```
  ************************************************************************
 Constructor Function and Procedure List of maXbox 3.9.9
************************************************************************

//////////////////////////////////////////////////////////////////////////
ref Help Extraxt of EXE Functions of maxbox3.exe BigBitBox API HEX in BOX
------------------------------------------------------------------------

File Size of EXE:20079616 V3.9.9.94 April 2014 To EKON/BASTA/JAX/IBZ/SWS
****************Now the Funclist****************************************
Funclist Function : 12264 //10766 //10165 (7648)
****************Now the Proclist****************************************
Proclist Procedure Size is: 7632  //6792 //6401 4752
****************Now Constructors****************************************
Constructlist Constructor Size is: 1253 //995 //
def head:max: maXbox7: 12.04.2014 19:11:55
file E:\maxbox\maxbox3\docs\maxbox_extract_funclist399.txt
doc file: maxbox_extract_funclist399_.txt (sort function list)
------------------------------------------------------------------------
Funclist total Size all is: 21149! Constructor, Function and Procedure
AExtraxt of EXE Functions of maxbox3.exe, locs of file = 20161
ASize of EXE: 20079616 (16586240) (13511680) (13023744)
SHA1 Hash of maXbox 3.9.9.94: 746418954E790A1713A93FF786ABCB50097FCB59

------------------------------------------------------------------------
------------------------------------------------------------------------
//////////////////////////////////////////////////////////////////////////


FUNCTION Metric of Script: 256_findfunctions2_of_EXE.txt
Function ************Now the Funclist****************
function  GetResStringChecked(Ident: string; const Args: array of const): string
Function ( Index : Longint) : Integer
function (Command: Word; Data: Longint; var CallHelp: Boolean): Boolean
Function _CheckAutoResult( ResultCode : HResult) : HResult
function _T(Name: tbtString): Variant;
function ABNFToText(const AText : String) : String
Function Abs(e : Extended) : Extended;
Function Ackermann( const A, B : Integer) : Integer
Function AcquireLayoutLock : Boolean
Function ActionByName( const AName : string) : TWebActionItem
Function ACTIVEBUFFER : PCHAR
Function Add : TAggregate
function Add : TCollectionItem
Function Add : TColumn
Function Add : TComboExItem
Function Add : TCookie
Function Add : TCoolBand
Function Add : TFavoriteLinkItem
Function Add : TFileTypeItem
Function Add : THeaderSection
Function Add : THTMLTableColumn
Function Add : TIdEMailAddressItem
Function Add : TIdMessagePart
Function Add : TIdUserAccount
Function Add : TListColumn
Function Add : TListItem
Function Add : TStatusPanel
Function Add : TTaskDialogBaseButtonItem
Function Add : TWebActionItem
Function Add : TWorkArea
Function Add( AClass : TClass) : Integer
Function Add( AComponent : TComponent) : Integer
Function Add( AItem, AData : Integer) : Integer
Function Add( AItem, AData : Pointer) : Pointer
Function Add( AItem, AData : TObject) : TObject
Function Add( AObject : TObject) : Integer
Function Add( const Access, Count : Cardinal; const Offset : Int64) : Integer
Function Add( const S : WideString) : Integer
Function Add( Image, Mask : TBitmap) : Integer
Function Add( Index : LongInt; const Text : string) : LongInt
Function Add( Sibling : TTreeNode; const S : string) : TTreeNode
Function Add(const S: string): Integer
function Add(S: string): Integer;
Function AddAt( const Access, Count : Cardinal; const Offset : Int64; const Address: Pointer) : Integer
Function ADDCHILD : TFIELDDEF
Function AddChild( Index : LongInt; const Text : string) : LongInt
Function AddChild( Parent : TTreeNode; const S : string) : TTreeNode
Function AddChildFirst( Parent : TTreeNode; const S : string) : TTreeNode
Function AddChildObject( Index : LongInt; const Text : string; const Data : Pointer) : LongInt
Function AddChildObject( Parent : TTreeNode; const S : string; Ptr : Pointer) : TTreeNode
Function AddChildObjectFirst( Parent : TTreeNode; const S : string; Ptr : Pointer) : TTreeNode
Function ADDFIELDDEF : TFIELDDEF
Function AddFileExtIfNecessary( AFileName, AExt : string) : string
Function AddFirst( Sibling : TTreeNode; const S : string) : TTreeNode
Function AddIcon( Image : TIcon) : Integer
Function AddImage( Value : TCustomImageList; Index : Integer) : Integer
Function ADDINDEXDEF : TINDEXDEF
Function AddItem(const Caption:String;const ImageIdx,SelectImageIdx,OverlayImagIdx,
Indent:Int;Data:Ptr):TComboExItem
```

```
Function AddItem( Item : THeaderSection; Index : Integer) : THeaderSection
Function AddItem( Item : TListItem; Index : Integer) : TListItem
Function AddItem( Item : TStatusPanel; Index : Integer) : TStatusPanel
Function AddMapping( const FieldName : string) : Boolean
Function AddMasked( Image : TBitmap; MaskColor : TColor) : Integer
Function AddModuleClass( AClass : TComponentClass) : TComponent
Function AddModuleName( const AClass : string) : TComponent
Function AddNode(Node,Relative: TTreeNode;const S: string;Ptr:Pointer;Method:TNodeAttachMode):TTreeNode
Function AddObject( const S : WideString; AObject : TObject) : Integer
Function AddObject( Index : LongInt; const Text : string; const Data : Pointer) : LongInt
Function AddObject( Sibling : TTreeNode; const S : string; Ptr : Pointer) : TTreeNode
function AddObject(S:String;AObject:TObject):integer
Function AddObjectFirst( Sibling : TTreeNode; const S : string; Ptr : Pointer) : TTreeNode
Function AddParameter : TParameter
Function AddParamSQLForDetail(Params:TParams;SQL:WideStr;Native:Bool;QuoteChar:WideString):WideString
 Function Addr64ToAddr32(const Value: TJclAddr64): TJclAddr32;
Function Addr32ToAddr64(const Value: TJclAddr32): TJclAddr64;
function AdjustLineBreaksS(const S: string): string)
TTextLineBreakStyle', '(tlbsLF, tlbsCRLF)
Function AdjustLineBreaks(const S: string; Style: TTextLineBreakStyle): string;
Function AllData : string
function AllocMemCount: integer;
function AllocMemSize: integer;
Function AllocPatternBitmap( BkColor, FgColor : TColor) : TBitmap
Function AllowRegKeyForEveryone( Key : HKEY; Path : string) : Boolean
Function AlphaComponent( const Color32 : TColor32) : Integer
Function AlphaSort : Boolean
Function AlphaSort( ARecurse : Boolean) : Boolean
Function AnsiCat( const x, y : AnsiString) : AnsiString
Function AnsiCompareFileName( S1, S2 : string) : Integer
function AnsiCompareFileName(const S1: string; const S2: string): Integer)
Function AnsiCompareStr( S1, S2 : string) : Integer
function AnsiCompareStr(const S1: string; const S2: string): Integer;)
Function AnsiCompareText( S1, S2 : string) : Integer
function AnsiCompareText(const S1: string; const S2: string): Integer;)
Function AnsiContainsStr( const AText, ASubText : string) : Boolean
Function AnsiContainsText( const AText, ASubText : string) : Boolean
Function AnsiCopy( const src : AnsiString; index, count : Integer) : AnsiString
Function AnsiDequotedStr( S : string; AQuote : Char) : string
Function AnsiEndsStr( const ASubText, AText : string) : Boolean
Function AnsiEndsText( const ASubText, AText : string) : Boolean
Function AnsiExtractQuotedStr( var Src : PChar; Quote : Char) : string
function AnsiExtractQuotedStr(var Src: PChar; Quote: Char): string)
Function AnsiIndexStr( const AText : string; const AValues : array of string) : Integer
Function AnsiIndexText( const AText : string; const AValues : array of string) : Integer
Function AnsiLastChar( S : string) : PChar
function AnsiLastChar(const S: string): PChar)
Function AnsiLeftStr( const AText : AnsiString; const ACount : Integer) : AnsiString
Function AnsiLowerCase( S : string) : string
Function AnsiLowercase(s : String) : String;
Function AnsiLowerCaseFileName( S : string) : string
Function AnsiMatchStr( const AText : string; const AValues : array of string) : Boolean
Function AnsiMatchText( const AText : string; const AValues : array of string) : Boolean
Function AnsiMidStr( const AText : AnsiString; const AStart, ACount : Integer) : AnsiString
Function AnsiPos( const src, sub : AnsiString) : Integer
Function AnsiPos( Substr, S : string) : Integer
function AnsiPos(const Substr: string; const S: string): Integer;)
Function AnsiQuotedStr( S : string; Quote : Char) : string
Function AnsiReplaceStr( const AText, AFromText, AToText : string) : string
Function AnsiReplaceText( const AText, AFromText, AToText : string) : string
Function AnsiResemblesText( const AText, AOther : string) : Boolean
Function AnsiReverseString( const AText : AnsiString) : AnsiString
Function AnsiRightStr( const AText : AnsiString; const ACount : Integer) : AnsiString
function AnsiSameCaption(const Text1: string; const Text2: string): Boolean)
Function AnsiSameStr( S1, S2 : string) : Boolean
function AnsiSameStr(const S1: string; const S2: string): Boolean)
Function AnsiSameText( const S1, S2 : string) : Boolean
Function AnsiSameText( S1, S2 : string) : Boolean
function AnsiSameText(const S1: string; const S2: string): Boolean)
Function AnsiStartsStr( const ASubText, AText : string) : Boolean
Function AnsiStartsText( const ASubText, AText : string) : Boolean
Function AnsiStrComp( S1, S2 : PChar) : Integer
function AnsiStrComp(S1: PChar; S2: PChar): Integer)
Function AnsiStrIComp( S1, S2 : PChar) : Integer
function AnsiStrIComp(S1: PChar; S2: PChar): Integer)
Function AnsiStrLastChar( P : PChar) : PChar
function AnsiStrLastChar(P: PChar): PChar)
Function AnsiStrLComp( S1, S2 : PChar; MaxLen : Cardinal) : Integer
Function AnsiStrLIComp( S1, S2 : PChar; MaxLen : Cardinal) : Integer
Function AnsiStrLower( Str : PChar) : PChar
Function AnsiStrPos( Str, SubStr : PChar) : PChar
function AnsiStrPos(Str: PChar; SubStr: PChar): PChar)
Function AnsiStrScan(Str: PChar; Chr: Char): PChar)
Function AnsiStrUpper( Str : PChar) : PChar
Function AnsiToUtf8( const S : string) : UTF8String
Function AnsiToUtf8Ex( const S : string; const cp : integer) : UTF8String
Function AnsiUpperCase( S : string) : string
Function AnsiUppercase(s : String) : String;
Function AnsiUpperCaseFileName( S : string) : string
Function ApplyUpdates(const Delta: OleVariant;MaxErrors:Integer; out ErrorCount: Integer): OleVariant
Function ApplyUpdates(const Delta:OleVariant;MaxErrors: Integer;out ErrorCount: Integer) : OleVariant;
Function ApplyUpdates( MaxErrors : Integer) : Integer
Function ApplyUpdates1(const Delta:OleVar;MaxErrs:Int;out ErrCount:Int;var OwnerData:OleVar):OleVariant;
Function ArcCos( const X : Extended) : Extended
Function ArcCosh( const X : Extended) : Extended
Function ArcCot( const X : Extended) : Extended
Function ArcCotH( const X : Extended) : Extended
Function ArcCsc( const X : Extended) : Extended
Function ArcCscH( const X : Extended) : Extended
Function ArcSec( const X : Extended) : Extended
Function ArcSecH( const X : Extended) : Extended
Function ArcSin( const X : Extended) : Extended
Function ArcSinh( const X : Extended) : Extended
Function ArcTan( const X : Extended) : Extended
Function ArcTan2( const Y, X : Extended) : Extended
Function ArithmeticMean( const X : TDynDoubleArray) : Float
function ArrayLength: integer;
Function AsHex( const AValue : T4x4LongWordRecord) : string
Function AsHex( const AValue : T5x4LongWordRecord) : string
Function ASNDecLen( var Start : Integer; const Buffer : string) : Integer
Function ASNDecOIDItem( var Start : Integer; const Buffer : string) : Integer
Function ASNEncInt( Value : Integer) : string
Function ASNEncLen( Len : Integer) : string
Function ASNEncOIDItem( Value : Integer) : string
Function ASNEncUInt( Value : Integer) : string
Function ASNItem( var Start : Integer; const Buffer : string; var ValueType : Integer) : string
Function ASNObject( const Data : string; ASNType : Integer) : string
Function Assigned(I: Longint): Boolean;
Function AspectRatio(aWidth, aHeight: Integer): String;
Function AsWideString( Field : TField) : WideString
Function AtLeast( ACount : Integer) : Boolean
Function AttemptToUseSharedMemoryManager : Boolean
Function Authenticate : Boolean
Function AuthenticateUser( const AUsername, APassword : String) : Boolean
Function Authentication : String
Function BatchMove( ASource : TBDEDataSet; AMode : TBatchMode) : Longint
Function BcdCompare( const bcd1, bcd2 : TBcd) : Integer
Function BcdFromBytes( const AValue : TBytes) : TBcd
Function BcdPrecision( const Bcd : TBcd) : Word
Function BcdScale( const Bcd : TBcd) : Word
Function BcdToBytes( const Value : TBcd) : TBytes
Function BCDToCurr( const BCD : TBcd; var Curr : Currency) : Boolean
Function BcdToDouble( const Bcd : TBcd) : Double
Function BcdToInteger( const Bcd : TBcd; Truncate : Boolean) : Integer
Function BcdToStr( const Bcd : TBcd) : string;
Function BcdToStrF(const Bcd : TBcd; Format: TFloatFormat; const Precision, Digits:Integer):string
function beep2(dwFreq, dwDuration: integer): boolean;
Function BeginPeriod( const Period : Cardinal) : Boolean
Function BeginTrans : Integer
Function BeginTransaction : TDBXTransaction;
Function BeginTransaction1( Isolation : TDBXIsolation) : TDBXTransaction;
function BigMulu(aone, atwo: string): string;
function BigNumber(aone, atwo: string): string;
function BigExp(aone, atwo: string): string;
function BigMul(aone, atwo: string): string;
function BigAdd(aone, atwo: string): string;
function BigSub(aone, atwo: string): string;
function BigFactorial(aone: string): string;
Function BinaryToDouble( ABinary : string; DefValue : Double) : Double
Function BinomialCoeff( N, R : Cardinal) : Float
function BinominalCoefficient(n, k: Integer): string;
Function BinStrToInt( const ABinary : String) : Integer
Function BinToByte(Binary: String): Byte;
function BinToHex2(Binary: String): string;
function BinToInt(Binary: String): Integer;
Function BinToChar(St: String): Char;
Function BinToStr(ans: string): string;
Function BitBlt(hdcDest:HDC;nXDest,nYDest,nWidth,nHeigh:Int;hdcSrc:HDC;nXSrc,nYSrc:Int;dwRop:DWORD):Bool;
Function BitmapsAreIdentical( ABitmap1, ABitmap2 : TBitmap) : Boolean
Function BitsHighest( X : Byte) : Integer;
Function BitsHighest1( X : ShortInt) : Integer;
Function BitsHighest2( X : SmallInt) : Integer;
Function BitsHighest3( X : Word) : Integer;
Function BitsHighest4( X : Integer) : Integer;
Function BitsHighest5( X : Cardinal) : Integer;
Function BitsHighest6( X : Int64) : Integer;
Function BitsLowest( X : Byte) : Integer;
Function BitsLowest1( X : Shortint) : Integer;
Function BitsLowest2( X : Smallint) : Integer;
Function BitsLowest3( X : Word) : Integer;
Function BitsLowest4( X : Cardinal) : Integer;
Function BitsLowest5( X : Integer) : Integer;
Function BitsLowest6( X : Int64) : Integer;
Function BitsNeeded( const X : Byte) : Integer;
Function BitsNeeded1( const X : Word) : Integer;
Function BitsNeeded2( const X : Integer) : Integer;
Function BitsNeeded3( const X : Int64) : Integer;
Function BlueComponent( const Color32 : TColor32) : Integer
Function BooleanToInteger( const Pb : Boolean) : Integer
Function BoolToStr(B: Boolean; UseBoolStrs: Boolean): string)
Function BoolToStr1(value : boolean) : string;
Function booltoint( aBool : Boolean) : LongInt
Function inttobool( aInt : LongInt) : Boolean
Function Bounds( ALeft, ATop, AWidth, AHeight : Integer) : TRect
function Bounds(ALeft, ATop, AWidth, AHeight: Integer): TRect)
Function BreakApart( BaseString, BreakString : string; StringList : TStrings) : TStrings
Function BrightColor( const Color : TColor; const Pct : Single) : TColor
Function BrightColorChannel( const Channel : Byte; const Pct : Single) : Byte
Function BufferRequest( Length : Integer) : TStream
Function BuildFileList( const Path : string; const Attr : Integer; const List : TStrings) : Boolean
Function Buttons : PTaskDialogButton
Function BytesPerScanline( PixelsPerScanline, BitsPerPixel, Alignment : Longint) : Longint
Function BytesToCardinal( const AValue : TIdBytes; const AIndex : Integer) : Cardinal
Function BytesToChar( const AValue : TIdBytes; const AIndex : Integer) : Char
Function BytesToInt64( const AValue : TIdBytes; const AIndex : Integer) : Int64
Function BytesToInteger( const AValue : TIdBytes; const AIndex : Integer) : Integer
Function BytesToIPv6( const AValue : TIdBytes; const AIndex : Integer) : TIdIPv6Address
Function BytesToShort( const AValue : TIdBytes; const AIndex : Integer) : Short
Function BytesToString(ABytes:TIdBytes; AStartIndex:Integer; AMaxCount:Integer): string;
Function BytesToStr(const Value: TBytes): String;
Function BytesToWord( const AValue : TIdBytes; const AIndex : Integer) : Word
Function ByteToBin(Int: Byte): String;
Function ByteToCharIndex( S : string; Index : Integer) : Integer
function ByteToCharIndex(const S: string; Index: Integer): Integer)
Function ByteToCharLen( S : string; MaxLen : Integer) : Integer
function ByteToCharLen(const S: string; MaxLen: Integer): Integer)
Function ByteToHex( const AByte : Byte) : string
Function ByteToOctal( const AByte : Byte) : string
Function ByteType( S : string; Index : Integer) : TMbcsByteType
function ByteType(const S: string; Index: Integer): TMbcsByteType)
Function CalcTitleRect( Col : TColumn; ARow : Integer; var MasterCol : TColumn) : TRect
Function CalculateDFAFingerprint( oStates : TList) : integer
function CallTerminateProcs: Boolean)
function CANFOCUS:BOOLEAN
Function CanLoad( const Ext : string) : Boolean
Function CanParse( AWebRequest : TWebRequest) : Boolean
Function CanSave( const Ext : string) : Boolean
Function CanStart( cChar : char) : boolean
Function CaptureScreen : TBitmap;
Function CaptureScreen1( Rec : TRect) : TBitmap;
Function CardinalToFourChar( ACardinal : LongWord) : string
Function CastSoapToNative(Info:PTypeInfo;const SoapData:WideString;NatData:Pointer;IsNull:Boolean): Boolean
Function CastSoapToVariant1( SoapInfo : PTypeInfo; const SoapData : WideString) : Variant;
Function Ceil( const X : Extended) : Integer
Function Ceil16( X : Integer) : Integer
Function Ceil4( X : Integer) : Integer
Function Ceil8( X : Integer) : Integer
Function Ceiling( const X : Extended) : Integer
Function CellRect( ACol, ARow : Longint) : TRect
Function CelsiusToFahrenheit( const AValue : Double) : Double
Function CenterPoint( const Rect : TRect) : TPoint
function CenterPoint(const Rect: TRect): TPoint)
Function ChangeFileExt( FileName, Extension : string) : string
function ChangeFileExt(const FileName: string; const Extension: string): string)
Function CharInSet2( const Ch : Char; const SetOfChar : TSetOfChar) : Boolean
Function CharInSet( const Ch : Char; const testSet: TSysCharSet): Boolean
Function CharIsInEOF( const AString : string; ACharPos : Integer) : Boolean
Function CharIsInSet( const AString : string; const ACharPos : Integer; const ASet : String) : Boolean
Function CharLength( S : String; Index : Integer) : Integer
Function CharRange( const AMin, AMax : Char) : String
function CharsetToIdent(Charset: Longint; var Ident: string): Boolean)
Function CharToBin(vChr: Char): String;
Function CharNext(lpsz: PChar): PChar; stdcall;
Function CharToByteIndex( S : string; Index : Integer) : Integer
function CharToByteIndex(const S: string; Index: Integer): Integer)
Function CharToByteLen( S : string; MaxLen : Integer) : Integer
function CharToByteLen(const S: string; MaxLen: Integer): Integer)
Function CharToHex(const APrefix : String; const cc : Char) : shortstring;
function CharToHexStr(Value: char): string);
function CharToOem(ins, outs: PChar):boolean;
function CharToUniCode(Value: Char): string;
Function CheckMenuDropdown : Boolean
Function CheckMessages : longint
Function CheckOpen( Status : DBIResult) : Boolean
Function CheckPassword( const APassword : String) : Boolean
Function CheckResponse(const AResponse:SmallInt;const AAllowedResponses:array of SmallInt): SmallInt
Function CheckCrc32( var X : array of Byte; N : Integer; Crc : Cardinal) : Integer;
function CheckSynchronize(Timeout: Integer): Boolean
Function CheckWin32Version( AMajor : Integer; AMinor : Integer) : Boolean
function ChrA(const a: byte): char;
Function ClassIDToProgID(const ClassID: TGUID): string;
Function ClassNameIs(const Name: string): Boolean
Function ClearBit( const Value : Byte; const Bit : TBitRange) : Byte;
Function ClearBit1( const Value : Shortint; const Bit : TBitRange) : Shortint;
Function ClearBit2( const Value : Smallint; const Bit : TBitRange) : Smallint;
Function ClearBit3( const Value : Word; const Bit : TBitRange) : Word;
Function ClearBit4( const Value : Integer; const Bit : TBitRange) : Integer;
Function ClearBit5( const Value : Cardinal; const Bit : TBitRange) : Cardinal;
Function ClearBit6( const Value : Int64; const Bit : TBitRange) : Int64;
function CLIENTTOSCREEN(POINT:TPOINT):TPOINT
Function Clipboard : TClipboard
Function ClipCodes( const X, Y, MinX, MinY, MaxX, MaxY : Float) : TClipCodes;
Function ClipCodes1( const X, Y : Float; const ClipRect : TRect) : TClipCodes;
Function ClipLine( var X1, Y1, X2, Y2 : Integer; const ClipRect : TRect) : Boolean;
Function ClipLineToRect( var P1, P2 : TFloatPoint; const Rect : TFloatRect) : Boolean
Function Clone( out stm : IStream) : HResult
Function CloneConnection : TSQLConnection
Function CloneMemoryStream( Original : TMemoryStream) : TMemoryStream
function CLOSEQUERY:BOOLEAN
Function CloseVolume( var Volume : THandle) : Boolean
Function CloseHandle(Handle: Integer): Integer; stdcall;
Function CPlApplet( hwndCPl : THandle; uMsg : DWORD; lParam1, lParam2 : Longint) : Longint
Function CmdLine: PChar;
function CmdShow: Integer;
Function Color32( const R, G, B : Byte; const A : Byte) : TColor32;
Function Color32( WinColor : TColor) : TColor32;
Function Color321( const Index : Byte; const Palette : TPalette32) : TColor32;
Function ColorAdjustLuma( clrRGB : TColor; n : Integer; fScale :BOOLean) : TColor
Function ColorHLSToRGB( Hue, Luminance, Saturation : Word) : TColorRef
Function ColorToHTML( const Color : TColor) : String
function ColorToIdent(Color: Longint; var Ident: string): Boolean)
Function ColorToRGB(color: TColor): Longint
function ColorToString(Color: TColor): string)
Function ColorToWebColorName( Color : TColor) : string
Function ColorToWebColorStr( Color : TColor) : string
Function ColumnAtDepth( Col : TColumn; ADepth : Integer) : TColumn
Function Combination(npr, ncr: integer): extended;
Function CombinationInt(npr, ncr: integer): Int64;
Function CombineInfo( Bitmap : TCustomBitmap32) : TCombineInfo
