  WMLstd               53*( 1)
  wordDelimiters     1615*( 1)
  workingInMargins    821*( 1)

-Z-
  zoomed             1237*( 1)
  zoomFactor         1238*( 1)

*** End Xref:  853 id's   1159 references     [391836 bytes/4146 id's/40687 refs]
```