Function CommaAdd( const AStr1, AStr2 : String) : string
Function CommercialRound( const X : Extended) : Int64
Function Commit( grfCommitFlags : Longint) : HResult
Function Compare( const NameExt : string) : Boolean
function CompareDate(const A, B: TDateTime): TValueRelationship;
Function CompareDateTime( const ADateTime1, ADateTime2 : TDateTime) : Integer
Function CompareFiles(const FN1,FN2 :string; Breathe:TNotifyEvent;BreathingSender:TObject): boolean
Function CompareMemoryStreams( S1, S2 : TMemoryStream) : boolean
Function CompareStr( S1, S2 : string) : Integer
function CompareStr(const S1: string; const S2: string): Integer)
function CompareString(const S1: string; const S2: string): Integer)
Function CompareText( S1, S2 : string) : Integer
function CompareText(const S1: string; const S2: string): Integer)
Function CompareTextLike(cWildStr,cStr:string;const cWildChar:char;lCaseSensitive:boolean): boolean
Function CompareTime(const A, B: TDateTime): TValueRelationship;
Function CompatibleConversionType( const AType : TConvType; const AFamily : TConvFamily) : Boolean
Function CompatibleConversionTypes( const AFrom, ATo : TConvType) : Boolean
Function ComponentTypeToString( const ComponentType : DWORD) : string
Function CompToCurrency( Value : Comp) : Currency
Function CompToDouble( Value : Comp) : Double
function ComputeFileCRC32(const FileName : String) : Integer;
function ComputeSHA256(astr: string; amode: char): string)  //mode F:File, S:String
function ComputeSHA512(astr: string; amode: char): string)  //mode F:File, S:String
Function Concat(s: string): string
Function ConnectAndGetAll : string
Function Connected : Boolean
function constrain(x, a, b: integer): integer;
Function ConstraintCallBack( Req : DsInfoReq; var ADataSources : DataSources) : DBIResult
Function ConstraintsDisabled : Boolean
function CONTAINSCONTROL(CONTROL:TCONTROL):BOOLEAN
Function ContainsState( oState : TniRegularExpressionState) : boolean
Function ContainsStr( const AText, ASubText : string) : Boolean
Function ContainsText( const AText, ASubText : string) : Boolean
Function ContainsTransition( oTransition : TniRegularExpressionTransition) : boolean
Function Content : string
Function ContentFromStream( Stream : TStream) : string
Function ContentFromString( const S : string) : string
Function CONTROLSDISABLED : BOOLEAN
Function Convert( const AValue : Double; const AFrom, ATo : TConvType) : Double;
Function Convert1( const AValue : Double; const AFrom1, AFrom2, ATo1, ATo2 : TConvType) : Double;
Function ConvertFrom( const AFrom : TConvType; const AValue : Double) : Double
Function ConvertReadStream( Stream : TStream; Buffer : PChar; BufSize : Integer) : Integer
Function ConvertTo( const AValue : Double; const ATo : TConvType) : Double
Function ConvertWriteStream( Stream : TStream; Buffer : PChar; BufSize : Integer) : Integer
Function ConvFamilyToDescription( const AFamily : TConvFamily) : string
Function ConvTypeToDescription( const AType : TConvType) : string
Function ConvTypeToFamily( const AFrom, ATo : TConvType) : TConvFamily;
Function ConvTypeToFamily( const AType : TConvType) : TConvFamily;
Function ConvAdd(const AVal:Double;const AType1:TConvType;const AVal2:Double;const AType2,
AResultType:TConvType): Double
Function ConvCompareValue(const AValue1:Double;const AType1:TConvType;const AValue2:Double;const
AType2:TConvType): TValueRelationship
Function ConvDec( const AValue : Double; const AType, AAmountType : TConvType) : Double;
Function ConvDec1(const AValue:Double; const AType: TConvType;const AAmount:Double;const
AAmountType:TConvType):Double;
Function ConvDiff(const AVal1:Double;const AType1:TConvType;const AVal2:Double;const AType2,
AResuType:TConvType):Double
```

```
Function ConvInc( const AValue : Double; const AType, AAmountType : TConvType) : Double;
Function ConvInc1(const AValue:Double;const AType:TConvType;const AAmount:Double;const AAmountType:TConvType): Double;
Function ConvSameValue(const AValue1:Double;const AType1:TConvType;const AValue2:Double;const AType2:TConvType):Boolean
Function ConvToStr( const AValue : Double; const AType : TConvType) : string
Function ConvWithinNext( const AValue, ATest : Double; const AType : TConvType; const AAmount : Double; const AAmountType : TConvType) : Boolean
Function ConvWithinPrevious(const AValue,ATest:Double;const AType:TConvType; const AAmount:Double;const AAmountType: TConvType) : Boolean
function Copy(s: AnyString; iFrom, iCount: Longint): AnyString;
Function CopyFile( Source, Dest : string; CanOverwrite : Boolean) : Boolean
Function CopyFileEx( Source, Dest : string; Flags : FILEOP_FLAGS) : Boolean
Function CopyFileTo( const Source, Destination : string) : Boolean
function CopyFrom(Source:TStream;Count:Int64):LongInt
Function CopyPalette( Palette : HPALETTE) : HPALETTE
Function CopyTo( Length : Integer) : string
Function CopyTo(stm: IStream; cb: Largeint;out cbRead: Largeint;out cbWritten:Largeint): HResult
Function CopyToEOF : string
Function CopyToEOL : string
Function Cos(e : Extended) : Extended;
Function Cosecant( const X : Extended) : Extended
Function Cot( const X : Extended) : Extended
Function Cotan( const X : Extended) : Extended
Function CotH( const X : Extended) : Extended
Function Count : Integer
Function CountBitsCleared( X : Byte) : Integer;
Function CountBitsCleared1( X : Shortint) : Integer;
Function CountBitsCleared2( X : Smallint) : Integer;
Function CountBitsCleared3( X : Word) : Integer;
Function CountBitsCleared4( X : Integer) : Integer;
Function CountBitsCleared5( X : Cardinal) : Integer;
Function CountBitsCleared6( X : Int64) : Integer;
Function CountBitsSet( X : Byte) : Integer;
Function CountBitsSet1( X : Word) : Integer;
Function CountBitsSet2( X : Smallint) : Integer;
Function CountBitsSet3( X : ShortInt) : Integer;
Function CountBitsSet4( X : Integer) : Integer;
Function CountBitsSet5( X : Cardinal) : Integer;
Function CountBitsSet6( X : Int64) : Integer;
function CountGenerations(Ancestor,Descendent: TClass): Integer
Function Coversine( X : Float) : Float
function CRC32(const fileName: string): LongWord;
Function CREATEBLOBSTREAM( FIELD : TFIELD; MODE : TBLOBSTREAMMODE) : TSTREAM
Function CreateColumns : TDBGridColumns
Function CreateDataLink : TGridDataLink
Function CreateDir( Dir : string) : Boolean
function CreateDir(const Dir: string): Boolean)
Function CreateDOSProcessRedirected( const CommandLine, InputFile, OutputFile : string) : Boolean
Function CreateEnvironmentBlock(const Options:TEnvironmentOptions;const AdditionalVars:TStrings): PChar
Function CREATEFIELD(OWNER:TCOMPONENT;PARENTFIELD:TOBJECTFIELD; const FIELDNAME:String;CREATECHILDREN:BOOLEAN):TFIELD
Function CreateGlobber( sFilespec : string) : TniRegularExpression
Function CreateGrayMappedBmp( Handle : HBITMAP) : HBITMAP
Function CreateGrayMappedRes( Instance : THandle; ResName : PChar) : HBITMAP
function CreateGUID(out Guid: TGUID): HResult)
Function CreateInstance( const unkOuter : IUnknown; const iid : TGUID; out obj) : HResult
Function CreateMappedBmp( Handle : HBITMAP; const OldColors, NewColors : array of TColor) : HBITMAP
Function CreateMappedRes(Instance:THandle;ResName:PChar;const OldColors,NewColors:array of TColor):HBITMAP
Function CreateMessageDialog(const Msg:string; DlgType:TMsgDlgType; Buttons: TMsgDlgButtons) : TForm;
Function CreateMessageDialog1(const Msg:string;DlgType:TMsgDlgType;Btns:TMsgDlgBtns;DefaultBtn:TMsgDlgBtn):TForm;
function CreateOleObject(const ClassName: String): IDispatch;
Function CREATEPARAM( FLDTYPE : TFIELDTYPE; const PARAMNAME : String; PARAMTYPE : TPARAMTYPE) : TPARAM
Function CreateParameter(const Name:WideString;DataType:TDataType;Direction:TParameterDirection;Size:Integer;Value: OleVariant):TParameter
Function CreateLocate( DataSet : TDataSet) : TJvLocateObject
Function CreateMappedBmp( Handle : HBITMAP; const OldColors, NewColors : array of TColor) : HBITMAP
Function CreateMappedRes(Instance:THandle;ResName:PChar;const OldColors,NewColors:array of TColor):HBITMAP
Function CreateRecordBuffer( Length : Integer) : TRecordBuffer
Function CreateValueBuffer( Length : Integer) : TValueBuffer
Function CreatePopupCalculator( AOwner : TComponent; ABiDiMode : TBiDiMode) : TWinControl
Function CreateRecordBuffer( Length : Integer) : TRecordBuffer
Function CreateRotatedFont( Font : TFont; Angle : Integer) : HFONT
Function CreateTwoColorsBrushPattern( Color1, Color2 : TColor) : TBitmap
Function CreateValueBuffer( Length : Integer) : TValueBuffer
Function CreateHexDump( AOwner : TWinControl) : THexDump
Function Csc( const X : Extended) : Extended
Function CscH( const X : Extended) : Extended
function currencyDecimals: Byte
function currencyFormat: Byte
function currencyString: String
Function CurrentProcessId : TIdPID
Function CurrentReadBuffer : string
Function CurrentThreadId : TIdPID
Function CurrentYear : Word
Function CurrToBCD(const Curr: Currency; var BCD: TBcd; Precision: Integer; Decimals: Integer): Boolean
Function CurrToStr( Value : Currency) : string;
Function CurrToStrF( Value : Currency; FormatSettings : TFormatSettings; Digits: Integer) : string;
Function CurrToStrFS(Value:Currency;Format:TFloatFormat;Digits:Integer;const
FormatSettings:TFormatSettings):string;
function CursorToIdent(cursor: Longint; var Ident: string): Boolean;
function CursorToString(cursor: TCursor): string;
Function CustomSort( SortProc : TLVCompare; lParam : Longint) : Boolean
Function CustomSort( SortProc : TTVCompare; Data : Longint; ARecurse : Boolean) : Boolean
Function CycleToDeg( const Cycles : Extended) : Extended
Function CycleToGrad( const Cycles : Extended) : Extended
Function CycleToRad( const Cycles : Extended) : Extended
Function D2H( N : Longint; A : Byte) : string
Function DarkColor( const Color : TColor; const Pct : Single) : TColor
Function DarkColorChannel( const Channel : Byte; const Pct : Single) : Byte
Function DataLinkDir : string
Function DataRequest( Data : OleVariant) : OleVariant
Function DataRequest( Input : OleVariant) : OleVariant
Function DataToRawColumn( ACol : Integer) : Integer
Function Date : TDateTime
function Date: TDateTime;
Function DateIsNull( const pdtValue : TDateTime; const pdtKind : TdtKind) : Boolean
Function DateOf( const AValue : TDateTime) : TDateTime
function DateSeparator: char;
Function DateTimeGMTToHttpStr( const GMTValue : TDateTime) : String
Function DateTimeToFileDate( DateTime : TDateTime) : Integer
function DateTimeToFileDate(DateTime: TDateTime): Integer;
Function DateTimeToGmtOffSetStr( ADateTime : TDateTime; SubGMT : Boolean) : string
Function DateTimeToInternetStr( const Value : TDateTime; const AIsGMT : Boolean) : String
Function DateTimeToJulianDate( const AValue : TDateTime) : Double
Function DateTimeToModifiedJulianDate( const AValue : TDateTime) : Double
Function DateTimeToStr( DateTime : TDateTime) : string;
Function DateTimeToStr2( DateTime : TDateTime; FormatSettings : TFormatSettings) : string;
function DateTimeToTimeStamp(DateTime: TDateTime): TTimeStamp
Function DateTimeToUnix( const AValue : TDateTime) : Int64
function DateTimeToUnix(D: TDateTime): Int64;
Function DateToStr( DateTime : TDateTime) : string;
function DateToStr(const DateTime: TDateTime): string;
function DateToStr(D: TDateTime): string;
Function DateToStr2( DateTime : TDateTime; FormatSettings : TFormatSettings) : string;
Function DayOf( const AValue : TDateTime) : Word
Function DayOfTheMonth( const AValue : TDateTime) : Word
function DayOfTheMonth(const AValue: TDateTime): Word;
Function DayOfTheWeek( const AValue : TDateTime) : Word
Function DayOfTheYear( const AValue : TDateTime) : Word
function DayOfTheYear(const AValue: TDateTime): Word;
Function DayOfWeek( DateTime : TDateTime) : Word
function DayOfWeek(const DateTime: TDateTime): Word;
Function DayOfWeekStr( DateTime : TDateTime) : string
Function DaysBetween( const ANow, AThen : TDateTime) : Integer
Function DaysInAMonth( const AYear, AMonth : Word) : Word
Function DaysInAYear( const AYear : Word) : Word
Function DaysInMonth( const AValue : TDateTime) : Word
Function DaysInYear( const AValue : TDateTime) : Word
Function DaySpan( const ANow, AThen : TDateTime) : Double
Function DBUseRightToLeftAlignment( AControl : TControl; AField : TField) : Boolean
function DecimalSeparator: char;
Function DecLimit( var B : Byte; const Limit : Byte; const Decr : Byte) : Byte;
Function DecLimit1( var B : Shortint; const Limit : Shortint; const Decr : Shortint) : Shortint;
Function DecLimit2( var B : Smallint; const Limit : Smallint; const Decr : Smallint) : Smallint;
Function DecLimit3( var B : Word; const Limit : Word; const Decr : Word) : Word;
Function DecLimit4( var B : Integer; const Limit : Integer; const Decr : Integer) : Integer;
Function DecLimit5( var B : Cardinal; const Limit : Cardinal; const Decr : Cardinal) : Cardinal;
Function DecLimit6( var B : Int64; const Limit : Int64; const Decr : Int64) : Int64;
Function DecLimitClamp( var B : Byte; const Limit : Byte; const Decr : Byte) : Byte;
Function DecLimitClamp1( var B : Shortint; const Limit : Shortint; const Decr : Shortint) : Shortint;
Function DecLimitClamp2( var B : Smallint; const Limit : Smallint; const Decr : Smallint) : Smallint;
Function DecLimitClamp3( var B : Word; const Limit : Word; const Decr : Word) : Word;
Function DecLimitClamp4( var B : Integer; const Limit : Integer; const Decr : Integer) : Integer;
Function DecLimitClamp5( var B : Cardinal; const Limit : Cardinal; const Decr : Cardinal) : Cardinal;
Function DecLimitClamp6( var B : Int64; const Limit : Int64; const Decr : Int64) : Int64;
Function DecodeDateFully( DateTime : TDateTime; var Year, Month, Day, DOW : Word) : Boolean
Function DecodeSoundexInt( AValue : Integer) : string
Function DecodeSoundexWord( AValue : Word) : string
Function DefaultAlignment : TAlignment
Function DefaultCaption : string
Function DefaultColor : TColor
Function DefaultFont : TFont
Function DefaultImeMode : TImeMode
Function DefaultImeName : TImeName
Function DefaultReadOnly : Boolean
Function DefaultWidth : Integer
Function DegMinSecToFloat( const Degs, Mins, Secs : Float) : Float
Function DegToCycle( const Degrees : Extended) : Extended
Function DegToGrad( const Degrees : Extended) : Extended
Function DegToGrad( const Value : Extended) : Extended;
Function DegToGrad1( const Value : Double) : Double;
Function DegToGrad2( const Value : Single) : Single;
Function DegToRad( const Degrees : Extended) : Extended
Function DegToRad( const Value : Extended) : Extended;
Function DegToRad1( const Value : Double) : Double;
Function DegToRad2( const Value : Single) : Single;
Function DelChar( const pStr : string; const pChar : Char) : string
Function DelEnvironmentVar( const Name : string) : Boolean
Function Delete( const MsgNum : Integer) : Boolean
Function DeleteDirectory( const DirectoryName : string; MoveToRecycleBin : Boolean) : Boolean
Function DeleteFile(const FileName: string): boolean)
Function DeleteFileEx( FileName : string; Flags : FILEOP_FLAGS) : Boolean
Function DelimiterPosn( const sString : string; const sDelimiters: string): integer;
Function DelimiterPosn1(const sString:string;const sDelimiters:string;out cDelimiter: char) : integer;
Function DelSpace( const pStr : string) : string
Function DelString( const pStr, pDelStr : string) : string
Function DelTree( const Path : string) : Boolean
Function Depth : Integer
Function Description : string
Function DescriptionsAvailable : Boolean
Function DescriptionToConvFamily( const ADescription : string; out AFamily : TConvFamily) : Boolean
Function DescriptionToConvType( const ADescription : string; out AType : TConvType) : Boolean;
Function DescriptionToConvType1(const AFamil:TConvFamily;const ADescr:string;out AType:TConvType):Boolean;
Function DetectUTF8Encoding( const s : UTF8String) : TEncodeType
Function DialogsToPixelsX( const Dialogs : Word) : Word
Function DialogsToPixelsY( const Dialogs : Word) : Word
Function Digits( const X : Cardinal) : Integer
Function DirectoryExists( const Name : string) : Boolean
Function DirectoryExists( Directory : string) : Boolean
Function DiskFree( Drive : Byte) : Int64
function DiskFree(Drive: Byte): Int64)
Function DiskInDrive( Drive : Char) : Boolean
Function DiskSize( Drive : Byte) : Int64
function DiskSize(Drive: Byte): Int64)
Function DISPATCHCOMMAND( ACOMMAND : WORD) : BOOLEAN
Function DispatchEnabled : Boolean
Function DispatchMask : TMask
Function DispatchMethodType : TMethodType
Function DISPATCHPOPUP( AHANDLE : HMENU) : BOOLEAN
Function DispatchRequest( Sender : TObject; Request : TWebRequest; Response : TWebResponse) : Boolean
Function DisplayCase( const S : String) : String
Function DisplayRect( Code : TDisplayCode) : TRect
Function DisplayRect( TextOnly : Boolean) : TRect
Function DisplayStream( Stream : TStream) : string
TBufferCoord', 'record Char : integer; Line : integer; end
TDisplayCoord', 'record Column : integer; Row : integer; end
Function DisplayCoord( AColumn, ARow : Integer) : TDisplayCoord
Function BufferCoord( AChar, ALine : Integer) : TBufferCoord
Function DomainName( const AHost : String) : String
Function DosPathToUnixPath( const Path : string) : string
Function DottedLineTo( const Canvas : TCanvas; const X, Y : Integer) : Boolean;
Function DoubleDecliningBalance( const Cost, Salvage : Extended; Life, Period : Integer) : Extended
Function DoubleToBcd( const AValue : Double) : TBcd;
Function DoubleToHex( const D : Double) : string
Function DoUpdates : Boolean
function Dragging: Boolean;
Function DrawCaption( p1 : HWND; p2 : HDC; const p3 : TRect; p4 : UINT) : BOOL
Function DrawAnimatedRects( hwnd : HWND; idAni : Integer; const lprcFrom, lprcTo : TRect) : BOOL
Function DrawEdge( hdc : HDC; var qrc : TRect; edge : UINT; grfFlags : UINT) : BOOL
Function DrawFrameControl( DC : HDC; const Rect : TRect; uType, uState : UINT) : BOOL
{Works like InputQuery but displays 2edits. If PasswordChar <> #0, the second edit's PasswordChar is set}
Function DualInputQuery(const ACapt,Prpt1,Prpt2:string;var AVal1,AVal2:string;PasswrdChar:Char= #0):Bool;
Function DupeString( const AText : string; ACount : Integer) : string
Function Edit : Boolean
Function EditCaption : Boolean
Function EditText : Boolean
Function EditFolderList( Folders : TStrings) : Boolean
Function EditQueryParams( DataSet : TDataSet; List : TParams; AHelpContext : THelpContext) : Boolean
Function Elapsed( const Update : Boolean) : Cardinal
Function EnableProcessPrivilege( const Enable : Boolean; const Privilege : string) : Boolean
Function EnableThreadPrivilege( const Enable : Boolean; const Privilege : string) : Boolean
Function EncodeDate( Year, Month, Day : Word) : TDateTime
function EncodeDate(Year, Month, Day: Word): TDateTime;
Function EncodeDateDay( const AYear, ADayOfYear : Word) : TDateTime
Function EncodeDateMonthWeek( const AYear, AMonth, AWeekOfMonth, ADayOfWeek : Word) : TDateTime
Function EncodeDateTime(const AYear,AMonth,ADay,AHour,AMinute,ASecond,AMilliSecond: Word): TDateTime
Function EncodeDateWeek( const AYear, AWeekOfYear : Word; const ADayOfWeek : Word) : TDateTime
Function EncodeDayOfWeekInMonth( const AYear, AMonth, ANthDayOfWeek, ADayOfWeek : Word) : TDateTime
Function EncodeString( s : string) : string
Function DecodeString( s : string) : string
Function EncodeTime( Hour, Min, Sec, MSec : Word) : TDateTime
function EncodeTime(Hour, Min, Sec, MSec: Word): TDateTime;
Function EndIP : String
Function EndOfADay( const AYear, AMonth, ADay : Word) : TDateTime;
Function EndOfADay1( const AYear, ADayOfYear : Word) : TDateTime;
Function EndOfAMonth( const AYear, AMonth : Word) : TDateTime
Function EndOfAWeek( const AYear, AWeekOfYear : Word; const ADayOfWeek : Word) : TDateTime
Function EndOfAYear( const AYear : Word) : TDateTime
Function EndOfTheDay( const AValue : TDateTime) : TDateTime
Function EndOfTheMonth( const AValue : TDateTime) : TDateTime
Function EndOfTheWeek( const AValue : TDateTime) : TDateTime
Function EndOfTheYear( const AValue : TDateTime) : TDateTime
Function EndPeriod( const Period : Cardinal) : Boolean
Function EndsStr( const ASubText, AText : string) : Boolean
Function EndsText( const ASubText, AText : string) : Boolean
Function EnsureMsgIDBrackets( const AMsgID : String) : String
Function EnsureRange( const AValue, AMin, AMax : Integer) : Integer;
Function EnsureRange1( const AValue, AMin, AMax : Int64) : Int64;
Function EnsureRange2( const AValue, AMin, AMax : Double) : Double;
Function EOF: boolean
Function EOln: boolean
Function EqualRect( const R1, R2 : TRect) : Boolean
function EqualRect(const R1, R2: TRect): Boolean)
Function Equals( Strings : TWideStrings) : Boolean
function Equals(Strings: TStrings): Boolean;
Function EqualState( oState : TniRegularExpressionState) : boolean
Function ErrOutput: Text)
function ExceptionParam: String;
function ExceptionPos: Cardinal;
function ExceptionProc: Cardinal;
function ExceptionToString(er: TIFException; Param: String): String;
function ExceptionType: TIFException;
Function ExcludeTrailingBackslash( S : string) : string
function ExcludeTrailingBackslash(const S: string): string)
Function ExcludeTrailingPathDelimiter( const APath : string) : string
Function ExcludeTrailingPathDelimiter( S : string) : string
function ExcludeTrailingPathDelimiter(const S: string): string)
function ExecConsoleApp(const AppName, Parameters: String; AppOutput: TStrings): boolean;
Function ExecProc : Integer
Function ExecSQL : Integer
Function ExecSQL( ExecDirect : Boolean) : Integer
Function Execute : _Recordset;
Function Execute : Boolean
Function Execute : Boolean;
Function Execute( const SQL : string; Params : TParams; Cache : Boolean; cursor : phDBICur) : Integer
Function Execute( const SQL : WideString; Params : TParams; ResultSet : TPSResult) : Integer
Function Execute( ParentWnd : HWND) : Boolean
Function Execute1(constCommText:WideString;const CType:TCommType;const
ExecuteOpts:TExecuteOpts):_Recordset;
Function Execute1( const Parameters : OleVariant) : _Recordset;
Function Execute1( ParentWnd : HWND) : Boolean;
Function Execute2( var RecordsAffected : Integer; const Parameters : OleVariant) : _Recordset;
Function ExecuteAction( Action : TBasicAction) : Boolean
Function ExecuteDirect( const SQL : WideString) : Integer
Function ExecuteFile(const FileName:string;const Params:string;const DefDir:string;ShowCmd:Int):THandle
Procedure ExecuteThread2(afunc:TThreadFunction2;thrOK:boolean);AddTypeS('TThreadFunction2','procedure
Function CreateThread2(ThreadFunc: TThreadFunction2) : THandle
function ExeFileIsRunning(ExeFile: string): boolean;
function ExePath: string;
function ExePathName: string;
Function Exists( AItem : Pointer) : Boolean
Function ExitWindows( ExitCode : Cardinal) : Boolean
function Exp(x: Extended): Extended;
Function ExpandEnvironmentVar( var Value : string) : Boolean
Function ExpandFileName( FileName : string) : string
function ExpandFileName(const FileName: string): string)
Function ExpandUNCFileName( FileName : string) : string
function ExpandUNCFileName(const FileName: string): string)
Function ExpJ( const X : Float) : Float;
Function Exsecans( X : Float) : Float
Function Extract( const AByteCount : Integer) : string
Function Extract( Item : TClass) : TClass
Function Extract( Item : TComponent) : TComponent
Function Extract( Item : TObject) : TObject
Function ExtractFileDir( FileName : string) : string
function ExtractFileDir(const FileName: string): string)
Function ExtractFileDrive( FileName : string) : string
function ExtractFileDrive(const FileName: string): string)
Function ExtractFileExt( FileName : string) : string
function ExtractFileExt(const FileName: string): string)
Function ExtractFileExtNoDot( const FileName : string) : string
Function ExtractFileExtNoDotUpper( const FileName : string) : string
Function ExtractFileName( FileName : string) : string
function ExtractFileName(const filename: string):string;
Function ExtractFilePath( FileName : string) : string
function ExtractFilePath(const filename: string):string;
Function ExtractRelativePath( BaseName, DestName : string) : string
function ExtractRelativePath(const BaseName: string; const DestName: string): string)
Function ExtractShortPathName( FileName : string) : string
function ExtractShortPathName(const FileName: string): string)
function ExtractStrings(Separators, WhiteSpace: TSysCharSet; Content: PChar;Strings: TStrings): Integer
function ExtractStrings(Separators:TSysCharSet;WhiteSpace:TSysCharSet;Content:PChar;Str:TStrings): Integer)
Function Fact(numb: integer): Extended;
Function FactInt(numb: integer): int64;
Function Factorial( const N : Integer) : Extended
Function FahrenheitToCelsius( const AValue : Double) : Double
function FalseBoolStrs: array of string
Function Fetch(var AInput:string;const ADelim:string;const ADelete:Bool;const ACaseSensitive:Bool):string
Function FetchCaseInsensitive(var AInput:string; const ADelim:string; const ADelete:Boolean): string
Function Fibo(numb: integer): Extended;
Function FiboInt(numb: integer): Int64;
Function Fibonacci( const N : Integer) : Integer
Function FIELDBYNAME( const FIELDNAME : STRING) : TFIELD
Function FIELDBYNAME( const FIELDNAME : WIDESTRING) : TFIELD
Function FIELDBYNAME( const NAME : String) : TFIELD
Function FIELDBYNAME( const NAME : String) : TFIELDDEF
Function FIELDBYNUMBER( FIELDNO : INTEGER) : TFIELD
Function FileAge( FileName : string) : Integer
Function FileAge(const FileName: string): integer)
Function FileCompareText( const A, B : String) : Integer
Function FileContains(const FileName:string;Text:string;CaseSensitive:Bool;ExceptionOnError:Bool): Boolean
Function FileCreate( FileName : string) : Integer;
Function FileCreate(const FileName: string): integer)
Function FileCreateTemp( var Prefix : string) : THandle
Function FileDateToDateTime( FileDate : Integer) : TDateTime
function FileDateToDateTime(FileDate: Integer): TDateTime;
Function FileExists( const FileName : string) : Boolean
Function FileExists( FileName : string) : Boolean
function fileExists(const FileName: string): Boolean;
Function FileGetAttr( FileName : string) : Integer
Function FileGetAttr(const FileName: string): integer)
Function FileGetDate( Handle : Integer) : Integer
Function FileGetDate(handle: integer): integer
Function FileGetDisplayName( const FileName : string) : string
Function FileGetSize( const FileName : string) : Integer
Function FileGetTempName( const Prefix : string) : string
Function FileGetTypeName( const FileName : string) : string
Function FileIsReadOnly( FileName : string) : Boolean
Function FileLoad( ResType : TResType; const Name : string; MaskColor : TColor) : Boolean
Function FileOpen( FileName : string; Mode : LongWord) : Integer
Function FileOpen(const FileName: string; mode:integer): integer)
Function FileRead(handle: integer; Buffer: PChar; count: LongWord): integer
Function FileSearch( Name, DirList : string) : string
Function FileSearch(const Name, dirList: string): string)
Function FileSeek( Handle : Integer; Offset : Int64; Origin : Integer) : Int64;
Function FileSeek( Handle, Offset, Origin : Integer) : Integer;
Function FileSeek(handle, offset, origin: integer): integer
Function FileSetAttr( FileName : string; Attr : Integer) : Integer
function FileSetAttr(const FileName: string; Attr: Integer): Integer)
Function FileSetDate(FileName : string; Age : Integer) : Integer;
Function FileSetDate(handle: integer; age: integer): integer
Function FileSetDate2(FileHandle : Integer; Age : Integer) : Integer;
Function FileSetDateH( Handle : Integer; Age : Integer) : Integer;
Function FileSetReadOnly( FileName : string; ReadOnly : Boolean) : Boolean
Function FileSize( const FileName : string) : int64
Function FileSizeByName( const AFilename : string) : Longint
function FileWrite(Handle: Integer; const Buffer: pChar; Count: LongWord): Integer)
Function FilterSpecArray : TComdlgFilterSpecArray
Function FIND( ACAPTION : String) : TMENUITEM
Function Find( AItem : Pointer; out AData : Pointer) : Boolean
Function FIND( const ANAME : String) : TNAMEDITEM
Function Find( const DisplayName : string) : TAggregate
Function Find( const Item : TBookmarkStr; var Index : Integer) : Boolean
Function FIND( const NAME : String) : TFIELD
Function FIND( const NAME : String) : TFIELDDEF
Function FIND( const NAME : String) : TINDEXDEF
Function Find( const S : WideString; var Index : Integer) : Boolean
function Find(S:String;var Index:Integer):Boolean
Function FindAuthClass( AuthName : String) : TIdAuthenticationClass
Function FindBand( AControl : TControl) : TCoolBand
Function FindBoundary( AContentType : string) : string
Function FindButton( AModalResult : TModalResult) : TTaskDialogBaseButtonItem
Function FindCaption(StartIndex: Integer;Value: string; Partial,Inclusive,Wrap: Boolean): TListItem
Function FindCdLineSwitch( Switch : string; IgnoreCase : Boolean) : Boolean;
Function FindCloseW(FindFile: Integer): LongBool; stdcall;
Function FindCmdLineSwitch( Switch : string; Chars : TSysCharSet; IgnoreCase : Boolean) : Boolean;
Function FindCmmdLineSwitch( Switch : string) : Boolean;
function FindComponent(AName: String): TComponent;
function FindComponent(vlabel: string): TComponent;
function FindComponent2(vlabel: string): TComponent;
function FindControl(Handle: HWnd): TWinControl;
Function FindData( StartIndex : Integer; Value : Pointer; Inclusive, Wrap : Boolean) : TListItem
Function FindDatabase( const DatabaseName : string) : TDatabase
function FindDragTarget(const Pos: TPoint; AllowDisabled: Boolean): TControl;
Function FINDFIELD( const FIELDNAME : STRING) : TFIELD
Function FINDFIELD( const FIELDNAME : WideString) : TFIELD
Function FindFirst2(const Path: string; Attr: Integer; var F: TSearchRec):Integer)
Function FindNext2(var F: TSearchRec): Integer)
procedure FindClose2(var F: TSearchRec)
Function FINDFIRST : BOOLEAN
TJvSpecialFolder = (sfRecycleBin, sfControlPanel, sfDesktop, sfDesktopDirectory,
  sfMyComputer, sfFonts, sfNetHood, sfNetwork, sfPersonal, sfPrinters, sfPrograms,sfRecent,sfSendTo, sfStartMenu, stStartUp, sfTemplates);
 FFolder: array [TJvSpecialFolder] of Integer =
    (CSIDL_BITBUCKET, CSIDL_CONTROLS, CSIDL_DESKTOP, CSIDL_DESKTOPDIRECTORY,
     CSIDL_DRIVES, CSIDL_FONTS, CSIDL_NETHOOD, CSIDL_NETWORK, CSIDL_PERSONAL,
     CSIDL_PRINTERS, CSIDL_PROGRAMS, CSIDL_RECENT, CSIDL_SENDTO, CSIDL_STARTMENU,
     CSIDL_STARTUP, CSIDL_TEMPLATES);
Function FindFilesDlg(StartIn: string; SpecialFolder: TJvSpecialFolder; UseFolder: Boolean): Boolean);
function Findfirst(const filepath: string; attr: integer): integer;
function FindFirst2(const Path: string; Attr: Integer; var F: TSearchRec): Integer)
Function FindFirstNotOf( AFind, AText : String) : Integer
Function FindFirstOf( AFind, AText : String) : Integer
Function FindImmediateTransitionOn( cChar : char) : TniRegularExpressionState
Function FINDINDEXFORFIELDS( const FIELDS : String) : TINDEXDEF
Function FindInstanceOf( AClass : TClass; AExact : Boolean; AStartAt : Integer) : Integer
Function FINDITEM( VALUE : INTEGER; KIND : TFINDITEMKIND) : TMENUITEM
function FindItemId( Id : Integer) : TCollectionItem
Function FindKey( const KeyValues : array of const) : Boolean
Function FINDLAST : BOOLEAN
Function FindLineControl( ComponentType, ControlType : DWORD) : TJclMixerLineControl
Function FindModuleClass( AClass : TComponentClass) : TComponent
Function FindModuleName( const AClass : string) : TComponent
Function FINDNEXT : BOOLEAN
function FindNext: integer;
function FindNext2(var F: TSearchRec): Integer)
Function FindNextPage( CurPage : TTabSheet; GoForward, CheckTabVisible : Boolean) : TTabSheet
Function FindNextToSelect : TTreeNode
Function FINDPARAM( const VALUE : String) : TPARAM
Function FindParam( const Value : WideString) : TParameter
Function FINDPRIOR : BOOLEAN
Function FindResource( ModuleHandle : HMODULE; ResourceName, ResourceType : PChar) : TResourceHandle
Function FindSession( const SessionName : string) : TSession
function FindStringResource(Ident: Integer): string)
Function FindText( const SearchStr:string;StartPos,Length: Integer; Options: TSearchTypes):Integer
Function FindUnusedFileName( const FileName, FileExt, Suffix : AnsiString) : AnsiString
function FindVCLWindow(const Pos: TPoint): TWinControl;
function FindWindow(C1, C2: PChar): Longint;
Function FindInPaths(const fileName,paths: String): String;
Function Finger : String
Function First : TClass
Function First : TComponent
Function First : TObject
Function FirstDelimiter( const delimiters : string; const Str : String) : integer;
Function FirstDelimiter1( const delimiters : WideString; const Str : WideString) : integer;
Function FirstInstance( const ATitle : string) : Boolean
Function FloatPoint( const X, Y : Float) : TFloatPoint;
Function FloatPoint1( const P : TPoint) : TFloatPoint;
Function FloatPtInRect( const Rect : TFloatRect; const P : TFloatPoint) : Boolean
Function FloatRect( const ALeft, ATop, ARight, ABottom : Double) : TFloatRect;
Function FloatRect1( const Rect : TRect) : TFloatRect;
Function FloatsEqual( const X, Y : Float) : Boolean
Function FloatToBin(const D: Double): string;     //doubletohex -> hextobin! in buffer
Function FloatToCurr( Value : Extended) : Currency
Function FloatToDateTime( Value : Extended) : TDateTime
Function FloatToStr( Value : Extended) : string;
Function FloatToStr(e : Extended) : String;
Function FloatToStrF( Value : Extended; Format : TFloatFormat; Precision, Digits : Integer) : string;
function FloatToStrF(Value: Extended; Format: TFloatFormat;Precision:Integer; Digits: Integer): string)
Function FloatToStr2( Value : Extended; Format : TFloatFormat; Precision, Digits : Integer; FormatSettings
: TFormatSettings) : string;
Function FloatToStrFS( Value : Extended; Format : TFloatFormat; Precision, Digits : Integer;
FormatSettings : TFormatSettings) : string;
function FloatToText(BufferArg: PChar; const Value: Extended; ValueType: TFloatValue;Format: TFloatFormat;
Precision,Digits: Integer): Integer)
Function Floor( const X : Extended) : Integer
Function FloorInt( Value : Integer; StepSize : Integer) : Integer
Function FloorJ( const X : Extended) : Integer
Function Flush( const Count : Cardinal) : Boolean
Function Flush(var t: Text): Integer
function FmtLoadStr(Ident: Integer; const Args: array of const): string)
function FOCUSED:BOOLEAN
Function ForceBackslash( const PathName : string) : string
Function ForceDirectories( const Dir : string) : Boolean
Function ForceDirectories( Dir : string) : Boolean
Function ForceDirectories( Name : string) : Boolean
Function ForceInBox( const Point : TPoint; const Box : TRect) : TPoint
Function ForceInRange( A, Min, Max : Integer) : Integer
Function ForceInRangeR( const A, Min, Max : Double) : Double
Function ForEach( AProc : TBucketProc; AInfo : Pointer) : Boolean;
Function ForEach1( AEvent : TBucketEvent) : Boolean;
Function ForegroundTask: Boolean
function Format(const Format: string; const Args: array of const): string;
Function FormatBcd( const Format : string; Bcd : TBcd) : string
FUNCTION FormatBigInt(s: string): STRING;
function FormatBuf(var Buffer:PChar;BufLen:Card;const Format:string;FmtLen:Cardinal;const Args:array of
const):Cardinal
Function FormatCount( iCount : integer; const sSingular : string; const sPlural : string) : string
Function FormatCurr( Format : string; Value : Currency) : string;
function FormatCurr(const Format: string; Value: Currency): string)
Function FormatDateTime( Format : string; DateTime : TDateTime) : string;
function FormatDateTime(const fmt: string; D: TDateTime): string;
Function FormatFloat( Format : string; Value : Extended) : string;
function FormatFloat(const Format: string; Value: Extended): string)
Function FormatFloat( Format : string; Value : Extended) : string;
Function FormatFloat2( Format : string; Value : Extended; FormatSettings : TFormatSettings) : string;
Function FormatCurr( Format : string; Value : Currency) : string;
Function FormatCurr2(Format: string; Value : Currency; FormatSettings : TFormatSettings) : string;
Function Format2(const Format:string;const Args:array of const;const FSettings:TFormatSettings): string
FUNCTION FormatInt(i: integer): STRING;
FUNCTION FormatInt64(i: int64): STRING;
Function FormatMaskText( const EditMask : string; const Value : string) : string
Function FormatValue( AValue : Cardinal) : string
Function FormatVersionString( const HiV, LoV : Word) : string;
Function FormatVersionString1( const Major, Minor, Build, Revision : Word) : string;
function Frac(X: Extended): Extended);
Function FreeResource( ResData : HGLOBAL) : LongBool
Function FromCommon( const AValue : Double) : Double
function FromCommon(const AValue: Double): Double;
Function FTPGMTDateTimeToMLS( const ATimeStamp : TDateTime; const AIncludeMSecs : Boolean) : String
Function FTPLocalDateTimeToMLS( const ATimeStamp : TDateTime; const AIncludeMSecs : Boolean) : String
Function FTPMLSToGMTDateTime( const ATimeStamp : String) : TDateTime
Function FTPMLSToLocalDateTime( const ATimeStamp : String) : TDateTime
Function FuncIn(AValue: Variant; ASet: Variant): Boolean;
//Function Funclist Size is: 6444 of mX3.9.8.9
Function FutureValue(const Rate:Extended;NPeriods:Integer;const Payment,PresentValue:Extended;PaymentTime:
TPaymentTime):Extended
Function Gauss( const x, Spread : Double) : Double
function Gauss(const x,Spread: Double): Double;
Function GCD(x, y : LongInt) : LongInt;
Function GCDJ( X, Y : Cardinal) : Cardinal
Function GDAL: LongWord
Function GdiFlush : BOOL
Function GdiSetBatchLimit( Limit : DWORD) : DWORD
Function GdiGetBatchLimit : DWORD
Function GenerateHeader : TIdHeaderList
Function GeometricMean( const X : TDynFloatArray) : Float
Function Get( AURL : string) : string;
Function Get2( AURL : string) : string;
Function Get8087CW : Word
function GetActiveOleObject(const ClassName: String): IDispatch;
Function GetAliasDriverName( const AliasName : string) : string
Function GetAPMBatteryFlag : TAPMBatteryFlag
Function GetAPMBatteryFullLifeTime : DWORD
Function GetAPMBatteryLifePercent : Integer
Function GetAPMBatteryLifeTime : DWORD
Function GetAPMLineStatus : TAPMLineStatus
Function GetAppdataFolder : string
Function GetAppDispatcher : TComponent
function GetArrayLength: integer;
Function GetASCII: string;
Function GetASCIILine: string;
Function GetAsHandle( Format : Word) : THandle
function GetAssociatedProgram(const Extension: string; var Filename, Description: string): boolean;
Function GetBackupFileName( const FileName : string) : string
Function GetBBitmap( Value : TBitmap) : TBitmap
Function GetBIOSCopyright : string
Function GetBIOSDate : TDateTime
Function GetBIOSExtendedInfo : string
Function GetBIOSName : string
Function getBitmap(apath: string): TBitmap;
Function GetBitmap( Index : Integer; Image : TBitmap) : Boolean  //object
Function getBitMapObject(const bitmappath: string): TBitmap;
Function GetButtonState( Button : TPageScrollerButton) : TPageScrollerButtonState
Function GetCapsLockKeyState : Boolean
function GetCaptureControl: TControl;
Function GetCDAudioTrackList( var TrackList : TJclCdTrackInfoArray; Drive : Char) : TJclCdTrackInfo;
Function GetCDAudioTrackList1( TrackList : TStrings; IncludeTrackType : Boolean; Drive : Char) : string;
Function GetCdInfo( InfoType : TJclCdMediaInfo; Drive : Char) : string
Function GetChangedText( const Text : string; SelStart, SelLength : Integer; Key : Char) : string
Function GetClientThread( ClientSocket : TServerClientWinSocket) : TServerClientThread
Function GetClockValue : Int64
function getCmdLine: PChar;
function getCmdShow: Integer;
function GetCPUSpeed: Double;
Function GetColField( DataCol : Integer) : TField
Function GetColorBlue( const Color : TColor) : Byte
Function GetColorFlag( const Color : TColor) : Byte
Function GetColorGreen( const Color : TColor) : Byte
Function GetColorRed( const Color : TColor) : Byte
Function GetComCtlVersion : Integer
Function GetComPorts: TStringlist;
Function GetCommonAppdataFolder : string
Function GetCommonDesktopdirectoryFolder : string
Function GetCommonFavoritesFolder : string
Function GetCommonFilesFolder : string
Function GetCommonProgramsFolder : string
Function GetCommonStartmenuFolder : string
Function GetCommonStartupFolder : string
Function GetComponent( Owner, Parent : TComponent) : TComponent
Function GetConnectionRegistryFile( DesignMode : Boolean) : string
Function GetCookiesFolder : string
Function GetCPUSpeed( var CpuSpeed : TFreqInfo) : Boolean
Function GetCurrent : TFavoriteLinkItem
Function GetCurrent : TListItem
Function GetCurrent : TTaskDialogBaseButtonItem
Function GetCurrent : TToolButton
Function GetCurrent : TTreeNode
Function GetCurrent : WideString
Function GetCurrentDir : string
function GetCurrentDir: string)
```

```
Function GetCurrentFolder : string
Function GETCURRENTRECORD( BUFFER : PCHAR) : BOOLEAN
Function GetCurrentProcessId : TIdPID
Function GetCurrentThreadHandle : THandle
Function GetCurrentThreadID: LongWord; stdcall;
Function GetCustomHeader( const Name : string) : String
Function GetDataItem( Value : Pointer) : Longint
Function GetDataLinkFiles( FileNames : TWideStrings; Directory : string) : Integer;
Function GetDataLinkFiles1( FileNames : TStrings; Directory : string) : Integer;
Function GETDATASIZE : INTEGER
Function GetDC(hdwnd: HWND): HDC;
Function GetDefaultFileExt( const MIMEType : string) : string
Function GetDefaults : Boolean
Function GetDefaultSchemaName : WideString
Function GetDefaultStreamLoader : IStreamLoader
Function GetDesktopDirectoryFolder : string
Function GetDesktopFolder : string
Function GetDFAState( oStates : TList) : TniRegularExpressionState
Function GetDirectorySize( const Path : string) : Int64
Function GetDisplayWidth : Integer
Function GetDLLVersion( const DLLName : string; var pdwMajor, pdwMinor : Integer) : Boolean
Function GetDomainName : string
Function GetDriverRegistryFile( DesignMode : Boolean) : string
function GetDriveType(rootpath: pchar): cardinal;
Function GetDriveTypeStr( const Drive : Char) : string
Function GetEnumerator : TFavoriteLinkItemsEnumerator
Function GetEnumerator : TListItemsEnumerator
Function GetEnumerator : TTaskDialogButtonsEnumerator
Function GetEnumerator : TToolBarEnumerator
Function GetEnumerator : TTreeNodesEnumerator
Function GetEnumerator : TWideStringsEnumerator
Function GetEnvVar( const VarName : string) : string
Function GetEnvironmentVar( const AVariableName : string) : string
Function GetEnvironmentVariable( const VarName : string) : string
Function GetEnvironmentVar( const Name : string; var Value : string; Expand : Boolean) : Boolean
Function GetEnvironmentVars( const Vars : TStrings; Expand : Boolean) : Boolean
Function getEnvironmentString: string;
Function GetExceptionHandler : TObject
Function GetFavoritesFolder : string
Function GetFieldByName( const Name : string) : string
Function GetFieldInfo( const Origin : Widestring; var FieldInfo : TFieldInfo) : Boolean
Function GetFieldValue( ACol : Integer) : string
Function GetFileAgeCoherence( const FileName : string) : Boolean
Function GetFileCreation( const FileName : string) : TFileTime
Function GetFileCreationTime( const Filename : string) : TDateTime
Function GetFileInformation( const FileName : string) : TSearchRec
Function GetFileLastAccess( const FileName : string) : TFileTime
Function GetFileLastWrite( const FileName : string) : TFileTime
Function GetFileList(FileList: TStringlist; apath: string): TStringlist;
Function GetFileList1(apath: string): TStringlist;
Function GetFileMIMEType( const AFileName : string) : string
Function GetFileSize( const FileName : string) : Int64
Function GetFileVersion( AFileName : string) : Cardinal
Function GetFileVersion( const AFilename : string) : Cardinal
Function GetFileSize2(Handle: Integer; x: Integer): Integer; stdcall;
Function GetFileDate(aFile:string; aWithTime:Boolean):string;
Function GetFilterData( Root : PExprNode) : TExprData
Function getFirstChild : LongInt
Function getFirstChild : TTreeNode
Function GetFirstDelimitedToken( const cDelim : char; const cStr : string) : string
Function GetFirstNode : TTreeNode
Function GetFontsFolder : string
Function GetFormulaValue( const Formula : string) : Extended
Function GetFreePageFileMemory : Integer
Function GetFreePhysicalMemory : Integer
Function GetFreeSystemResources( const ResourceType : TFreeSysResKind) : Integer;
Function GetFreeSystemResources1 : TFreeSystemResources;
Function GetFreeVirtualMemory : Integer
Function GetFromClipboard : Boolean
Function GetFullURI( const AOptionalFileds : TIdURIOptionalFieldsSet) : String
Function GetGBitmap( Value : TBitmap) : TBitmap
Function GetGMTDateByName( const AFileName : TIdFileName) : TDateTime
Function GetGroupState( Level : Integer) : TGroupPosInds
Function GetHandle : HWND
Function GETHELPCONTEXT( VALUE : INTEGER; BYCOMMAND : BOOLEAN) : THELPCONTEXT
function GetHexArray(ahexdig: THexArray): THexArray;
Function GetHighLightColor( const Color : TColor; Luminance : Integer) : TColor
function GetHINSTANCE: longword;
Function GetHistoryFolder : string
Function GetHitTestInfoAt( X, Y : Integer) : THitTests
function getHMODULE: longword;
Function GetHostByName(const AComputerName: String): String;
Function GetHostName : string
Function getHostIP: string;
Function GetHotSpot : TPoint
Function GetHueBitmap( Value : TBitmap) : TBitmap
Function GetImageBitmap : HBITMAP
Function GETIMAGELIST : TCUSTOMIMAGELIST
Function GetIncome( const aNetto : Currency) : Currency
Function GetIncome( const aNetto : Extended) : Extended
Function GetIncome( const aNetto : Extended): Extended
Function GetIncome(const aNetto : Extended) : Extended
function GetIncome(const aNetto: Currency): Currency
Function GetIncome2( const aNetto : Currency) : Currency
Function GetIncome2( const aNetto : Currency): Currency
Function getIndex_Attrs( tag : string; var idx : Integer; var Attrs : string) : string
Function GETINDEXFORFIELDS( const FIELDS : String; CASEINSENSITIVE : BOOLEAN) : TINDEXDEF
Function GetIndexForOrderBy( const SQL : WideString; DataSet : TDataSet) : TIndexDef
Function GetInstRes(Instance:THandle;ResType:TResType;const
Name:string;Width:Integer;LoadFlags:TLoadResources;MaskColor:TColor):Boolean;
Function
GetInstRes1(Instance:THandle;ResType:TResType;ResID:DWORD;Width:Integer;LoadFlags:TLoadResources;MaskColor:
TColor):Boolean;
Function GetIntelCacheDescription( const D : Byte) : string
Function GetInteractiveUserName : string
Function GetInternetCacheFolder : string
Function GetInternetFormattedFileTimeStamp( const AFilename : String) : String
Function GetIPAddress( const HostName : string) : string
Function GetIP( const HostName : string) : string
Function GetIPHostByName(const AComputerName: String): String;
Function GetIsAdmin: Boolean;
Function GetItem( X, Y : Integer) : LongInt
Function GetItemAt( X, Y : Integer) : TListItem
Function GetItemHeight(Font: TFont): Integer;
Function GetItemPath( Index : Integer) : string
Function GetKeyFieldNames( List : TStrings) : Integer;
Function GetKeyFieldNames1( List : TWideStrings) : Integer;
Function GetKeyState( const VirtualKey : Cardinal) : Boolean
Function GetLastChild : LongInt
Function GetLastChild : TTreeNode
Function GetLastDelimitedToken( const cDelim : char; const cStr : string) : string
function GetLastError: Integer
Function GetLAT_CONV_FACTOR: double;  //for WGS84 power(1 - 1 / 298.257223563, 2);
Function GetLoader( Ext : string) : TBitmapLoader
Function GetLoadFilter : string
Function GetLocalComputerName : string
Function GetLocaleChar( Locale, LocaleType : Integer; Default : Char) : Char
Function GetLocaleStr( Locale, LocaleType : Integer; Default : string) : string
Function GetLocalUserName : string
Function GetLoginUsername : WideString
function getLongDayNames: string)
Function GetLongHint(const hint: string): string
function getLongMonthNames: string)
Function GetMacAddresses( const Machine : string; const Addresses : TStrings) : Integer
Function GetMainAppWndFromPid( PID : DWORD) : HWND
Function GetMaskBitmap : HBITMAP
Function GetMaxAppAddress : Integer
Function GetMciErrorMessage( const MciErrNo : MCIERROR) : string
Function GetMemoryLoad : Byte
Function GetMIMEDefaultFileExt( const MIMEType : string) : TIdFileName
Function GetMIMETypeFromFile( const AFile : string) : string
Function GetMIMETypeFromFile( const AFile : TIdFileName) : string
Function GetMinAppAddress : Integer
Function GetModule : TComponent
Function GetModuleHandle( ModuleName : PChar) : HMODULE
Function GetModuleName( Module : HMODULE) : string
Function GetModulePath( const Module : HMODULE) : string
Function GetModuleFileName(Module: Integer; Filename: PChar;Size: Integer): Integer; stdcall;
Function GetCommandLine: PChar; stdcall;
Function GetMonochromeBitmap( Value : TBitmap) : TBitmap
Function GetMultiN(aval: integer): string;
Function GetName : String
Function GetNearestItem( Point : TPoint; Direction : TSearchDirection) : TListItem
Function GetNethoodFolder : string
Function GetNext : TTreeNode
Function GetNextChild( Value : LongInt) : LongInt
Function GetNextChild( Value : TTreeNode) : TTreeNode
Function GetNextDelimitedToken( const cDelim : char; var cStr : String) : String
Function GetNextItem( StartItem: TListItem;Direction:TSearchDirection;States:TItemStates) : TListItem
Function GetNextPacket : Integer
Function getNextSibling : TTreeNode
Function GetNextVisible : TTreeNode
Function GetNode( ItemId : HTreeItem) : TTreeNode
Function GetNodeAt( X, Y : Integer) : TTreeNode
Function GetNodeDisplayWidth( Node : TOutlineNode) : Integer
function GetNumberOfProcessors: longint;
Function GetNumLockKeyState : Boolean
Function GetObjectProperty( Instance : TPersistent; const PropName : string) : TObject
Function GetOnlyTransitionOn( cChar : char) : TniRegularExpressionState
Function GetOptionalParam( const ParamName : string) : OleVariant
Function GetOSName: string;
Function GetOSVersion: string;
Function GetOSNumber: string;
Function GetOsVersionInfo: TOSVersionInfo;      //thx to wischnewski
Function GetPackageModuleHandle( PackageName : PChar) : HMODULE
function GetPageSize: Cardinal;
Function GetParameterFileName : string
Function GetParams( var OwnerData : OleVariant) : OleVariant
Function GETPARENTCOMPONENT : TCOMPONENT
Function GetParentForm(control: TControl): TForm
Function GETPARENTMENU : TMENU
Function GetPassword : Boolean
Function GetPassword : string
Function GetPersonalFolder : string
Function GetPidFromProcessName( const ProcessName : string) : DWORD
Function GetPosition : TPoint
Function GetPrev : TTreeNode
Function GetPrevChild( Value : LongInt) : LongInt
Function GetPrevChild( Value : TTreeNode) : TTreeNode
Function getPrevSibling : TTreeNode
Function GetPrevVisible : TTreeNode
Function GetPrinthoodFolder : string
Function GetPrivilegeDisplayName( const PrivilegeName : string) : string
Function getProcessList: TStrings;
Function GetProcessId : TIdPID
Function GetProcessNameFromPid( PID : DWORD) : string
Function GetProcessNameFromWnd( Wnd : HWND) : string
Function GetProcessMemoryInfo(Process: THandle;ppsmemCounters: TProcessMemoryCounters;cb: DWORD):BOOL
Function getProcessAllMemory(ProcessID : DWORD): TProcessMemoryCounters;
Function getProcessMemoryInfo2(ProcessID : DWORD): TProcessMemoryCounters;
Function GetProgramFilesFolder : string
Function GetProgramsFolder : string
Function GetProxy : string
Function GetQuoteChar : WideString
Function GetQrCode4(Width,Height:Word; Correct_Level:string;const
Data:string;aformat:string):TLinearBitmap;
Function GetQrCodetoFile(Width,Height:Word;Correct_Level:string;const
Data:string;aformat:string):TLinearBitmap;
Function GetRate : Double
Function getPerfTime: string;
Function getRuntime: string;
Function GetRBitmap( Value : TBitmap) : TBitmap
Function GetReadableName( const AName : string) : string
Function GetRecentDocs : TStringList
Function GetRecentFolder : string
Function GetRecords( Count : Integer; out RecsOut : Integer; Options : Integer) : OleVariant;
Function GetRecords1(Count:Integer; out RecsOut:Integer;Options:Integer;const CommandText:WideString;var
Params, OwnerData : OleVariant) : OleVariant;
Function GetRecordset( const CommandText : WideString; ConnectionString : WideString) : _Recordset
Function GetRegisteredCompany : string
Function GetRegisteredOwner : string
Function GetResource(ResType:TResType;const
Name:string;Width:Integer;LoadFlags:TLoadResources;MaskColor:TColor:Boolean
Function GetResourceName( ObjStream : TStream; var AName : string) : Boolean
Function GetResponse( const AAllowedResponses : array of SmallInt) : SmallInt;
Function GetResponse1( const AAllowedResponse : SmallInt) : SmallInt;
Function GetRValue( rgb : DWORD) : Byte
Function GetGValue( rgb : DWORD) : Byte
Function GetBValue( rgb : DWORD) : Byte
Function GetCValue( cmyk : COLORREF) : Byte
Function GetMValue( cmyk : COLORREF) : Byte
Function GetYValue( cmyk : COLORREF) : Byte
Function GetKValue( cmyk : COLORREF) : Byte
Function CMYK( c, m, y, k : Byte) : COLORREF
Function GetOSName: string;
Function GetProcAddress( hModule : HMODULE; lpProcName : LPCSTR) : FARPROC
Function GetProcAddress(Module : HMODULE; Proc : PChar): Dword
Function GetSafeCallExceptionMsg : String
Function GetSaturationBitmap( Value : TBitmap) : TBitmap
Function GetSaveFilter : string
Function GetSaver( Ext : string) : TBitmapLoader
Function GetScrollLockKeyState : Boolean
Function GetSearchString : string
Function GetSelections( AList : TList) : TTreeNode
function GETSELTEXTBUF(BUFFER:PCHAR;BUFSIZE:INTEGER):INTEGER
Function GetSendToFolder : string
Function GetServer : IAppServer
Function GetServerList : OleVariant
Function GetShadowColor( const Color : TColor; Luminance : Integer) : TColor
Function GetShellProcessHandle : THandle
Function GetShellProcessName : string
Function GetShellVersion : Cardinal
function getShortDayNames: string)
Function GetShortHint(const hint: string): string
function getShortMonthNames: string)
Function GetSizeOfFile( const FileName : string) : Int64;
Function GetSizeOfFile1( Handle : THandle) : Int64;
Function GetStdHandle(nStdHandle: Integer): Integer; stdcall;
Function GetStartmenuFolder : string
Function GetStartupFolder : string
Function GetStringProperty( Instance : TPersistent; const PropName : string) : WideString
Function GetSuccessor( cChar : char) : TniRegularExpressionState
Function GetSwapFileSize : Integer
Function GetSwapFileUsage : Integer
Function GetSystemLocale : TIdCharSet
Function GetSystemMetrics( nIndex : Integer) : Integer
Function GetSystemPathSH(Folder: Integer): TFilename ;
Function GetTableNameFromQuery( const SQL : Widestring) : Widestring
Function GetTableNameFromSQL( const SQL : WideString) : WideString
Function GetTableNameFromSQLEx( const SQL : WideString; IdOption : IDENTIFIEROption) : WideString
Function GetTasksList( const List : TStrings) : Boolean
Function getTeamViewerID: string;
Function GetTemplatesFolder : string
Function GetText : PwideChar
function GetText:PChar
Function GetTextBuf( Buffer : PChar; BufSize : Integer) : Integer
function GETTEXTBUF(BUFFER:PCHAR;BUFSIZE:INTEGER):INTEGER
Function GetTextItem( const Value : string) : Longint
function GETTEXTLEN:INTEGER
Function GetThreadLocale: Longint; stdcall
Function GetCurrentThreadID: LongWord; stdcall;
Function GetTickCount : Cardinal
Function GetTickDiff( const AOldTickCount, ANewTickCount : Cardinal) : Cardinal
Function GetTicketNr : longint
Function GetTime : Cardinal
Function GetTime : TDateTime
Function GetTimeout : Integer
Function GetTimeStr: String
Function GetTimeString: String
Function GetTodayFiles(startdir, amask: string): TStringlist;
Function getTokenCounts : integer
Function GetTotalPageFileMemory : Integer
Function GetTotalPhysicalMemory : Integer
Function GetTotalVirtualMemory : Integer
Function GetUniqueFileName( const APath, APrefix, AExt : String) : String
Function GetUseNowForDate : Boolean
Function GetUserDomainName( const CurUser : string) : string
Function GetUserName : string
Function GetUserName: string;
Function GetUserObjectName( hUserObject : THandle) : string
Function GetValueBitmap( Value : TBitmap) : TBitmap
Function GetValueMSec : Cardinal
Function GetValueStr : String
Function GetVersion: int;
Function GetVersionString(FileName: string): string;
Function GetVisibleNode( Index : LongInt) : TOutlineNode
Function GetVolumeFileSystem( const Drive : string) : string
Function GetVolumeName( const Drive : string) : string
Function GetVolumeSerialNumber( const Drive : string) : string
Function GetWebAppServices : IWebAppServices
Function GetWebRequestHandler : IWebRequestHandler
Function GetWindowCaption( Wnd : HWND) : string
Function GetWindowDC(hdwnd: HWND): HDC;
Function GetWindowIcon( Wnd : HWND; LargeIcon : Boolean) : HICON
Function GetWindowRect(hwnd: HWND; arect: TRect): Boolean
Function GetWindowsComputerID : string
function GetWindowsDirectory(lpBuffer: PChar; uSize: longword): longword;
Function GetWindowsFolder : string
Function GetWindowsServicePackVersion : Integer
Function GetWindowsServicePackVersionString : string
Function GetWindowsSystemFolder : string
Function GetWindowsTempFolder : string
Function GetWindowsUserID : string
Function GetWindowsVersion : TWindowsVersion
Function GetWindowsVersionString : string
Function GmtOffsetStrToDateTime( S : string) : TDateTime
Function GMTToLocalDateTime( S : string) : TDateTime
Function GotoKey : Boolean
Function GradToCycle( const Grads : Extended) : Extended
Function GradToDeg( const Grads : Extended) : Extended
Function GradToDeg( const Value : Extended) : Extended;
Function GradToDeg1( const Value : Double) : Double;
Function GradToDeg2( const Value : Single) : Single;
Function GradToRad( const Grads : Extended) : Extended
Function GradToRad( const Value : Extended) : Extended;
Function GradToRad1( const Value : Double) : Double;
Function GradToRad2( const Value : Single) : Single;
Function Gray32( const Intensity : Byte; const Alpha : Byte) : TColor32
Function GreenComponent( const Color32 : TColor32) : Integer
function GUIDToString(const GUID: TGUID): string)
Function HandleAllocated : Boolean
function HandleAllocated: Boolean;
Function HandleRequest : Boolean
Function HandleRequest( Request : TWebRequest; Response : TWebResponse) : Boolean
Function HarmonicMean( const X : TDynFloatArray) : Float
Function HasAsParent( Value : TTreeNode) : Boolean
Function HASCHILDDEFS : BOOLEAN
Function HasCurValues : Boolean
Function HasExtendCharacter( const s : UTF8String) : Boolean
Function HasFormat( Format : Word) : Boolean
Function HashValue( AStream : TStream) : T5x4LongWordRecord;
Function HashValue(AStream : TStream) : T4x4LongWordRecord
Function HashValue(AStream: TStream): LongWord
Function HashValue(AStream: TStream): Word
Function HashValue1( AStream : TStream; const ABeginPos, AEndPos : Int64) : T5x4LongWordRecord;
Function HashValue1(AStream : TStream): T4x4LongWordRecord
Function HashValue128(const ASrc: string): T4x4LongWordRecord;
Function HashValue128Stream(AStream: TStream): T4x4LongWordRecord;
Function HashValue16( const ASrc : string) : Word;
Function HashValue16stream( AStream : TStream) : Word;
Function HashValue32( const ASrc : string) : LongWord;
Function HashValue32Stream( AStream : TStream) : LongWord;
Function HasMergeConflicts : Boolean
Function hasMoreTokens : boolean
Function HASPARENT : BOOLEAN
function HasParent: Boolean
Function HasTransaction( Transaction : TDBXTransaction) : Boolean
Function HasUTF8BOM( S : TStream) : boolean;
Function HasUTF8BOM1( S : AnsiString) : boolean;
Function Haversine( X : Float) : Float
Function Head( s : string; const subs : string; var tail : string) : string
function HELPCOMMAND(COMMAND:INTEGER;DATA:LONGINT):BOOLEAN
function HELPCONTEXT(CONTEXT:THELPCONTEXT):BOOLEAN
function HELPJUMP(JUMPID:STRING):BOOLEAN
Function HeronianMean( const a, b : Float) : Float
function HexStrToStr(Value: string): string;
function HexToBin(Text,Buffer:PChar; BufSize:Integer):Integer;
function HexToBin2(HexNum: string): string;
Function HexToDouble( const Hex : string) : Double
function HexToInt(hexnum: string): LongInt;
function HexToStr(Value: string): string;
Function HexifyBlock( var Buffer, BufferSize : Integer) : string
function Hi(vdat: word): byte;
function HiByte(W: Word): Byte)
function High: Int64;
Function HighlightCell(DataCol,DataRow: Integer; const Value:string; AState:TGridDrawState): Boolean
function HINSTANCE: longword;
function HiWord(l: DWORD): Word)
function HMODULE: longword;
Function HourOf( const AValue : TDateTime) : Word
Function HourOfTheDay( const AValue : TDateTime) : Word
Function HourOfTheMonth( const AValue : TDateTime) : Word
Function HourOfTheWeek( const AValue : TDateTime) : Word
Function HourOfTheYear( const AValue : TDateTime) : Word
Function HoursBetween( const ANow, AThen : TDateTime) : Int64
Function HourSpan( const ANow, AThen : TDateTime) : Double
Function HSLToRGB1( const H, S, L : Single) : TColor32;
Function HTMLDecode( const AStr : String) : String
Function HTMLEncode( const AStr : String) : String
Function HTMLEscape( const Str : string) : string
Function HtmlTable( DataSet : TDataSet; DataSetHandler : TDSTableProducer; MaxRows : Integer) : string
Function HTTPDecode( const AStr : String) : string
Function HTTPEncode( const AStr : String) : string
Function Hypot( const X, Y : Extended) : Extended
Function IBMax( n1, n2 : Integer) : Integer
Function IBMin( n1, n2 : Integer) : Integer
Function IBRandomString( iLength : Integer) : String
Function IBRandomInteger( iLow, iHigh : Integer) : Integer
Function IBStripString( st : String; CharsToStrip : String) : String
Function IBFormatIdentifier( Dialect : Integer; Value : String) : String
Function IBFormatIdentifierValue( Dialect : Integer; Value : String) : String
Function IBExtractIdentifier( Dialect : Integer; Value : String) : String
Function IBQuoteIdentifier( Dialect : Integer; Value : String) : String
Function IBAddIBParamSQLForDetail(Params:TParams;SQL:string;Native:Boolean;Dialect:Integer):string
Procedure IBDecomposeDatabaseName(DatabaseName:String;var ServerName,Protocol,DatabasePath:String)
 Function RandomString( iLength : Integer) : String');
 Function RandomInteger( iLow, iHigh : Integer) : Integer');
 Function StripString( st : String; CharsToStrip : String) : String');
 Function FormatIdentifier( Dialect : Integer; Value : String) : String');
 Function FormatIdentifierValue( Dialect : Integer; Value : String) : String');
 Function ExtractIdentifier( Dialect : Integer; Value : String) : String');
 Function QuoteIdentifier( Dialect : Integer; Value : String) : String');
 Function AddIBParamSQLForDetail( Params : TParams; SQL : string; Native : Boolean; Dialect : Integer) : string');
 Procedure DecomposeDatabaseName( DatabaseName : String; var ServerName, Protocol, DatabasePath : String)');
 function NextSQLToken(var p: PChar; var Token: String; CurSection: TSQLToken): TSQLToken;');
Function IconToBitmap( Ico : HICON) : TBitmap
Function IconToBitmap2( Ico : HICON; Size : Integer; TransparentColor : TColor) : TBitmap
Function IconToBitmap3( Ico : HICON; Size : Integer; TransparentColor : TColor) : TBitmap
function IdentToCharset(const Ident: string; var Charset: Longint): Boolean)
function IdentToColor(const Ident: string; var Color: Longint): Boolean)
function IdentToCursor(const Ident: string; var cursor: Longint): Boolean;
Function IdGetDefaultCharSet : TIdCharSet
function IDispatchInvoke(Self:IDispatch;ProperSet:Boolean;const Name:String;Par:array of variant):variant
Function IdPorts2 : TStringList
Function IdToMib( const Id : string) : string
Function IdSHA1Hash(apath: string): string;
Function IdHashSHA1(apath: string): string;
Function IfStr( const bCondition : boolean; const sTrue : string; const sFalse : string) : string
Function IfThen( AValue : Boolean; const ATrue : string; AFalse : string) : string;
Function iif1( ATest : Boolean; const ATrue : Integer; const AFalse : Integer) : Integer;
Function iif2( ATest : Boolean; const ATrue : string; const AFalse : string) : string;
Function iif3( ATest : Boolean; const ATrue : Boolean; const AFalse : Boolean) : Boolean;
function ImportTest(S1: string;s2:longint; s3:Byte; s4:word; var s5:string): string;
```

```
Function IncDay( const AValue : TDateTime; const ANumberOfDays : Integer) : TDateTime
Function IncHour( const AValue : TDateTime; const ANumberOfHours : Int64) : TDateTime
Function IncLimit( var B : Byte; const Limit : Byte; const Incr : Byte) : Byte;
Function IncLimit1( var B : Shortint; const Limit : Shortint; const Incr : Shortint) : Shortint;
Function IncLimit2( var B : Smallint; const Limit : Smallint; const Incr : Smallint) : Smallint;
Function IncLimit3( var B : Word; const Limit : Word; const Incr : Word) : Word;
Function IncLimit4( var B : Integer; const Limit : Integer; const Incr : Integer) : Integer;
Function IncLimit5( var B : Cardinal; const Limit : Cardinal; const Incr : Cardinal) : Cardinal;
Function IncLimit6( var B : Int64; const Limit : Int64; const Incr : Int64) : Int64;
Function IncLimitClamp( var B : Byte; const Limit : Byte; const Incr : Byte) : Byte;
Function IncLimitClamp1( var B : Shortint; const Limit : Shortint; const Incr : Shortint) : Shortint;
Function IncLimitClamp2( var B : Smallint; const Limit : Smallint; const Incr : Smallint) : Smallint;
Function IncLimitClamp3( var B : Word; const Limit : Word; const Incr : Word) : Word;
Function IncLimitClamp4( var B : Integer; const Limit : Integer; const Incr : Integer) : Integer;
Function IncLimitClamp5( var B : Cardinal; const Limit : Cardinal; const Incr : Cardinal) : Cardinal;
Function IncLimitClamp6( var B : Int64; const Limit : Int64; const Incr : Int64) : Int64;
Function IncludeTrailingBackslash( S : string) : string
function IncludeTrailingBackslash(const S: string): string)
Function IncludeTrailingPathDelimiter( const APath : string) : string
Function IncludeTrailingPathDelimiter( S : string) : string
function IncludeTrailingPathDelimiter(const S: string): string)
Function IncludeTrailingSlash( const APath : string) : string
Function IncMilliSecond( const AValue : TDateTime; const ANumberOfMilliSeconds : Int64) : TDateTime
Function IncMinute( const AValue : TDateTime; const ANumberOfMinutes : Int64) : TDateTime
Function IncMonth( DateTime : TDateTime; NumberOfMonths : Integer) : TDateTime
function IncMonth(const DateTime: TDateTime; NumberOfMonths: Integer): TDateTime)
Function IncSecond( const AValue : TDateTime; const ANumberOfSeconds : Int64) : TDateTime
Function IncWeek( const AValue : TDateTime; const ANumberOfWeeks : Integer) : TDateTime
Function IncYear( const AValue : TDateTime; const ANumberOfYears : Integer) : TDateTime
Function IndexOf( AClass : TClass) : Integer
Function IndexOf( AComponent : TComponent) : Integer
Function IndexOf( AObject : TObject) : Integer
Function INDEXOF( const ANAME : String) : INTEGER
Function IndexOf( const DisplayName : string) : Integer
Function IndexOf( const Item : TBookmarkStr) : Integer
Function IndexOf( const S : WideString) : Integer
Function IndexOf( const View : TJclFileMappingView) : Integer
Function INDEXOF( FIELD : TFIELD) : INTEGER
Function IndexOf( ID : LCID) : Integer
Function INDEXOF( ITEM : TMENUITEM) : INTEGER
Function IndexOf( Value : TListItem) : Integer
Function IndexOf( Value : TTreeNode) : Integer
function IndexOf(const S: string): Integer;
Function IndexOfName( const Name : WideString) : Integer
function IndexOfName(Name: string): Integer;
Function IndexOfObject( AObject : TObject) : Integer
function IndexofObject(AObject:tObject):Integer
Function IndexOfTabAt( X, Y : Integer) : Integer
Function IndexStr( const AText : string; const AValues : array of string) : Integer
Function IndexText( const AText : string; const AValues : array of string) : Integer
Function IndexOfInteger( AList : TStringList; Value : Variant) : Integer
Function IndexOfFloat( AList : TStringList; Value : Variant) : Integer
Function IndexOfDate( AList : TStringList; Value : Variant) : Integer
Function IndexOfString( AList : TStringList; Value : Variant) : Integer
Function IndyCompareStr( const A1 : string; const A2 : string) : Integer
Function IndyGetHostName : string
Function IndyInterlockedDecrement( var I : Integer) : Integer
Function IndyInterlockedExchange( var A : Integer; B : Integer) : Integer
Function IndyInterlockedExchangeAdd( var A : Integer; B : Integer) : Integer
Function IndyInterlockedIncrement( var I : Integer) : Integer
Function IndyLowerCase( const A1 : string) : string
Function IndyStrToBool( const AString : String) : Boolean
Function IndyUpperCase( const A1 : string) : string
Function InitCommonControl( CC : Integer) : Boolean
Function InitTempPath : string
Function InMainThread : boolean
Function inOpArray( W : WideChar; sets : array of WideChar) : boolean
Function Input: Text)
Function InputBox( const ACaption, APrompt, ADefault : string) : string
function InputBox(const ACaption: string; const APrompt: string; const ADefault: string): string)
Function InputLn(const AMask: string; AEcho:Boolean;ATabWidth:Integer;AMaxLineLength:Integer): string
Function InputQuery( const ACaption, APrompt : string; var Value : string) : Boolean
function InputQuery(const ACaption: string; const APrompt: string; var Value: string): Boolean)
Function InquireSignal( RtlSigNum : Integer) : TSignalState
Function InRangeR( const A, Min, Max : Double) : Boolean
function Insert( Index : Integer) : TCollectionItem
Function Insert( Index : Integer) : TComboExItem
Function Insert( Index : Integer) : THeaderSection
Function Insert( Index : Integer) : TListItem
Function Insert( Index : Integer) : TStatusPanel
Function Insert( Index : Integer) : TWorkArea
Function Insert( Index : LongInt; const Text : string) : LongInt
Function Insert( Sibling : TTreeNode; const S : string) : TTreeNode
Function INSERTNEWLINEAFTER( AITEM : TMENUITEM) : INTEGER
Function INSERTNEWLINEBEFORE( AITEM : TMENUITEM) : INTEGER
Function InsertNode( Node, Sibling : TTreeNode; const S : string; Ptr : Pointer) : TTreeNode
Function InsertObject( Index : LongInt; const Text : string; const Data : Pointer) : LongInt
Function InsertObject( Sibling : TTreeNode; const S : string; Ptr : Pointer) : TTreeNode
Function Instance : Longint
```

```
function InstanceSize: Longint
Function Int(e : Extended) : Extended;
function Int64ToStr(i: Int64): String;
Function IntegerToBcd( const AValue : Integer) : TBcd
Function Intensity( const Color32 : TColor32) : Integer;
Function Intensity( const R, G, B : Single) : Single;
Function InterestPayment(const Rate:Extended;Period,NPeriods:Integer;const PresentValue, FutureValue:Extended; PaymentTime : TPaymentTime) : Extended
Function InterestRate(NPeriods:Integer;const Payment,PresentVal, FutureVal:Extended;PaymentTime:TPaymentTime): Extended
Function InternalDecodeDate( DateTime : TDateTime; var Year, Month, Day, DOW : Word) : Boolean
Function InternalRateOfReturn( const Guess : Extended; const CashFlows : array of Double) : Extended
Function InternalUpdateRecord( Tree : TUpdateTree) : Boolean
Function IntersectRect( out Rect : TRect; const R1, R2 : TRect) : Boolean
function IntersectRect(out Rect: TRect; const R1, R2: TRect): Boolean)
Function IntMibToStr( const Value : string) : string
Function IntPower( const Base : Extended; const Exponent : Integer) : Extended
Function IntToBin( Value : cardinal) : string
Function IntToHex( Value : Integer; Digits : Integer) : string;
function IntToHex(a: integer; b: integer): string;
Function IntToHex64( Value : Int64; Digits : Integer) : string;
function IntToHex64(Value: Int64; Digits: Integer): string)
Function IntTo3Str( Value : Longint; separator: string) : string
Function inttobool( aInt : LongInt) : Boolean
function IntToStr(i: Int64): String;
Function IntToStr64(Value: Int64): string)
function IOResult: Integer
Function IPv6AddressToStr(const AValue: TIdIPv6Address): string
Function IsAccel(VK: Word; const Str: string): Boolean
Function IsAddressInNetwork( Address : String) : Boolean
Function IsAdministrator : Boolean
Function IsAlias( const Name : string) : Boolean
Function IsApplicationRunning( const AClassName, ApplName : string) : Boolean
Function IsASCII( const AByte : Byte) : Boolean;
Function IsASCIILDH( const AByte : Byte) : Boolean;
Function IsAssembly(const FileName: string): Boolean;
Function IsBcdNegative( const Bcd : TBcd) : Boolean
Function IsBinary(const AChar : Char) : Boolean
function IsConsole: Boolean)
Function IsDelimiter( Delimiters, S : string; Index : Integer) : Boolean
function IsDelimiter(const Delimiters: string; const S: string; Index: Integer): Boolean)
Function IsDelphiDesignMode : boolean
Function IsDelphiRunning : boolean
Function IsDFAState : boolean
Function IsDirectory( const FileName : string) : Boolean
Function IsDomain( const S : String) : Boolean
Function IsDragObject(Sender: TObject): Boolean;
Function IsEditing : Boolean
Function ISEMPTY : BOOLEAN
Function IsEqual( Value : TParameters) : Boolean
Function ISEQUAL( VALUE : TPARAMS) : BOOLEAN
function IsEqualGUID(const guid1, guid2: TGUID): Boolean)
Function IsFirstNode : Boolean
Function IsFloatZero( const X : Float) : Boolean
Function IsFormatRegistered( Extension, AppID : string) : Boolean
Function IsFormOpen(const FormName: string): Boolean;
Function IsFQDN( const S : String) : Boolean
Function IsGrayScale : Boolean
Function IsHex( AChar : Char) : Boolean;
Function IsHexString(const AString: string): Boolean;
Function IsHostname( const S : String) : Boolean
Function IsInfinite( const AValue : Double) : Boolean
Function IsInLeapYear( const AValue : TDateTime) : Boolean
Function IsInternet: boolean;
Function IsLeadChar( ACh : Char) : Boolean
Function IsLeapYear( Year : Word) : Boolean
function IsLeapYear(Year: Word): Boolean)
function IsLibrary: Boolean)
Function ISLINE : BOOLEAN
Function IsLinkedTo( DataSet : TDataSet) : Boolean
Function ISLINKEDTO( DATASOURCE : TDATASOURCE) : BOOLEAN
Function IsLiteralChar( const EditMask : string; Offset : Integer) : Boolean
Function IsMatch( const Pattern, Text : string) : Boolean  //Grep like RegEx
Function IsMainAppWindow( Wnd : HWND) : Boolean
Function IsMediaPresentInDrive( Drive : Char) : Boolean
function IsMemoryManagerSet: Boolean)
Function IsMultiTableQuery( const SQL : WideString) : Boolean
function IsMultiThread: Boolean)
Function IsNumeric( AChar : Char) : Boolean;
Function IsNumeric2( const AString : string) : Boolean;
Function IsOctal( AChar : Char) : Boolean;
Function IsOctalString(const AString: string) : Boolean;
Function IsPathDelimiter( S : string; Index : Integer) : Boolean
Function IsPathDelimiter(const S: string; Index: Integer): Boolean)
Function IsPM( const AValue : TDateTime) : Boolean
Function IsPositiveFloatArray( const X : TDynFloatArray) : Boolean
Function IsPrimeFactor( const F, N : Cardinal) : Boolean
Function IsPrimeRM( N : Cardinal) : Boolean  //rabin miller
Function IsPrimeTD( N : Cardinal) : Boolean  //trial division
Function IsPrivilegeEnabled( const Privilege : string) : Boolean
Function ISqrt( const I : Smallint) : Smallint
Function IsReadOnly(const Filename: string): boolean;
Function IsRectEmpty( const Rect : TRect) : Boolean
function IsRectEmpty(const Rect: TRect): Boolean)
Function IsRelativePrime( const X, Y : Cardinal) : Boolean
Function ISRIGHTTOLEFT : BOOLEAN
function IsRightToLeft: Boolean
Function IsSameDay( const AValue, ABasis : TDateTime) : Boolean
Function ISSEQUENCED : BOOLEAN
Function IsSystemModule( const Module : HMODULE) : Boolean
Function IsSystemResourcesMeterPresent : Boolean
Function IsTCPPortOpen(dwPort : Word; ipAddressStr: String): boolean;
Function IsToday( const AValue : TDateTime) : Boolean
function IsToday(const AValue: TDateTime): Boolean;
Function IsTopDomain( const AStr : string) : Boolean
Function IsUTF8LeadByte( Lead : Char) : Boolean
Function IsUTF8String( const s : UTF8String) : Boolean
Function IsUTF8TrailByte( Lead : Char) : Boolean
Function ISVALIDCHAR( INPUTCHAR : CHAR) : BOOLEAN
Function IsValidDate( const AYear, AMonth, ADay : Word) : Boolean
Function IsValidDateDay( const AYear, ADayOfYear : Word) : Boolean
Function IsValidDateMonthWeek( const AYear, AMonth, AWeekOfMonth, ADayOfWeek : Word) : Boolean
Function IsValidDateTime(const AYear,AMonth, ADay,AHour,AMinute, ASecond, AMilliSecond: Word): Boolean
Function IsValidDateWeek( const AYear, AWeekOfYear, ADayOfWeek : Word) : Boolean
Function IsValidIdent( Ident : string) : Boolean
function IsValidIdent(const Ident: string; AllowDots: Boolean): Boolean)
Function IsValidIP( const S : String) : Boolean
Function IsValidTime( const AHour, AMinute, ASecond, AMilliSecond : Word) : Boolean
Function IsValidISBN( const ISBN : AnsiString) : Boolean
Function IsVariantManagerSet: Boolean; //deprecated;
Function IsVirtualPcGuest : Boolean;
Function IsVmWareGuest : Boolean;
Function IsVCLControl(Handle: HWnd): Boolean;
Function IsWhiteString( const AStr : String) : Boolean
Function IsWindowResponding( Wnd : HWND; Timeout : Integer) : Boolean
Function IsWoW64: boolean;
Function IsWin64: boolean;
Function IsWow64String(var s: string): Boolean;
Function IsWin64String(var s: string): Boolean;
Function IsWindowsVista: boolean;
Function isPowerof2(num: int64): boolean;
Function powerOf2(exponent: integer): int64;
function IsZero(const A: Extended; Epsilon: Extended): Boolean //overload;
function IsZero1(const A: Double; Epsilon: Double): Boolean //overload;
function IsZero2(const A: Single; Epsilon: Single): Boolean //overload;
Function ItemAtPos( Pos : TPoint; IgnoreTabHeight : Boolean) : Integer
function ITEMATPOS(POS:TPOINT;EXISTING:BOOLEAN):INTEGER
Function ItemRect( Index : Integer) : TRect
function ITEMRECT(INDEX:INTEGER):TRECT
Function ItemWidth( Index : Integer) : Integer
Function JavahashCode(val: string): Integer;
Function JosephusG(n,k: integer; var graphout: string): integer;
Function JulianDateToDateTime( const AValue : Double) : TDateTime
Function JvMessageBox( const Text, Caption : string; Flags : DWORD) : Integer;
Function JvMessageBox1( const Text : string; Flags : DWORD) : Integer;
Function KeepAlive : Boolean
Function KeysToShiftState(Keys: Word): TShiftState;
Function KeyDataToShiftState(KeyData: Longint): TShiftState;
Function KeyboardStateToShiftState2(const KeyboardState: TKeyboardState): TShiftState;
Function KeyboardStateToShiftState: TShiftState; overload;
Function Languages : TLanguages
Function Last : TClass
Function Last : TComponent
Function Last : TObject
Function LastDelimiter( Delimiters, S : string) : Integer
function LastDelimiter(const Delimiters: string; const S: string): Integer)
Function LastPos( const ASubstr : string; const AStr : string) : Integer
Function Latitude2WGS84(lat: double): double;
Function LCM(m,n:longint):longint;
Function LCMJ( const X, Y : Cardinal) : Cardinal
Function Ldexp( const X : Extended; const P : Integer) : Extended
Function LeftStr( const AText : AnsiString; const ACount : Integer) : AnsiString;
Function LeftStr( const AText : WideString; const ACount : Integer) : WideString;
function Length: Integer;
Procedure LetFileList(FileList: TStringlist; apath: string);
function lengthmp3(mp3path: string):integer;
Function LineInRect( const P1, P2 : TPoint; const Rect : TRect) : Boolean;
Function LineInRect1( const P1, P2 : TFloatPoint; const Rect : TFloatRect) : Boolean;
Function LineSegmentIntersection(const L1P1 : TFloatPoint; L1P2 : TFloatPoint; const L2P1 : TFloatPoint;
L2P2 : TFloatPoint; var P : TFloatPoint) : Boolean
function LineStart(Buffer, BufPos: PChar): PChar
function LineStart(Buffer, BufPos: PChar): PChar)
function ListSeparator: char;
function Ln(x: Extended): Extended;
Function LnXP1( const X : Extended) : Extended
function Lo(vdat: word): byte;
Function LoadCursor(hInstance: HINST; lpCursorName: PChar): HCURSOR
Function LoadedModulesList( const List : TStrings; ProcessID : DWORD; HandlesOnly : Boolean) : Boolean
Function LoadFileAsString( const FileName : string) : string
Function LoadFromFile( const FileName : string) : TBitmapLoader
Function LoadLibraryEx(LibName: PChar; hFile: Longint; Flags: Longint): Longint; stdcall;
Function LoadPackage(const Name: string): HMODULE
Function LoadResource( ModuleHandle : HMODULE; ResHandle : TResourceHandle) : HGLOBAL
Function LoadStr( Ident : Integer) : string
Function LoadString(Instance: Longint; IDent: Integer; Buffer: PChar; Size: Integer): Integer; stdcall;
Function LoadWideStr( Ident : Integer) : WideString
Function LOCATE(const KEYFIELDS: String;const KEYVALUES:VARIANT;OPTIONS: TLOCATEOPTIONS) : BOOLEAN
Function LockRegion( libOffset : Longint; cb : Largeint; dwLockType : Longint) : HResult
Function LockServer( fLock : LongBool) : HResult
Function LockVolume( const Volume : string; var Handle : THandle) : Boolean
Function Log( const X : Extended) : Extended
Function Log10( const X : Extended) : Extended
Function Log2( const X : Extended) : Extended
function LogBase10(X: Float): Float;
Function LogBase2(X: Float): Float;
Function LogBaseN(Base, X: Float): Float;
Function LogN( const Base, X : Extended) : Extended
Function LogOffOS : Boolean
Function LoginDialog( const ADatabaseName : string; var AUserName, APassword : string) : Boolean
Function LoginDialogEx(const ADatabaseName:string;var AUserName,APassword:string;NameReadOnly:Bool):Bool;
Function LongDateFormat: string;
function LongTimeFormat: string;
Function LongWordToFourChar( ACardinal : LongWord) : string
Function LOOKUP(const KEYFIELDS: String; const KEYVALUES: VARIANT; const RESULTFIELDS: String): VARIANT
Function LookupName( const name : string) : TInAddr
Function LookupService( const service : string) : Integer
function Low: Int64;
Function LowerCase( S : string) : string
Function Lowercase(s : AnyString) : AnyString;
Function LRot( const Value : Byte; const Count : TBitRange) : Byte;
Function LRot1( const Value : Word; const Count : TBitRange) : Word;
Function LRot2( const Value : Integer; const Count : TBitRange) : Integer;
function MainInstance: longword
function MainThreadID: longword
Function Map(x, in_min, in_max, out_min, out_max: integer): integer;  //arduino
Function mapMax(ax, in_min, in_max, out_min, out_max: integer): integer;
Function MakeCanonicalIPv4Address( const AAddr : string) : string
Function MakeCanonicalIPv6Address( const AAddr : string) : string
Function MakeDIB( out Bitmap : PBitmapInfo) : Integer
Function MakeDWordIntoIPv4Address( const ADWord : Cardinal) : string
Function MakeLong(A, B: Word): Longint)
Function MakeTempFilename( const APath : String) : string
Function MakeValidFileName( const Str : string) : string
Function MakeValueMap( Enumeration : string; ToCds : Boolean) : string
Function MakeWord(A, B: Byte): Word)
Function MakeYear4Digit( Year, Pivot : Integer) : Integer
Function MapDateTime(const DateFormatType:string; DateFormat:string;Value:string;ToCds:Boolean): string
Function MapValues( Mapping : string; Value : string) : string
Function MaskDoFormatText( const EditMask : string; const Value : string; Blank : Char) : string
Function MaskGetCharType( const EditMask : string; MaskOffset : Integer) : TMaskCharType
Function MaskGetCurrentDirectives( const EditMask : string; MaskOffset : Integer) : TMaskDirectives
Function MaskGetFldSeparator( const EditMask : string) : Integer
Function MaskGetMaskBlank( const EditMask : string) : Char
Function MaskGetMaskSave( const EditMask : string) : Boolean
Function MaskIntlLiteralToChar( IChar : Char) : Char
Function MaskOffsetToOffset( const EditMask : String; MaskOffset : Integer) : Integer
Function MaskOffsetToWideOffset( const EditMask : String; MaskOffset : Integer) : Integer
Function MaskString( Mask, Value : String) : String
Function Match( const sString : string) : TniRegularExpressionMatchResul
Function Match1( const sString : string; iStart : integer) : TniRegularExpressionMatchResult
Function Matches( const Filename : string) : Boolean
Function MatchesMask( const Filename, Mask : string) : Boolean
Function MatchStr( const AText : string; const AValues : array of string) : Boolean
Function MatchText( const AText : string; const AValues : array of string) : Boolean
Function Max( AValueOne, AValueTwo : Integer) : Integer
function Max(const x,y: Integer): Integer;
Function Max1( const B1, B2 : Shortint) : Shortint;
Function Max2( const B1, B2 : Smallint) : Smallint;
Function Max3( const B1, B2 : Word) : Word;
function Max3(const x,y,z: Integer): Integer;
Function Max4( const B1, B2 : Integer) : Integer;
Function Max5( const B1, B2 : Cardinal) : Cardinal;
Function Max6( const B1, B2 : Int64) : Int64;
Function Max64( const AValueOne, AValueTwo : Int64) : Int64
Function MaxFloat( const X, Y : Float) : Float
Function MaxFloatArray( const B : TDynFloatArray) : Float
Function MaxFloatArrayIndex( const B : TDynFloatArray) : Integer
function MaxIntValue(const Data: array of Integer):Integer)
Function MaxJ( const B1, B2 : Byte) : Byte;
function MaxPath: string;
function MaxValue(const Data: array of Double): Double)
Function MaxCalc( const Formula : string) : Extended  //math expression parser
Procedure MaxCalcF( const Formula : string);   //out to console memo2
function MD5(const fileName: string): string;
Function Mean( const Data : array of Double) : Extended
Function Median( const X : TDynFloatArray) : Float
Function Memory : Pointer
Function MemoryPos( const ASubStr : string; MemBuff : PChar; MemorySize : Integer) : Integer
Function MessageBox(hndl: cardinal; text, caption: string; utype: cardinal): Integer;
function MESSAGEBOX(TEXT,CAPTION:PCHAR;FLAGS:WORD):INTEGER
Function MessageDlg(const Msg:string;DlgType:TMsgDlgType;Buttons:TMsgDlgButtons;HelpCtx:Longint): Integer;
Function MessageDlg1(const Msg:string;DlgType:TMsgDlgType;Buttons:TMsgDlgButtons;HelpCtx:Longint;DefaultButton:TMsgDlgBtn): Integer;
Function MessageDlgPos(const Msg:string;DlgType:TMsgDlgType;Buttons:TMsgDlgButtons;HelpCtx:Longint;X, Y:Integer):Integer;
Function MessageDlgPos1( const Msg : string; DlgType : TMsgDlgType; Buttons : TMsgDlgButtons; HelpCtx : Longint; X, Y : Integer; DefaultButton : TMsgDlgBtn) : Integer;
Function MessageDlgPosHelp( const Msg : string; DlgType : TMsgDlgType; Buttons : TMsgDlgButtons; HelpCtx : Longint; X, Y : Integer; const HelpFileName : string) : Integer;
Function MessageDlgPosHelp1( const Msg : string; DlgType : TMsgDlgType; Buttons : TMsgDlgButtons; HelpCtx : Longint; X, Y : Integer; const HelpFileName : string; DefaultButton : TMsgDlgBtn) : Integer;
Function MibToId( Mib : string) : string
Function MidStr( const AText : AnsiString; const AStart, ACount : Integer) : AnsiString;
Function MidStr( const AText : WideString; const AStart, ACount : Integer) : WideString;
Function microsecondsToCentimeters(mseconds: longint): longint;  //340m/s speed of sound
Function Micros(const Timer:THPTimer;const TimerRunning:Boolean):Int64//TypeS('THPTimer', 'Int64
Function MIDIOut( DeviceID : Cardinal) : IJclMIDIOut
Procedure GetMidiOutputs( const List : TStrings)
Function MIDISingleNoteTuningData( Key : TMIDINote; Frequency : Single) : TSingleNoteTuningData
Function MIDINoteToStr( Note : TMIDINote) : string
Function WinMidiOut( DeviceID : Cardinal) : IJclWinMidiOut
Procedure GetMidiOutputs( const List : TStrings)
Procedure MidiOutCheck( Code : MMResult)
Procedure MidiInCheck( Code : MMResult)
Function MilliSecondOf( const AValue : TDateTime) : Word
Function MilliSecondOfTheDay( const AValue : TDateTime) : LongWord
Function MilliSecondOfTheHour( const AValue : TDateTime) : LongWord
Function MilliSecondOfTheMinute( const AValue : TDateTime) : LongWord
Function MilliSecondOfTheMonth( const AValue : TDateTime) : LongWord
Function MilliSecondOfTheSecond( const AValue : TDateTime) : Word
Function MilliSecondOfTheWeek( const AValue : TDateTime) : LongWord
Function MilliSecondOfTheYear( const AValue : TDateTime) : Int64
Function MilliSecondsBetween( const ANow, AThen : TDateTime) : Int64
Function MilliSecondSpan( const ANow, AThen : TDateTime) : Double
Function Micros( const Timer : THPTimer; const TimerRunning : Boolean) : Int64
Function millis: int64;
Function Min( AValueOne, AValueTwo : Integer) : Integer
Function Min1( const B1, B2 : Shortint) : Shortint;
Function Min2( const B1, B2 : Smallint) : Smallint;
Function Min3( const B1, B2 : Word) : Word;
Function Min4( const B1, B2 : Integer) : Integer;
Function Min5( const B1, B2 : Cardinal) : Cardinal;
Function Min6( const B1, B2 : Int64) : Int64;
Function Min64( const AValueOne, AValueTwo : Int64) : Int64
Function MinClientRect : TRect;
Function MinClientRect1( IncludeScroller : Boolean) : TRect;
Function MinClientRect2( TabCount : Integer; IncludeScroller : Boolean) : TRect;
Function MinFloat( const X, Y : Float) : Float
Function MinFloatArray( const B : TDynFloatArray) : Float
Function MinFloatArrayIndex( const B : TDynFloatArray) : Integer
Function MinimizeName( const Filename : string; Canvas : TCanvas; MaxLen : Integer) : string
Function MinimizeName( const Filename : TFileName; Canvas : TCanvas; MaxLen : Integer) : TFileName
function MinimizeName(const Filename: String; Canvas: TCanvas;MaxLen: Integer): TFileName
Function MinIntValue( const Data : array of Integer) : Integer
function MinIntValue(const Data: array of Integer):Integer)
Function MinJ( const B1, B2 : Byte) : Byte;
Function MinuteOf( const AValue : TDateTime) : Word
Function MinuteOfTheDay( const AValue : TDateTime) : Word
Function MinuteOfTheHour( const AValue : TDateTime) : Word
Function MinuteOfTheMonth( const AValue : TDateTime) : Word
Function MinuteOfTheWeek( const AValue : TDateTime) : Word
Function MinuteOfTheYear( const AValue : TDateTime) : LongWord
Function MinutesBetween( const ANow, AThen : TDateTime) : Int64
Function MinuteSpan( const ANow, AThen : TDateTime) : Double
Function MinValue( const Data : array of Double) : Double
function MinValue(const Data: array of Double): Double)
Function MixerLeftRightToArray( Left, Right : Cardinal) : TDynCardinalArray
Function MMCheck( const MciError : MCIERROR; const Msg : string) : MCIERROR
Function ModFloat( const X, Y : Float) : Float
Function ModifiedJulianDateToDateTime( const AValue : Double) : TDateTime
Function Modify( const Key : string; Value : Integer) : Boolean
Function ModuleCacheID : Cardinal
Function ModuleFromAddr( const Addr : Pointer) : HMODULE
Function MonitorFromPoint( const Point : TPoint; MonitorDefault : TMonitorDefaultTo) : TMonitor
Function MonitorFromRect( const Rect : TRect; MonitorDefault : TMonitorDefaultTo) : TMonitor
Function MonitorFromWindow( const Handle : THandle; MonitorDefault : TMonitorDefaultTo) : TMonitor
Function MonthOf( const AValue : TDateTime) : Word
Function MonthOfTheYear( const AValue : TDateTime) : Word
Function MonthsBetween( const ANow, AThen : TDateTime) : Integer
Function MonthSpan( const ANow, AThen : TDateTime) : Double
Function MonthStr( DateTime : TDateTime) : string
Function MouseCoord( X, Y : Integer) : TGridCoord
Function MOVEBY( DISTANCE : INTEGER) : INTEGER
Function MoveFile( Source, Dest : string; Flags : FILEOP_FLAGS) : Boolean
Function MoveNext : Boolean
Function MSecsToTimeStamp( MSecs : Comp) : TTimeStamp
```

```
1923: function MSecsToTimeStamp(MSecs: Comp): TTimeStamp)
1924: Function Name : string
1925: Function NetPresentValue(const Rate:Extended;const CashFlows:array of
      Double;PaymentTime:TPaymentTime):Extended
1926: function NetworkVolume(DriveChar: Char): string
1927: Function NEWBOTTOMLINE : INTEGER
1928: Function NewCompareNode( Field : TField; Operator : TCANOperator; const Value : Variant) : PExprNode
1929: Function NEWITEM( const ACAPTION : String; ASHORTCUT : TSHORTCUT; ACHECKED, AENABLED : BOOLEAN; AONCLICK :
      TNOTIFYEVENT; HCTX : WORD; const ANAME : String) : TMENUITEM
1930: Function NEWLINE : TMENUITEM
1931: Function NEWMENU( OWNER : TCOMPONENT; const ANAME : STRING; ITEMS : array of TMenuItem) : TMAINMENU
1932: Function NewNode(Kind: TExprNodeKind; Operator:TCANOperator; const Data:Variant; Left,
      Right:PExprNode):PExprNode
1933: Function NEWPOPUPMENU(OWNER:TCOMPONENT;const ANAME:String; ALIGNMENT:TPOPUPALIGNMENT;AUTOPOPUP:BOOLEAN;
      const ITEMS : array of TCMENUITEM) : TPOPUPMENU
1934: Function NewState( eType : TniRegularExpressionStateType) : TniRegularExpressionState
1935: Function NEWSUBMENU(const ACAPT:String;HCTX:WORD;const ANAME:String;ITEMS:array of
      TMenuItem;AENABLED:BOOL): TMENUITEM
1936: Function NEWTOPLINE : INTEGER
1937: Function Next : TIdAuthWhatsNext
1938: Function NextCharIndex( S : String; Index : Integer) : Integer
1939: Function NextRecordSet : TCustomSQLDataSet
1940: Function NextRecordset( var RecordsAffected : Integer) : _Recordset
1941: Function NextSQLToken1( var p : WideChar; out Token : WideString; CurSection : TSQLToken) : TSQLToken;
1942: Function NextToken : Char
1943: Function nextToken : WideString
1944: function NextToken:Char
1945: Function Norm( const Data : array of Double) : Extended
1946: Function NormalizeAngle( const Angle : Extended) : Extended
1947: Function NormalizeBcd( const InBcd : TBcd; var OutBcd : TBcd; const Prec, Scale : Word) : Boolean
1948: Function NormalizeRect( const Rect : TRect) : TRect
1949: function NormalizeRect(const Rect: TRect): TRect;
1950: Function Now : TDateTime
1951: function Now2: tDateTime
1952: Function NumProcessThreads : integer
1953: Function NumThreadCount : integer
1954: Function NthDayOfWeek( const AValue : TDateTime) : Word
1955: Function NtProductType : TNtProductType
1956: Function NtProductTypeString : string
1957: function Null: Variant;
1958: Function NullPoint : TPoint
1959: Function NullRect : TRect
1960: Function Null2Blank(aString:String):String;
1961: Function NumberOfPeriods( const Rate : Extended; Payment : Extended; const PresentValue, FutureValue :
      Extended; PaymentTime : TPaymentTime) : Extended
1962: Function NumIP : integer
1963: function Odd(x: Longint): boolean;
1964: Function OffsetFromUTC : TDateTime
1965: Function OffsetPoint( const P, Offset : TPoint) : TPoint
1966: Function OffsetRect( var Rect : TRect; DX : Integer; DY : Integer) : Boolean
1967: function OffsetRect(var Rect: TRect; DX:Integer; DY:Integer): Boolean)
1968: Function OffsetToMaskOffset( const EditMask : string; Offset : Integer) : Integer
1969: Function OkToChangeFieldAlignment( AField : TField; Alignment : TAlignment) : Boolean
1970: Function OldBCDToCurr( const BCD : TBcd; var Curr : Currency) : Boolean
1971: Function OldCurrToBCD(const Curr:Currency; var BCD:TBcd; Precision:Integer;Decimals:Integer): Boolean
1972: function OpenBit:Integer
1973: Function OpenDatabase : TDatabase
1974: Function OpenDatabase( const DatabaseName : string) : TDatabase
1975: Function OpenGLColorToWinColor( const Red, Green, Blue : Float) : TColor
1976: Function OpenObject( Value : PChar) : Boolean;
1977: Function OpenObject1( Value : string) : Boolean;
1978: Function OpenSession( const SessionName : string) : TSession
1979: Function OpenVolume( const Drive : Char) : THandle
1980: function OrdFourByteToCardinal(AByte1, AByte2, AByte3, AByte4 : Byte): Cardinal
1981: Function OrdFourByteToLongWord( AByte1, AByte2, AByte3, AByte4 : Byte) : LongWord
1982: Function OrdToBinary( const Value : Byte) : string;
1983: Function OrdToBinary1( const Value : Shortint) : string;
1984: Function OrdToBinary2( const Value : Smallint) : string;
1985: Function OrdToBinary3( const Value : Word) : string;
1986: Function OrdToBinary4( const Value : Integer) : string;
1987: Function OrdToBinary5( const Value : Cardinal) : string;
1988: Function OrdToBinary6( const Value : Int64) : string;
1989: Function OSCheck( RetVal : Boolean) : Boolean
1990: Function OSFileTypeToString( const OSFileType : DWORD; const OSFileSubType : DWORD) : string
1991: Function OSIdentToString( const OSIdent : DWORD) : string
1992: Function Output: Text)
1993: Function Overlay( ImageIndex : Integer; Overlay : TOverlay) : Boolean
1994: Function Owner : TCustomListView
1995: function Owner : TPersistent
1996: Function PadInputLiterals( const EditMask : String; const Value : string; Blank : Char) : string
1997: Function PadL( pStr : String; pLth : integer) : String
1998: Function Padl(s : AnyString;I : longInt) : AnyString;
1999: Function PadLCh( pStr : String; pLth : integer; pChr : char) : String
2000: Function PadR( pStr : String; pLth : integer) : String
2001: Function Padr(s : AnyString;I : longInt) : AnyString;
2002: Function PadRCh( pStr : String; pLth : integer; pChr : char) : String
2003: Function PadString( const AString : String; const ALen : Integer; const AChar : Char) : String
2004: Function Padz(s : AnyString;I : longInt) : AnyString;
2005: Function PaethPredictor( a, b, c : Byte) : Byte
```

```
Function PARAMBYNAME( const VALUE : String) : TPARAM
Function ParamByName( const Value : WideString) : TParameter
Function ParamCount: Integer
Function ParamsEncode( const ASrc : string) : string
function ParamStr(Index: Integer): string)
Function ParseDate( const DateStr : string) : TDateTime
Function PARSESQL( SQL : String; DOCREATE : BOOLEAN) : String
Function ParseSQL( SQL : WideString; DoCreate : Boolean) : WideString
Function PathAddExtension( const Path, Extension : string) : string
Function PathAddSeparator( const Path : string) : string
Function PathAppend( const Path, Append : string) : string
Function PathBuildRoot( const Drive : Byte) : string
Function PathCanonicalize( const Path : string) : string
Function PathCommonPrefix( const Path1, Path2 : string) : Integer
Function PathCompactPath(const DC:HDC;const Path:string;const Width:Integer;CmpFmt:TCompactPath):string;
Function PathCompactPath1(const Canv:TCanvas;const Path:string;const Width:Int;CmpFmt:TCompactPath):string;
Function PathEncode( const ASrc : string) : string
Function PathExtractFileDirFixed( const S : AnsiString) : AnsiString
Function PathExtractFileNameNoExt( const Path : string) : string
Function PathExtractPathDepth( const Path : string; Depth : Integer) : string
Function PathGetDepth( const Path : string) : Integer
Function PathGetLongName( const Path : string) : string
Function PathGetLongName2( Path : string) : string
Function PathGetShortName( const Path : string) : string
Function PathIsAbsolute( const Path : string) : Boolean
Function PathIsChild( const Path, Base : AnsiString) : Boolean
Function PathIsDiskDevice( const Path : string) : Boolean
Function PathIsUNC( const Path : string) : Boolean
Function PathRemoveExtension( const Path : string) : string
Function PathRemoveSeparator( const Path : string) : string
Function Payment(Rate:Extended;NPeriods:Int;const PresentVal,
FutureVal:Extended;PaymentTime:TPaymentTime):Extended
Function Peek : Pointer
Function Peek : TObject
function PERFORM(MSG:CARDINAL;WPARAM,LPARAM:LONGINT):LONGINT
Function PeriodPayment(const Rate:Extended;Period,NPeriods:Integer; const PresentValue, FutureValue :
Extended; PaymentTime : TPaymentTime) : Extended
function Permutation(npr, k: integer): extended;
function PermutationInt(npr, k: integer): Int64;
Function PermutationJ( N, R : Cardinal) : Float
Function Pi : Extended;
Function PiE : Extended;
Function PixelsToDialogsX( const Pixels : Word) : Word
Function PixelsToDialogsY( const Pixels : Word) : Word
Function PlaySound(s: pchar; flag,syncflag: integer): boolean;
Function Point( X, Y : Integer) : TPoint
function Point(X, Y: Integer): TPoint)
Function PointAssign( const X, Y : Integer) : TPoint
Function PointDist( const P1, P2 : TPoint) : Double;
function PointDist(const P1,P2: TFloatPoint): Double;
Function PointDist1( const P1, P2 : TFloatPoint) : Double;
function PointDist2(const P1,P2: TPoint): Double;
Function PointEqual( const P1, P2 : TPoint) : Boolean
Function PointIsNull( const P : TPoint) : Boolean
Function PointToLineSegmentDist( const Point, LineP1, LineP2 : TFloatPoint) : Double
Function Poly( const X : Extended; const Coefficients : array of Double) : Extended
Function PortTCPIsOpen(dwPort : Word; ipAddressStr: String): boolean;
Function IsTCPPortOpen(dwPort : Word; ipAddressStr: String): boolean;
Function Pop : Pointer
Function Pop : TObject
Function PopnStdDev( const Data : array of Double) : Extended
Function PopnVariance( const Data : array of Double) : Extended
Function PopulationVariance( const X : TDynFloatArray) : Float
function Pos(SubStr, S: AnyString): Longint;
Function PosEqual( const Rect : TRect) : Boolean
Function PosEx( const SubStr, S : string; Offset : Integer) : Integer
Function PosInSmallIntArray( const ASearchInt : SmallInt; AArray : array of SmallInt) : Integer
Function PosInStrArray(const SearchStr:string;Contents:array of string;const CaseSensitive:Boolean):Integer
Function Post1( AURL : string; const ASource : TStrings) : string;
Function Post2( AURL : string; const ASource : TStream) : string;
Function Post3( AURL : string; const ASource : TIdMultiPartFormDataStream) : string;
Function PostData( const UserData : WideString; const CheckSum : DWORD) : Boolean
Function PostData( const UserData : WideString; const CheckSum : integer): Boolean
function PostMessage(hWnd: HWND; Msg: longword; wParam: longint; lParam: longint): Boolean;
Function Power( const Base, Exponent : Extended) : Extended
Function PowerBig(aval, n:integer): string;
Function PowerIntJ( const X : Float; N : Integer) : Float;
Function PowerJ( const Base, Exponent : Float) : Float;
Function PowerOffOS : Boolean
Function PreformatDateString( Ps : string) : string
Function PresentValue(const Rate:Extend;NPeriods:Int;const Payment,
FutureVal:Extend;PaymentTime:TPaymentTime):Extended
Function PrimeFactors( N : Cardinal) : TDynCardinalArray
Function Printer : TPrinter
Function ProcessPath2( const ABasePath:String; const APath: String; const APathDelim:string): string
Function ProcessResponse : TIdHTTPWhatsNext
Function ProduceContent : string
Function ProduceContentFromStream( Stream : TStream) : string
Function ProduceContentFromString( const S : string) : string
Function ProgIDToClassID(const ProgID: string): TGUID;
Function PromptDataLinkFile( ParentHandle : THandle; InitialFile : WideString) : WideString
Function PromptDataSource( ParentHandle : THandle; InitialString : WideString) : WideString
Function PromptForFileName( var AFileName : string; const AFilter : string; const ADefaultExt : string;
const ATitle : string; const AInitialDir : string; SaveDialog : Boolean) : Boolean
function PromptForFileName(var AFileName: string; const AFilter: string; const ADefaultExt: string;const
ATitle: string; const AInitialDir: string; SaveDialog: Boolean): Boolean)
Function PSScriptNeedFile(Sender:TObject;const OrginFileName:String;var FileName,Output:String):Boolean
Function PtInRect( const Rect : TRect; const P : TPoint) : Boolean
function PtInRect(const Rect: TRect; const P: TPoint): Boolean)
Function Push( AItem : Pointer) : Pointer
Function Push( AObject : TObject) : TObject
Function Put1( AURL : string; const ASource : TStream) : string;
Function Pythagoras( const X, Y : Extended) : Extended
Function queryDLLInterface( var queryList : TStringList) : TStringList
Function queryDLLInterfaceTwo( var queryList : TStringList) : TStringList
Function QueryInterface(const IID: TGUID; out Obj): HResult, CdStdCall
Function queryPerformanceCounter2(mse: int64): int64;
//Function QueryPerformanceCounter(var lpPerformanceCount: Int64): LongBool; stdcall;
//Function QueryPerformanceFrequency(mse: int64): boolean;
Function QueryPerformanceCounter(var lcount: Int64): Boolean; stdcall;
Function QueryPerformanceFrequency(var lfreq: int64): boolean; stdcall;
Procedure QueryPerformanceCounter1(var aC: Int64);
Function QueryPerformanceFrequency1(var freq: int64): boolean;
Function Quote( const ACommand : String) : SmallInt
Function QuotedStr( S : string) : string
Function RadToCycle( const Radians : Extended) : Extended
Function RadToDeg( const Radians : Extended) : Extended
Function RadToDeg( const Value : Extended) : Extended;
Function RadToDeg1( const Value : Double) : Double;
Function RadToDeg2( const Value : Single) : Single;
Function RadToGrad( const Radians : Extended) : Extended
Function RadToGrad( const Value : Extended) : Extended;
Function RadToGrad1( const Value : Double) : Double;
Function RadToGrad2( const Value : Single) : Single;
Function RandG( Mean, StdDev : Extended) : Extended
function Random(const ARange: Integer): Integer;
function random2(a: integer): double
function RandomE: Extended;
function RandomF: Extended;
Function RandomFrom( const AValues : array of string) : string;
Function RandomRange( const AFrom, ATo : Integer) : Integer
function randSeed: longint
Function RawToDataColumn( ACol : Integer) : Integer
Function Read : Char
Function Read( pv : Pointer; cb : Longint; pcbRead : PLongint) : HResult
function Read(Buffer:String;Count:LongInt):LongInt
Function ReadBinaryStream( const Section, Name : string; Value : TStream) : Integer
Function ReadBool( const Section, Ident : string; Default : Boolean) : Boolean
Function ReadCardinal( const AConvert : boolean) : Cardinal
Function ReadChar : Char
Function ReadClient( var Buffer, Count : Integer) : Integer
Function ReadDate( const Section, Name : string; Default : TDateTime) : TDateTime
Function ReadDateTime( const Section, Name : string; Default : TDateTime) : TDateTime
Function ReadFloat( const Section, Name : string; Default : Double) : Double
Function ReadFromStack(const ARaiseExceptfDisconnected:Bool;ATimeout:Int;const ARaiseExceptonTimeout:
Boolean):Integer
Function ReadInteger( const AConvert : boolean) : Integer
Function ReadInteger( const Section, Ident : string; Default : Longint) : Longint
Function ReadLn : string
Function ReadLn( ATerminator : string; const ATimeout : Integer; AMaxLineLength : Integer) : string
function Readln(question: string): string;
Function ReadLnWait( AFailCount : Integer) : string
Function ReadReg(Base: HKEY; KeyName, ValueName: string): string;
Function ReadRegistry(Base: HKEY; KeyName, ValueName: string): string;
Function ReadSmallInt( const AConvert : boolean) : SmallInt
Function ReadString( const ABytes : Integer) : string
Function ReadString( const Section, Ident, Default : string) : string
Function ReadString( Count : Integer) : string
Function ReadTime( const Section, Name : string; Default : TDateTime) : TDateTime
Function ReadTimeStampCounter : Int64
Function RebootOS : Boolean
Function Receive( ATimeOut : Integer) : TReplyStatus
Function ReceiveBuf( var Buf, Count : Integer) : Integer
Function ReceiveLength : Integer
Function ReceiveText : string
Function ReceiveSerialData(var Data: TByteArray; DataSize: cardinal): cardinal
Function ReceiveSerialText: string
Function RecodeDate( const AValue : TDateTime; const AYear, AMonth, ADay : Word) : TDateTime
Function RecodeDateTime(const AValue:TDateTime;const AYear,AMonth,ADay,AHr,AMin,ASec,
AMilliSec:Word):TDateTime
Function RecodeDay( const AValue : TDateTime; const ADay : Word) : TDateTime
Function RecodeHour( const AValue : TDateTime; const AHour : Word) : TDateTime
Function RecodeMilliSecond( const AValue : TDateTime; const AMilliSecond : Word) : TDateTime
Function RecodeMinute( const AValue : TDateTime; const AMinute : Word) : TDateTime
Function RecodeMonth( const AValue : TDateTime; const AMonth : Word) : TDateTime
Function RecodeSecond( const AValue : TDateTime; const ASecond : Word) : TDateTime
Function RecodeTime( const AValue: TDateTime;const AHour,AMinute,ASecond,AMilliSecond:Word):TDateTime
Function RecodeYear( const AValue : TDateTime; const AYear : Word) : TDateTime
Function Reconcile( const Results : OleVariant) : Boolean
Function Rect( Left, Top, Right, Bottom : Integer) : TRect
function Rect(ALeft: Integer; ATop: Integer; ARight: Integer; ABottom: Integer): TRect)
Function Rect2( const ATopLeft, ABottomRight : TPoint) : TRect;
Function RectAssign( const Left, Top, Right, Bottom : Integer) : TRect
Function RectAssignPoints( const TopLeft, BottomRight : TPoint) : TRect
Function RectBounds( const Left, Top, Width, Height : Integer) : TRect
Function RectCenter( const R : TRect) : TPoint
Function RectEqual( const R1, R2 : TRect) : Boolean
Function RectHeight( const R : TRect) : Integer
Function RectIncludesPoint( const R : TRect; const Pt : TPoint) : Boolean
Function RectIncludesRect( const R1, R2 : TRect) : Boolean
Function RectIntersection( const R1, R2 : TRect) : TRect
Function RectIntersectRect( const R1, R2 : TRect) : Boolean
Function RectIsEmpty( const R : TRect) : Boolean
Function RectIsNull( const R : TRect) : Boolean
Function RectIsSquare( const R : TRect) : Boolean
Function RectIsValid( const R : TRect) : Boolean
Function RectsAreValid( R : array of TRect) : Boolean
Function RectUnion( const R1, R2 : TRect) : TRect
Function RectWidth( const R : TRect) : Integer
Function RedComponent( const Color32 : TColor32) : Integer
Function Refresh : Boolean
Function RefStringListCopy(aRefArray:TStringlist):TStringList;
Function RegisterConversionFamily( const ADescription : string) : TConvFamily
Function RegisterConversionType( AConvTypeInfo : TConvTypeInfo; out AType : TConvType) : Boolean;
Function RegisterConversionType(const AFam:TConvFamil;const ADescr:string;const AFact:Double):TConvType
Function RegistryRead(keyHandle: Longint; keyPath, myField: String): string;
Function ReleaseDC(hdwnd: HWND; hdc: HDC): integer;
Function ReleaseHandle : HBITMAP
Function ReleaseHandle : HENHMETAFILE
Function ReleaseHandle : HICON
Function ReleasePalette : HPALETTE
Function RemainderFloat( const X, Y : Float) : Float
Function Remove( AClass : TClass) : Integer
Function Remove( AComponent : TComponent) : Integer
Function Remove( AItem : Integer) : Integer
Function Remove( AItem : Pointer) : Pointer
Function Remove( AItem : TObject) : TObject
Function Remove( AObject : TObject) : Integer
Function RemoveBackslash( const PathName : string) : string
Function RemoveDF( aString : String) : String  //removes thousand separator
Function RemoveDir( Dir : string) : Boolean
function RemoveDir(const Dir: string): Boolean)
Function RemoveDirectory(PathName: PChar): WordBool; stdcall;
Function RemoveFileExt( const FileName : string) : string
Function RemoveHeaderEntry( AHeader, AEntry : string) : string
Function RenameFile( OldName, NewName : string) : Boolean
function RenameFile(const OldName: string; const NewName: string): Boolean)
Function ReplaceStr( const AText, AFromText, AToText : string) : string
Function ReplaceText( const AText, AFromText, AToText : string) : string
Function Replicate(c : char;I : longInt) : String;
Function Request : TWebRequest
Function ResemblesText( const AText, AOther : string) : Boolean
Function Reset : Boolean
function Reset2(mypath: string):string;
Function ResInstLoad(Instance:THandle;ResType:TResType; const Name:string;MaskColor: TColor): Boolean
Function ResourceLoad( ResType : TResType; const Name : string; MaskColor : TColor) : Boolean
Function Response : TWebResponse
Function ResumeSupported : Boolean
Function RETHINKHOTKEYS : BOOLEAN
Function RETHINKLINES : BOOLEAN
Function Retrieve( const MsgNum : Integer; AMsg : TIdMessage) : Boolean
Function RetrieveCurrentDir : string
Function RetrieveDeltas( const cdsArray : array of TClientDataset) : Variant
Function RetrieveHeader( const MsgNum : Integer; AMsg : TIdMessage) : Boolean
Function RetrieveMailBoxSize : integer
Function RetrieveMsgSize( const MsgNum : Integer) : Integer
Function RetrieveProviders( const cdsArray : array of TClientDataset) : Variant
Function RetrieveRaw( const MsgNum : Integer; const Dest : TStrings) : boolean
Function ReturnMIMEType( var MediaType, EncType : String) : Boolean
Function ReverseBits( Value : Byte) : Byte;
Function ReverseBits1( Value : Shortint) : Shortint;
Function ReverseBits2( Value : Smallint) : Smallint;
Function ReverseBits3( Value : Word) : Word;
Function ReverseBits4( Value : Cardinal) : Cardinal;
Function ReverseBits4( Value : Integer) : Integer;
Function ReverseBits5( Value : Int64) : Int64;
Function ReverseBytes( Value : Word) : Word;
Function ReverseBytes1( Value : Smallint) : Smallint;
Function ReverseBytes2( Value : Integer) : Integer;
Function ReverseBytes3( Value : Cardinal) : Cardinal;
Function ReverseBytes4( Value : Int64) : Int64;
Function ReverseString( const AText : string) : string
Function ReverseDNSLookup(const IPAddrs:String;DNSServer:String;Timeout,Retries:Int;var
HostName:String):Bool;
Function Revert : HResult
Function RGB(R,G,B: Byte): TColor;
Function RGB2BGR( const Color : TColor) : TColor
Function RGB2TColor( R, G, B : Byte) : TColor
Function RGBToWebColorName( RGB : Integer) : string
Function RGBToWebColorStr( RGB : Integer) : string
Function RgbToHtml( Value : TColor) : string
Function HtmlToRgb(const Value: string): TColor;
Function RightStr( const AStr : String; Len : Integer) : String
Function RightStr( const AText : AnsiString; const ACount : Integer) : AnsiString;
Function RightStr( const AText : WideString; const ACount : Integer) : WideString;
Function ROL( AVal : LongWord; AShift : Byte) : LongWord
Function ROR( AVal : LongWord; AShift : Byte) : LongWord
Function RotatePoint( Point : TFloatPoint; const Center : TFloatPoint; const Angle : Float) : TFloatPoint
function RotatePoint(Point: TFloatPoint; const Center: TFloatPoint; const Angle: Double): TFloatPoint;
Function Round(e : Extended) : Longint;
Function Round64(e: extended): Int64;
Function RoundAt( const Value : string; Position : SmallInt) : string
Function RoundFrequency( const Frequency : Integer) : Integer
Function RoundInt( Value : Integer; StepSize : Integer) : Integer
Function RoundPoint( const X, Y : Double) : TPoint
Function RoundRect( const ALeft, ATop, ARight, ABottom : Double) : TRect
Function RowCount : Integer
Function RowRequest( const Row : OleVariant; RequestType : Integer; var OwnerData : OleVariant): OleVariant
Function RowRequest( Row : OleVariant; Options : TFetchOptions) : OleVariant
Function RPos( const ASub, AIn : String; AStart : Integer) : Integer
Function RRot( const Value : Byte; const Count : TBitRange) : Byte;
Function RRot1( const Value : Word; const Count : TBitRange) : Word;
Function RRot2( const Value : Integer; const Count : TBitRange) : Integer;
Function RunDLL32(const ModuleNa,FuncName,CmdLine:string;WaitForCompletion:Bool;CmdShow:Integer):Boolean
Function RunningProcessesList( const List : TStrings; FullPath : Boolean) : Boolean
Function S_AddBackSlash( const ADirName : string) : string
Function S_AllTrim( const cStr : string) : string
Function S_AtRepl( const cAT, cStr, cRepl : string) : string
Function S_Cut( const cStr : string; const iLen : integer) : string
Function S_DecryptCRC32( const crc : string; StartKey, MultKey, AddKey : integer) : integer
Function S_DirExists( const ADir : string) : Boolean
Function S_Empty( const cStr : string) : boolean
Function S_EncryptCRC32( const crc : LongWORD; StartKey, MultKey, AddKey : integer) : string
Function S_LargeFontsActive : Boolean
Function S_LimitDigits( AValue : Extended; ANumDigits : Integer) : Extended
Function S_LTrim( const cStr : string) : string
Function S_ReadNextTextLineFromStream( stream : TStream) : string
Function S_RepeatChar( const iLen : integer; const AChar : Char) : String
Function S_ReplFirst( const cAT, cStr, cRepl : string) : string
Function S_RoundDecimal( AValue : Extended; APlaces : Integer) : Extended
Function S_RTrim( const cStr : string) : string
Function S_RTrimCopy( const cStr : string; iPos, iLen : integer) : string
//Type TS_ShellExecuteCmd = (seCmdOpen,seCmdPrint,seCmdExplore);
Function S_ShellExecute( aFilename : string; aParameters : string; aCommand : TS_ShellExecuteCmd) : string
Function S_Space( const iLen : integer) : String
Function S_StrBlanks( const cStr : string; const iLen : integer) : string
Function S_StrBlanksCuttooLong( const cStr : string; const iLen : integer) : string
Function S_StrCRC32( const Text : string) : LongWORD
Function S_StrDecrypt96( const InString : string; StartKey, MultKey, AddKey : Integer) : string
Function S_StrEncrypt96( const InString : string; StartKey, MultKey, AddKey : Integer) : string
Function S_StringtoUTF_8( const AString : string) : string
Function S_StrLBlanks( const cStr : string; const iLen : integer) : string
function S_StrToReal(const cStr: string; var R: Double): Boolean
Function S_TokenEnd( cBuffer : PChar; lEmptyToken : boolean) : boolean
Function S_TokenNext( cBuffer : PChar; lEmptyToken : boolean) : string
Function S_UTF_8ToString( const AString : string) : string
Function S_WBox( const AText : string) : integer
Function SameDate( const A, B : TDateTime) : Boolean
function SameDate(const A, B: TDateTime): Boolean;
Function SameDateTime( const A, B : TDateTime) : Boolean
function SameDateTime(const A, B: TDateTime): Boolean;
Function SameFileName( S1, S2 : string) : Boolean
Function SameText( S1, S2 : string) : Boolean
function SameText(const S1: string; const S2: string): Boolean)
Function SameTime( const A, B : TDateTime) : Boolean
function SameTime(const A, B: TDateTime): Boolean;
function SameValue(const A, B: Extended; Epsilon: Extended): Boolean //overload;
function SameValue1(const A, B: Double; Epsilon: Double): Boolean //overload;
function SameValue2(const A, B: Single; Epsilon: Single): Boolean //overload;
Function SampleVariance( const X : TDynFloatArray) : Float
Function Sar( const Value : Shortint; const Count : TBitRange) : Shortint;
Function Sar1( const Value : Smallint; const Count : TBitRange) : Smallint;
Function Sar2( const Value : Integer; const Count : TBitRange) : Integer;
Function SaveToFile( const AFileName : TFileName) : Boolean
Function SaveAsExcelFile(AGrid: TStringGrid; ASheetName, AFileName: string; open: boolean): Boolean;
Function SaveAsExcel(aGrid: TStringGrid; aSheetName, aFileName: string; openexcel: boolean): Boolean;
Function ScanF(const aformat: String; const args: array of const): string;
Function SCREENTOCLIENT(POINT:TPOINT):TPOINT
Function SearchBuf(Buf:PChar; BufLen:Integer;SelStart,SelLength:Integer;SearchString:String;Options:
TStringSearchOptions):PChar
Function SearchBuf2(Buf: String;SelStart,SelLength:Integer; SearchString:
String;Options:TStringSearchOptions):Integer;
function SearchRecattr: integer;
function SearchRecExcludeAttr: integer;
Function SearchRecFileSize64( const SearchRec : TSearchRec) : Int64
function SearchRecname: string;
function SearchRecsize: integer;
function SearchRecTime: integer;
Function Sec( const X : Extended) : Extended
Function Secant( const X : Extended) : Extended
Function SecH( const X : Extended) : Extended
Function SecondOf( const AValue : TDateTime) : Word
Function SecondOfTheDay( const AValue : TDateTime) : LongWord
Function SecondOfTheHour( const AValue : TDateTime) : Word
Function SecondOfTheMinute( const AValue : TDateTime) : Word
Function SecondOfTheMonth( const AValue : TDateTime) : LongWord
Function SecondOfTheWeek( const AValue : TDateTime) : LongWord
Function SecondOfTheYear( const AValue : TDateTime) : LongWord
Function SecondsBetween( const ANow, AThen : TDateTime) : Int64
Function SecondSpan( const ANow, AThen : TDateTime) : Double
Function SectionExists( const Section : string) : Boolean
Function Seek( const KeyValues : Variant; SeekOption : TSeekOption) : Boolean
Function Seek( dlibMove : Longint; dwOrigin : Longint; out libNewPosition : Largeint) : HResult
function Seek(Offset:LongInt;Origin:Word):LongInt
Function SelectDirectory( var Directory : string; Options : TSelectDirOpts; HelpCtx:Longint): Boolean;
Function SelectDirectory1( const Caption : string; const Root : WideString; var Directory : string; Options : TSelectDirExtOpts; Parent : TWinControl) : Boolean;
Function SelectImage( var AFileName : string; const Extensions, Filter : string) : Boolean
function SendAppMessage(Msg: Cardinal; WParam, LParam: Longint): Longint
Function SendBuf( var Buf, Count : Integer) : Integer
Function SendCmd( const AOut : string; const AResponse : SmallInt) : SmallInt;
Function SendCmd1( const AOut : string; const AResponse : array of SmallInt) : SmallInt;
Function SendKey( AppName : string; Key : Char) : Boolean
function SendMessage(hWnd: HWND; Msg: longword; wParam: longint; lParam: longint): Boolean;
Function SendStream( AStream : TStream) : Boolean
Function SendStreamThenDrop( AStream : TStream) : Boolean
Function SendText( const S : string) : Integer
Function SendSerialData(Data: TByteArray; DataSize: cardinal): cardinal
Function SendSerialText(Data: String): cardinal
Function Sent : Boolean
Function ServicesFilePath: string
Function SetAlpha( const Color32 : TColor32; NewAlpha : Integer) : TColor32
Function SetBit( const Value : Byte; const Bit : TBitRange) : Byte;
Function SetBit1( const Value : Shortint; const Bit : TBitRange) : Shortint;
Function SetBit2( const Value : Smallint; const Bit : TBitRange) : Smallint;
Function SetBit3( const Value : Word; const Bit : TBitRange) : Word;
Function SetBit4( const Value : Cardinal; const Bit : TBitRange) : Cardinal;
Function SetBit4( const Value : Integer; const Bit : TBitRange) : Integer;
Function SetBit5( const Value : Int64; const Bit : TBitRange) : Int64;
Function SetClipboard( NewClipboard : TClipboard) : TClipboard
Function SetColorBlue( const Color : TColor; const Blue : Byte) : TColor
Function SetColorFlag( const Color : TColor; const Flag : Byte) : TColor
Function SetColorGreen( const Color : TColor; const Green : Byte) : TColor
Function SetColorRed( const Color : TColor; const Red : Byte) : TColor
Function SetCurrentDir( Dir : string) : Boolean
function SetCurrentDir(const Dir: string): Boolean)
Function SetCurrentDirectory(PathName: PChar): WordBool; stdcall;
Function SetDirCreation( const DirName : string; const DateTime : TDateTime) : Boolean
Function SetDirLastAccess( const DirName : string; const DateTime : TDateTime) : Boolean
Function SetDirLastWrite( const DirName : string; const DateTime : TDateTime) : Boolean
Function SetDisplayResolution( const XRes, YRes : DWORD) : Longint
Function SetEndOfFile(Handle: Integer): LongBool; stdcall;
Function SetEnvironmentVar( const Name, Value : string) : Boolean
Function SetErrorProc( ErrorProc : TSocketErrorProc) : TSocketErrorProc
Function SetFileCreation( const FileName : string; const DateTime : TDateTime) : Boolean
Function SetFileLastAccess( const FileName : string; const DateTime : TDateTime) : Boolean
Function SetFileLastWrite( const FileName : string; const DateTime : TDateTime) : Boolean
Function SetFileTimeStamp( const FileName : string; TimeStamp : Integer) : Boolean
function SETFOCUSEDCONTROL(CONTROL:TWINCONTROL):BOOLEAN
Function SetLocalTime( Value : TDateTime) : boolean
Function SetPrecisionTolerance( NewTolerance : Float) : Float
Function SetPrinter( NewPrinter : TPrinter) : TPrinter
Function SetRGBValue( const Red, Green, Blue : Byte) : TColor
Function SetSequence( S, Localizar, Substituir : shortstring) : shortstring
Function SetSize( libNewSize : Longint) : HResult
Function SetUserObjectFullAccess( hUserObject : THandle) : Boolean
Function Sgn( const X : Extended) : Integer
function SHA1(const fileName: string): string;
function SHA256(astr: string; amode: char): string)
function SHA512(astr: string; amode: char): string)
Function ShareMemoryManager : Boolean
function ShellExecute(hWnd:HWND;Operation,FileN,Parameters,Dir:string;ShowCmd:Integer):integer;stdcall;
function Shellexecute2(hwnd: HWND; const FileName: string):integer; stdcall;
Function ShellExecute3(aFilename: string; aParameters: string; aCommand:TS_ShellExecuteCmd): string;
Function SHORTCUT( KEY : WORD; SHIFT : TSHIFTSTATE) : TSHORTCUT
Function SHORTCUTTOTEXT( SHORTCUT : TSHORTCUT) : String
function ShortDateFormat: string;
Function ShortenString(const DC:HDC;const S:WideString;const Width:Int;const RTL:Bool;EllipsisWidth:Int):WideString
function ShortTimeFormat: string;
function SHOWMODAL:INTEGER
function ShowWindow(C1: HWND; C2: integer): boolean;
procedure ShowMemory     //in Dialog
function ShowMemory2: string;
Function ShutDownOS : Boolean
```

```
2439: Function Signe( const X, Y : Extended) : Extended
2440: Function Sign( const X : Extended) : Integer
2441: Function Sin(e : Extended) : Extended;
2442: Function sinc( const x : Double) : Double
2443: Function SinJ( X : Float) : Float
2444: Function Size( const AFileName : String) : Integer
2445: function SizeOf: Longint;
2446: Function SizeofResource( ModuleHandle : HMODULE; ResHandle : TResourceHandle) : Integer
2447: Function SlashSep(const Path, S: String): String
2448: Function SLNDepreciation( const Cost, Salvage : Extended; Life : Integer) : Extended
2449: Function SleepEx( dwMilliseconds : DWORD; bAlertable : BOOL) : DWORD
2450: Function SmallPoint(X, Y: Integer): TSmallPoint)
2451: Function Soundex( const AText : string; ALength : TSoundexLength) : string
2452: Function SoundexCompare( const AText, AOther : string; ALength : TSoundexLength) : Integer
2453: Function SoundexInt( const AText : string; ALength : TSoundexIntLength) : Integer
2454: Function SoundexProc( const AText, AOther : string) : Boolean
2455: Function SoundexSimilar( const AText, AOther : string; ALength : TSoundexLength) : Boolean
2456: Function SoundexWord( const AText : string) : Word
2457: Function SourcePos : Longint
2458: function SourcePos:LongInt
2459: Function Split0( Str : string; const substr : string) : TStringList
2460: Procedure SplitNameValue( const Line : string; var Name, Value : string)
2461: Function SQLRequiresParams( const SQL : WideString) : Boolean
2462: Function Sqr(e : Extended) : Extended;
2463: Function Sqrt(e : Extended) : Extended;
2464: Function StartIP : String
2465: Function StartPan( WndHandle : THandle; AControl : TControl) : Boolean
2466: Function StartOfADay( const AYear, AMonth, ADay : Word) : TDateTime;
2467: Function StartOfADay1( const AYear, ADayOfYear : Word) : TDateTime;
2468: Function StartOfAMonth( const AYear, AMonth : Word) : TDateTime
2469: Function StartOfAWeek( const AYear, AWeekOfYear : Word; const ADayOfWeek : Word) : TDateTime
2470: Function StartOfAYear( const AYear : Word) : TDateTime
2471: Function StartOfTheDay( const AValue : TDateTime) : TDateTime
2472: Function StartOfTheMonth( const AValue : TDateTime) : TDateTime
2473: Function StartOfTheWeek( const AValue : TDateTime) : TDateTime
2474: Function StartOfTheYear( const AValue : TDateTime) : TDateTime
2475: Function StartsStr( const ASubText, AText : string) : Boolean
2476: Function StartsText( const ASubText, AText : string) : Boolean
2477: Function StartsWith( const ANSIStr, APattern : String) : Boolean
2478: Function StartsWith( const str : string; const sub : string) : Boolean
2479: Function StartsWithACE( const ABytes : TIdBytes) : Boolean
2480: Function StatusString( StatusCode : Integer) : string
2481: Function StdDev( const Data : array of Double) : Extended
2482: Function Stop : Float
2483: Function StopCount( var Counter : TJclCounter) : Float
2484: Function StoreColumns : Boolean
2485: Function StrAfter( const sString : string; const sDelimiters : string) : string;
2486: Function StrAfter1( const sString : string; const sDelimiters : string; out cDelimiter : char) : string;
2487: Function StrAlloc( Size : Cardinal) : PChar
2488: function StrAlloc(Size: Cardinal): PChar)
2489: Function StrBefore( const sString : string; const sDelimiters : string) : string;
2490: Function StrBefore1( const sString : string; const sDelimiters:string; out cDelimiter:char): string;
2491: Function StrBufSize( Str : PChar) : Cardinal
2492: function StrBufSize(const Str: PChar): Cardinal)
2493: Function StrByteType( Str : PChar; Index : Cardinal) : TMbcsByteType
2494: function StrByteType(Str: PChar; Index: Cardinal): TMbcsByteType)
2495: Function StrCat( Dest : PChar; Source : PChar) : PChar
2496: function StrCat(Dest: PChar; const Source: PChar): PChar)
2497: Function StrCharLength( Str : PChar) : Integer
2498: Function StrComp( Str1, Str2 : PChar) : Integer
2499: function StrComp(const Str1: PChar; const Str2: PChar): Integer)
2500: Function StrCopy( Dest : PChar; Source : PChar) : PChar
2501: function StrCopy(Dest: PChar; const Source: PChar): PChar)
2502: Function Stream_to_AnsiString( Source : TStream) : ansistring
2503: Function Stream_to_Base64( Source : TStream) : ansistring
2504: Function Stream_to_decimalbytes( Source : TStream) : string
2505: Function Stream2WideString( oStream : TStream) : WideString
2506: Function StreamtoAnsiString( Source : TStream) : ansistring
2507: Function StreamToByte( Source : TStream) : string
2508: Function StreamToDecimalbytes( Source : TStream) : string
2509: Function StreamtoOrd( Source : TStream) : string
2510: Function StreamToString( Source : TStream) : string
2511: Function StrECopy( Dest : PChar; Source : PChar) : PChar
2512: Function StrEmpty( const sString : string) : boolean
2513: Function StrEnd( Str : PChar) : PChar
2514: function StrEnd(const Str: PChar): PChar)
2515: Function StrFilter( const sString : string; xValidChars : TCharSet) : string
2516: Function StrFmt(Buffer, Format: PChar; const Args: array of const): PChar)
2517: Function StrGet(var S : String; I : Integer) : Char;
2518: Function StrGet2(S : String; I : Integer) : Char;
2519: Function StrHasPrefix( const sString : string; const sPrefix : string) : boolean
2520: Function StrHasSuffix( const sString : string; const sSuffix : string) : boolean
2521: Function StrHtmlDecode( const AStr : String) : String
2522: Function StrHtmlEncode( const AStr : String) : String
2523: Function StrToBytes(const Value: String): TBytes;
2524: Function StrIComp( Str1, Str2 : PChar) : Integer
2525: Function StringOfChar(c : char;I : longInt) : String;
2526: Function StringOfChar2( ch : WideChar; Count : Integer) : WideString;
2527: Function StringPad(InputStr,FillChar: String; StrLen:Integer; StrJustify:Boolean): String;
```

```
Function StringRefCount(const s: String): integer;
Function StringReplace( S, OldPattern, NewPattern : string; Flags : TReplaceFlags) : string
Function JStringReplace( const S, OldPattern, NewPattern : string; Flags : TReplaceFlags) : string
Function StringReplace(const SourceString, OldPattern,NewPattern:string; Flags:TReplaceFlags): string;
Function StringRemove( const S, Pattern : string; Flags : TReplaceFlags) : string
Function StringToBoolean( const Ps : string) : Boolean
function StringToColor(const S: string): TColor)
function StringToCursor(const S: string): TCursor;
function StringToGUID(const S: string): TGUID)
Function StringTokenizer( const str : string; const delim : string) : IStringTokenizer
Function StringToStringArray( const str : string; const delim : string) : TStringDynArray
Function StringWidth( S : string) : Integer
Function StrInternetToDateTime( Value : string) : TDateTime
Function StrIsDateTime( const Ps : string) : Boolean
Function StrIsFloatMoney( const Ps : string) : Boolean
Function StrIsInteger( const S : string) : Boolean
Function StrLCat( Dest : PChar; Source : PChar; MaxLen : Cardinal) : PChar
Function StrLComp( Str1, Str2 : PChar; MaxLen : Cardinal) : Integer
Function StrLCopy( Dest : PChar; Source : PChar; MaxLen : Cardinal) : PChar
Function StrLen( Str : PChar) : Cardinal
function StrLen(const Str: PChar): Cardinal)
Function StrLessPrefix( const sString : string; const sPrefix : string) : string
Function StrLessSuffix( const sString : string; const sSuffix : string) : string
Function StrLIComp( Str1, Str2 : PChar; MaxLen : Cardinal) : Integer
Function StrLower( Str : PChar) : PChar
Function StrMove( Dest : PChar; Source : PChar; Count : Cardinal) : PChar
function StrMove(Dest: PChar; const Source: PChar; Count: Cardinal): PChar)
Function StrNew( Str : PChar) : PChar
function StrNew(const Str: PChar): PChar)
Function StrNextChar( Str : PChar) : PChar
Function StrPad( const sString : string; const sPad : string; const iLength : integer) : string
Function StrParse( var sString : string; const sDelimiters : string) : string;
Function StrParse1( var sString : string; const sDelimiters : string; out cDelimiter : char) : string;
Function StrPas( Str : PChar) : string
function StrPas(const Str: PChar): string)
Function StrPCopy( Dest : PChar; Source : string) : PChar
function StrPCopy(Dest: PChar; const Source: string): PChar)
Function StrPLCopy( Dest : PChar; Source : string; MaxLen : Cardinal) : PChar
Function StrPos( Str1, Str2 : PChar) : PChar
Function StrScan(const Str: PChar; Chr: Char): PChar)
Function StrRScan(const Str: PChar; Chr: Char): PChar)
Function StrToBcd( const AValue : string) : TBcd
Function StrToBool( S : string) : Boolean
Function StrToBoolDef( S : string; Default : Boolean) : Boolean
Function StrToCard( const AStr : String) : Cardinal
Function StrToConv( AText : string; out AType : TConvType) : Double
Function StrToCurr( S : string) : Currency;
function StrToCurr(const S: string): Currency)
Function StrToCurrDef( S : string; Default : Currency) : Currency;
Function StrToDate( S : string) : TDateTime;
function StrToDate(const s: string): TDateTime;
Function StrToDateDef( S : string; Default : TDateTime) : TDateTime;
Function StrToDateTime( S : string) : TDateTime;
function StrToDateTime(const S: string): TDateTime)
Function StrToDateTimeDef( S : string; Default : TDateTime) : TDateTime;
Function StrToDay( const ADay : string) : Byte
Function StrToFloat( S : string) : Extended;
function StrToFloat(s: String): Extended;
Function StrToFloatDef( S : string; Default : Extended) : Extended;
function StrToFloatDef(const S: string; const Default: Extended): Extended)
Function StrToFloat( S : string) : Extended;
Function StrToFloat2( S : string; FormatSettings : TFormatSettings) : Extended;
Function StrToFloatDef( S : string; Default : Extended) : Extended;
Function StrToFloatDef2(S: string; Default: Extended;FormatSettings:TFormatSettings): Extended;
Function StrToCurr( S : string) : Currency;
Function StrToCurr2( S : string; FormatSettings : TFormatSettings) : Currency;
Function StrToCurrDef( S : string; Default : Currency) : Currency;
Function StrToCurrDef2( S : string; Default : Currency; FormatSettings : TFormatSettings) : Currency;
Function StrToTime2( S : string; FormatSettings : TFormatSettings) : TDateTime;
Function StrToTimeDef( S : string; Default : TDateTime) : TDateTime;
Function StrToTimeDef2( S : string; Default : TDateTime; FormatSettings:TFormatSettings):TDateTime;
Function TryStrToTime( S : string; Value : TDateTime) : Boolean;
Function StrToDateTime( S : string) : TDateTime;
Function StrToDateTime2( S : string; FormatSettings : TFormatSettings) : TDateTime;
Function StrToDateTimeDef( S : string; Default : TDateTime) : TDateTime;
Function StrToFloatRegionalIndependent(aValue: String;aDecimalSymbol:Char;aDigitGroupSymbol:Char): Extended
Function StrToInt( S : string) : Integer
function StrToInt(s: String): Longint;
Function StrToInt64( S : string) : Int64
function StrToInt64(s: String): int64;
Function StrToInt64Def( S : string; Default : Int64) : Int64
function StrToInt64Def(const S: string; const Default: Int64):Int64)
Function StrToIntDef( S : string; Default : Integer) : Integer
function StrToIntDef(const S: string; Default: Integer): Integer)
function StrToIntDef(s: String; def: Longint): Longint;
Function StrToMonth( const AMonth : string) : Byte
Function StrToTime( S : string) : TDateTime;
function StrToTime(const S: string): TDateTime)
Function StrToTimeDef( S : string; Default : TDateTime) : TDateTime;
Function StrToWord( const Value : String) : Word
Function StrToXmlDate( const DateStr : string; const Format : string) : string
Function StrToXmlDateTime( const DateStr : string; const Format : string) : string
Function StrToXmlTime( const TimeStr : string; const Format : string) : string
Function StrUpper( Str : PChar) : PChar
Function StuffString( const AText : string; AStart, ALength : Cardinal; const ASubText : string) : string
Function Sum( const Data : array of Double) : Extended
Function SumFloatArray( const B : TDynFloatArray) : Float
Function SumInt( const Data : array of Integer) : Integer
Function SumOfSquares( const Data : array of Double) : Extended
Function SumPairProductFloatArray( const X, Y : TDynFloatArray) : Float
Function SumSquareDiffFloatArray( const B : TDynFloatArray; Diff : Float) : Float
Function SumSquareFloatArray( const B : TDynFloatArray) : Float
Function Supports( CursorOptions : TCursorOptions) : Boolean
Function SupportsClipboardFormat( AFormat : Word) : Boolean
Function SwapWord(w : word): word)
Function SwapInt(i : integer): integer)
Function SwapLong(L : longint): longint)
Function Swap(i : integer): integer)
Function SYDDepreciation( const Cost, Salvage : Extended; Life, Period : Integer) : Extended
Function SyncTime : Boolean
Function SysErrorMessage( ErrorCode : Integer) : string
function SysErrorMessage(ErrorCode: Integer): string)
Function SystemTimeToDateTime( SystemTime : TSystemTime) : TDateTime
function SystemTimeToDateTime(const SystemTime: TSystemTime): TDateTime;
Function SysStringLen(const S: WideString): Integer; stdcall;
Function TabRect( Index : Integer) : TRect
Function Tan( const X : Extended) : Extended
Function TaskMessageDlg(const Title,
Msg:string;DlgType:TMsgDlgType;Buttons:TMsgDlgButtons;HelpCtx:Longint): Integer;
Function TaskMessageDlg1( const Title, Msg : string; DlgType : TMsgDlgType; Buttons : TMsgDlgButtons;
HelpCtx : Longint; DefaultButton : TMsgDlgBtn) : Integer;
Function TaskMessageDlgPos( const Title, Msg : string; DlgType : TMsgDlgType; Buttons : TMsgDlgButtons;
HelpCtx : Longint; X, Y : Integer) : Integer;
Function TaskMessageDlgPos1( const Title, Msg : string; DlgType : TMsgDlgType; Buttons : TMsgDlgButtons;
HelpCtx : Longint; X, Y : Integer; DefaultButton : TMsgDlgBtn) : Integer;
Function TaskMessageDlgPosHelp( const Title, Msg : string; DlgType : TMsgDlgType; Buttons :
TMsgDlgButtons; HelpCtx : Longint; X, Y : Integer; const HelpFileName : string) : Integer;
Function TaskMessageDlgPosHelp1(const Title, Msg:string;DlgType: TMsgDlgType; Buttons : TMsgDlgButtons;
HelpCtx:Longint; X,Y: Integer;const HelpFileName:string;DefaultButton:TMsgDlgBtn): Integer;
Function TenToY( const Y : Float) : Float
Function TerminateApp( ProcessID : DWORD; Timeout : Integer) : TJclTerminateAppResult
Function TerminateTask( Wnd : HWND; Timeout : Integer) : TJclTerminateAppResult
Function TestBit( const Value : Byte; const Bit : TBitRange) : Boolean;
Function TestBit2( const Value : Shortint; const Bit : TBitRange) : Boolean;
Function TestBit3( const Value : Smallint; const Bit : TBitRange) : Boolean;
Function TestBit4( const Value : Word; const Bit : TBitRange) : Boolean;
Function TestBit5( const Value : Cardinal; const Bit : TBitRange) : Boolean;
Function TestBit6( const Value : Integer; const Bit : TBitRange) : Boolean;
Function TestBit7( const Value : Int64; const Bit : TBitRange) : Boolean;
Function TestBits( const Value, Mask : Byte) : Boolean;
Function TestBits1( const Value, Mask : Shortint) : Boolean;
Function TestBits2( const Value, Mask : Smallint) : Boolean;
Function TestBits3( const Value, Mask : Word) : Boolean;
Function TestBits4( const Value, Mask : Cardinal) : Boolean;
Function TestBits5( const Value, Mask : Integer) : Boolean;
Function TestBits6( const Value, Mask : Int64) : Boolean;
Function TestFDIVInstruction : Boolean
function TestStreamFormat(Stream: TStream): TStreamOriginalFormat
Function TextExtent( const Text : string) : TSize
function TextHeight(Text: string): Integer;
Function TextIsSame( const A1 : string; const A2 : string) : Boolean
Function TextStartsWith( const S, SubS : string) : Boolean
function TextToFloat(Buffer: PChar; var Value: Extended; ValueType: TFloatValue): Boolean)
Function ConvInteger(i : integer):string;
Function IntegerToText(i : integer):string;
Function TEXTTOSHORTCUT( TEXT : String) : TSHORTCUT
function TextWidth(Text: string): Integer;
Function ThreadCount : integer
function ThousandSeparator: char;
Function Ticks : Cardinal
Function Time : TDateTime
function Time: TDateTime;
function TimeGetTime: int64;
Function TimeOf( const AValue : TDateTime) : TDateTime
function TimeSeparator: char;
function TimeStampToDateTime(const TimeStamp: TTimeStamp): TDateTime
Function TimeStampToMSecs( TimeStamp : TTimeStamp) : Comp
function TimeStampToMSecs(const TimeStamp: TTimeStamp): Comp)
Function TimeToStr( DateTime : TDateTime) : string;
function TimeToStr(const DateTime: TDateTime): string;
Function TimeZoneBias : TDateTime
Function ToCommon( const AValue : Double) : Double
function ToCommon(const AValue: Double): Double;
Function Today : TDateTime
Function ToggleBit( const Value : Byte; const Bit : TBitRange) : Byte;
Function ToggleBit1( const Value : Shortint; const Bit : TBitRange) : Shortint;
Function ToggleBit2( const Value : Smallint; const Bit : TBitRange) : Smallint;
Function ToggleBit3( const Value : Word; const Bit : TBitRange) : Word;
Function ToggleBit4( const Value : Cardinal; const Bit : TBitRange) : Cardinal;
Function ToggleBit5( const Value : Integer; const Bit : TBitRange) : Integer;
Function ToggleBit6( const Value : Int64; const Bit : TBitRange) : Int64;
function TokenComponentIdent:String
Function TokenFloat : Extended
function TokenFloat:Extended
Function TokenInt : Longint
function TokenInt:LongInt
Function TokenString : string
function TokenString:String
Function TokenSymbolIs( const S : string) : Boolean
function TokenSymbolIs(S:String):Boolean
Function Tomorrow : TDateTime
Function ToRightOf( const pc : TControl; piSpace : Integer) : Integer
Function ToString : string
Function TotalVariance( const Data : array of Double) : Extended
Function Trace2( AURL : string) : string;
Function TrackMenu( Button : TToolButton) : Boolean
Function TRANSLATE( SRC, DEST : PCHAR; TOOEM : BOOLEAN) : INTEGER
Function TranslateURI( const URI : string) : string
Function TranslationMatchesLanguages( Exact : Boolean) : Boolean
Function TransparentStretchBlt( DstDC :HDC;DstX,DstY,DstW,DstH:Integer;SrcDC:HDC;SrcX,SrcY,SrcW,
SrcH:Integer;MaskDC : HDC; MaskX, MaskY : Integer) : Boolean
Function Trim( S : string) : string;
Function Trim( S : WideString) : WideString;
Function Trim(s : AnyString) : AnyString;
Function TrimAllOf( ATrim, AText : String) : String
Function TrimLeft( S : string) : string;
Function TrimLeft( S : WideString) : WideString;
function TrimLeft(const S: string): string)
Function TrimRight( S : string) : string;
Function TrimRight( S : WideString) : WideString;
function TrimRight(const S: string): string)
function TrueBoolStrs: array of string
Function Trunc(e : Extended) : Longint;
Function Trunc64(e: extended): Int64;
Function TruncPower( const Base, Exponent : Float) : Float
Function TryConvTypeToFamily( const AFrom, ATo : TConvType; out AFamily : TConvFamily) : Boolean;
Function TryConvTypeToFamily1( const AType : TConvType; out AFamily : TConvFamily) : Boolean;
function TryEncodeDate(Year, Month, Day: Word; var Date: TDateTime): Boolean;
Function TryEncodeDateDay( const AYear, ADayOfYear : Word; out AValue : TDateTime) : Boolean
Function TryEncodeDateMonthWeek(const AY,AMonth,AWeekOfMonth,ADayOfWeek:Word;var AValue:TDateTime): Boolean
Function TryEncodeDateTime(const AYear,AMonth,ADay,AHour,AMin,ASec,AMilliSecond:Word; out
AValue:TDateTime):Boolean
Function TryEncodeDateWeek(const AY,AWeekOfYear:Word;out AValue:TDateTime;const ADayOfWeek:Word): Boolean
Function TryEncodeDayOfWeekInMonth(const AYear,AMonth,ANthDayOfWeek,ADayOfWeek:Word;out
AVal:TDateTime):Bool
function TryEncodeTime(Hour, Min, Sec, MSec: Word; var Time: TDateTime): Boolean;
Function TryFloatToDateTime( Value : Extended; AResult : TDateTime) : Boolean
Function TryJulianDateToDateTime( const AValue : Double; out ADateTime : TDateTime) : Boolean
Function TryLock : Boolean
Function TryModifiedJulianDateToDateTime( const AValue : Double; out ADateTime : TDateTime) : Boolean
Function TryRecodeDateTime( const AValue : TDateTime; const AYear, AMonth, ADay, AHour, AMinute, ASecond,
AMilliSecond : Word; out AResult : TDateTime) : Boolean
Function TryStrToBcd( const AValue : string; var Bcd : TBcd) : Boolean
Function TryStrToConv( AText : string; out AValue : Double; out AType : TConvType) : Boolean
Function TryStrToDate( S : string; Value : TDateTime) : Boolean;
Function TryStrToDateTime( S : string; Value : TDateTime) : Boolean;
Function TryStrToTime( S : string; Value : TDateTime) : Boolean;
Function TryStrToInt(const S: AnsiString; var I: Integer): Boolean;
Function TryStrToInt64(const S: AnsiString; var I: Int64): Boolean;
Function TwoByteToWord( AByte1, AByte2 : Byte) : Word
Function TwoCharToWord( AChar1, AChar2 : Char) : Word
Function TwoToY( const Y : Float) : Float
Function UCS4StringToWideString( const S : UCS4String) : WideString
Function UIDL( const ADest : TStrings; const AMsgNum : Integer) : Boolean
function Unassigned: Variant;
Function UndoLastChange( FollowChange : Boolean) : Boolean
function UniCodeToStr(Value: string): string;
Function UnionRect( out Rect : TRect; const R1, R2 : TRect) : Boolean
function UnionRect(out Rect: TRect; const R1, R2: TRect): Boolean)
Function UnixDateTimeToDelphiDateTime( UnixDateTime : Cardinal) : TDateTime
Function UnixPathToDosPath( const Path : string) : string
Function UnixToDateTime( const AValue : Int64) : TDateTime
function UnixToDateTime(U: Int64): TDateTime;
Function UnlockRegion( libOffset : Longint; cb : Largeint; dwLockType : Longint) : HResult
Function UnlockResource( ResData : HGLOBAL) : LongBool
Function UnlockVolume( var Handle : THandle) : Boolean
Function UnMaskString( Mask, Value : String) : String
function UpCase(ch : Char ) : Char;
Function UpCaseFirst( const AStr : string) : string
Function UpCaseFirstWord( const AStr : string) : string
Function UpdateAction( Action : TBasicAction) : Boolean
Function UpdateKind : TUpdateKind
Function UPDATESTATUS : TUPDATESTATUS
Function UpperCase( S : string) : string
Function Uppercase(s : AnyString) : AnyString;
Function URLDecode( ASrc : string) : string
Function URLEncode( const ASrc : string) : string
Function UseRightToLeftAlignment : Boolean
Function UseRightToLeftAlignmentForField( const AField : TField; Alignment : TAlignment) : Boolean
Function UseRightToLeftReading : Boolean
Function UTF8CharLength( Lead : Char) : Integer
Function UTF8CharSize( Lead : Char) : Integer
Function UTF8Decode( const S : UTF8String) : WideString
Function UTF8Encode( const WS : WideString) : UTF8String
Function UTF8LowerCase( const S : UTF8string) : UTF8string
Function Utf8ToAnsi( const S : UTF8String) : string
Function Utf8ToAnsiEx( const S : UTF8String; const cp : integer) : string
Function UTF8UpperCase( const S : UTF8string) : UTF8string
Function ValidFieldIndex( FieldIndex : Integer) : Boolean
Function ValidParentForm(control: TControl): TForm
Function Value : Variant
Function ValueExists( const Section, Ident : string) : Boolean
Function ValueOf( const Key : string) : Integer
Function ValueInSet(AValue: Variant; ASet: Variant): Boolean;
Function VALUEOFKEY( const AKEY : VARIANT) : VARIANT
Function VarArrayFromStrings( Strings : TStrings) : Variant
Function VarArrayFromWideStrings( Strings : TWideStrings) : Variant
Function VarArrayGet(var S : Variant; I : Integer) : Variant;
Function VarFMTBcd : TVarType
Function VarFMTBcdCreate1 : Variant;
Function VarFMTBcdCreate2( const AValue : string; Precision, Scale : Word) : Variant;
Function VarFMTBcdCreate3( const AValue : Double; Precision : Word; Scale : Word) : Variant;
Function VarFMTBcdCreate4( const ABcd : TBcd) : Variant;
Function Variance( const Data : array of Double) : Extended
Function VariantAdd2( const V1 : Variant; const V2 : Variant) : Variant
Function VariantAnd2( const V1 : Variant; const V2 : Variant) : Variant
Function VariantDiv2( const V1 : Variant; const V2 : Variant) : Variant
Function VariantGetElement( const V : Variant; i1 : integer) : Variant;
Function VariantGetElement1( const V : Variant; i1, i2 : integer) : Variant;
Function VariantGetElement2( const V : Variant; i1, i2, i3 : integer) : Variant;
Function VariantGetElement3( const V : Variant; i1, i2, i3, i4 : integer) : Variant;
Function VariantGetElement4( const V : Variant; i1, i2, i3, i4, i5 : integer) : Variant;
Function VariantMod2( const V1 : Variant; const V2 : Variant) : Variant
Function VariantMul2( const V1 : Variant; const V2 : Variant) : Variant
Function VariantNeg( const V1 : Variant) : Variant
Function VariantNot( const V1 : Variant) : Variant
Function VariantOr2( const V1 : Variant; const V2 : Variant) : Variant
Function VariantShl2( const V1 : Variant; const V2 : Variant) : Variant
Function VariantShr2( const V1 : Variant; const V2 : Variant) : Variant
Function VariantSub2( const V1 : Variant; const V2 : Variant) : Variant
Function VariantXor2( const V1 : Variant; const V2 : Variant) : Variant
function VarIsEmpty(const V: Variant): Boolean;
Function VarIsFMTBcd( const AValue : Variant) : Boolean;
function VarIsNull(const V: Variant): Boolean;
Function VarToBcd( const AValue : Variant) : TBcd
function VarType(const V: Variant): TVarType;
Function VarType( const V : Variant) : TVarType
Function VarAsType( const V : Variant; AVarType : TVarType) : Variant
Function VarIsType( const V : Variant; AVarType : TVarType) : Boolean;
Function VarIsType1( const V : Variant; const AVarTypes : array of TVarType) : Boolean;
Function VarIsByRef( const V : Variant) : Boolean
Function VarIsEmpty( const V : Variant) : Boolean
Procedure VarCheckEmpty( const V : Variant)
Function VarIsNull( const V : Variant) : Boolean
Function VarIsClear( const V : Variant) : Boolean
Function VarIsCustom( const V : Variant) : Boolean
Function VarIsOrdinal( const V : Variant) : Boolean
Function VarIsFloat( const V : Variant) : Boolean
Function VarIsNumeric( const V : Variant) : Boolean
Function VarIsStr( const V : Variant) : Boolean
Function VarToStr( const V : Variant) : string
Function VarToStrDef( const V : Variant; const ADefault : string) : string
Function VarToWideStr( const V : Variant) : WideString
Function VarToWideStrDef( const V : Variant; const ADefault : WideString) : WideString
Function VarToDateTime( const V : Variant) : TDateTime
Function VarFromDateTime( const DateTime : TDateTime) : Variant
Function VarInRange( const AValue, AMin, AMax : Variant) : Boolean
Function VarEnsureRange( const AValue, AMin, AMax : Variant) : Variant
TVariantRelationship', '( vrEqual, vrLessThan, vrGreaterThan, vrNotEqual )
Function VarSameValue( const A, B : Variant) : Boolean
Function VarCompareValue( const A, B : Variant) : TVariantRelationship
Function VarIsEmptyParam( const V : Variant) : Boolean
Function VarIsError( const V : Variant; out AResult : HRESULT) : Boolean;
Function VarIsError1( const V : Variant) : Boolean;
Function VarAsError( AResult : HRESULT) : Variant
Procedure VarCopyNoInd( var Dest : Variant; const Source : Variant)
Function VarIsArray( const A : Variant) : Boolean;
Function VarIsArray1( const A : Variant; AResolveByRef : Boolean) : Boolean;
Function VarArrayCreate( const Bounds : array of Integer; AVarType : TVarType) : Variant
Function VarArrayOf( const Values : array of Variant) : Variant
Function VarArrayRef( const A : Variant) : Variant
Function VarTypeIsValidArrayType( const AVarType : TVarType) : Boolean
Function VarTypeIsValidElementType( const AVarType : TVarType) : Boolean
Function VarArrayDimCount( const A : Variant) : Integer
Function VarArrayLowBound( const A : Variant; Dim : Integer) : Integer
Function VarArrayHighBound( const A : Variant; Dim : Integer) : Integer
Function VarArrayLock( const A : Variant) : ___Pointer
Procedure VarArrayUnlock( const A : Variant)
Function VarArrayGet( const A : Variant; const Indices : array of Integer) : Variant
Procedure VarArrayPut( var A : Variant; const Value : Variant; const Indices : array of Integer)
Procedure DynArrayToVariant( var V : Variant; const DynArray : ___Pointer; TypeInfo : ___Pointer)
Procedure DynArrayFromVariant( var DynArray : ___Pointer; const V : Variant; TypeInfo : ___Pointer)
Function Unassigned : Variant
Function Null : Variant
Function VectorAdd( const V1, V2 : TFloatPoint) : TFloatPoint
function VectorAdd(const V1,V2: TFloatPoint): TFloatPoint;
Function VectorDot( const V1, V2 : TFloatPoint) : Double
function VectorDot(const V1,V2: TFloatPoint): Double;
Function VectorLengthSqr( const V : TFloatPoint) : Double
function VectorLengthSqr(const V: TFloatPoint): Double;
Function VectorMult( const V : TFloatPoint; const s : Double) : TFloatPoint
function VectorMult(const V: TFloatPoint; const s: Double): TFloatPoint;
Function VectorSubtract( const V1, V2 : TFloatPoint) : TFloatPoint
function VectorSubtract(const V1,V2: TFloatPoint): TFloatPoint;
Function Verify( AUserName : String) : String
Function Versine( X : Float) : Float
function VersionCheck: boolean;
Function VersionLanguageId( const LangIdRec : TLangIdRec) : string
Function VersionLanguageName( const LangId : Word) : string
Function VersionResourceAvailable( const FileName : string) : Boolean
Function Visible : Boolean
function VolumeID(DriveChar: Char): string
Function WaitFor( const AString : string) : string
Function WaitFor( const TimeOut : Cardinal) : TJclWaitResult
Function WaitFor1 : TWaitResult;
Function WaitForData( Timeout : Longint) : Boolean
Function WebColorNameToColor( WebColorName : string) : TColor
Function WebColorStrToColor( WebColor : string) : TColor
Function WebColorToRGB( WebColor : Integer) : Integer
Function wGet(aURL, afile: string): boolean;'
Function wGet2(aURL, afile: string): boolean;'  //without file open
Function WebGet(aURL, afile: string): boolean;'
Function WebExists: boolean;   //alias to isinternet
Function WeekOf( const AValue : TDateTime) : Word
Function WeekOfTheMonth( const AValue : TDateTime) : Word;
Function WeekOfTheMonth1( const AValue : TDateTime; var AYear, AMonth : Word) : Word;
Function WeekOfTheYear( const AValue : TDateTime) : Word;
Function WeekOfTheYear1( const AValue : TDateTime; var AYear : Word) : Word;
Function WeeksBetween( const ANow, AThen : TDateTime) : Integer
Function WeeksInAYear( const AYear : Word) : Word
Function WeeksInYear( const AValue : TDateTime) : Word
Function WeekSpan( const ANow, AThen : TDateTime) : Double
Function WideAdjustLineBreaks( const S : WideString; Style : TTextLineBreakStyle) : WideString
Function WideCat( const x, y : WideString) : WideString
Function WideCompareStr( S1, S2 : WideString) : Integer
function WideCompareStr(const S1: WideString; const S2: WideString): Integer)
Function WideCompareText( S1, S2 : WideString) : Integer
function WideCompareText(const S1: WideString; const S2: WideString): Integer)
Function WideCopy( const src : WideString; index, count : Integer) : WideString
Function WideDequotedStr( const S : WideString; AQuote : WideChar) : WideString
Function WideEqual( const x, y : WideString) : Boolean
function WideFormat(const Format: WideString; const Args: array of const): WideString)
Function WideGreater( const x, y : WideString) : Boolean
Function WideLength( const src : WideString) : Integer
Function WideLess( const x, y : WideString) : Boolean
Function WideLowerCase( S : WideString) : WideString
function WideLowerCase(const S: WideString): WideString)
Function WidePos( const src, sub : WideString) : Integer
Function WideQuotedStr( const S : WideString; Quote : WideChar) : WideString
Function WideReplaceStr( const AText, AFromText, AToText : WideString) : WideString
Function WideReplaceText( const AText, AFromText, AToText : WideString) : WideString
Function WideSameStr( S1, S2 : WideString) : Boolean
function WideSameStr(const S1: WideString; const S2: WideString): Boolean)
Function WideSameText( S1, S2 : WideString) : Boolean
function WideSameText(const S1: WideString; const S2: WideString): Boolean)
Function WideStringReplace(const S,OldPattern, NewPattern: Widestring; Flags: TReplaceFlags): Widestring
Function WideStringToUCS4String( const S : WideString) : UCS4String
Function WideUpperCase( S : WideString) : WideString
Function Win32BackupFile( const FileName : string; Move : Boolean) : Boolean
function Win32Check(RetVal: boolean): boolean)
Function Win32DeleteFile( const FileName : string; MoveToRecycleBin : Boolean) : Boolean
Function Win32RestoreFile( const FileName : string) : Boolean
Function Win32Type : TIdWin32Type
Function WinColor( const Color32 : TColor32) : TColor
function winexec(FileName: pchar; showCmd: integer): integer;
Function WinExec32( const Cmd : string; const CmdShow : Integer) : Boolean
Function WinExec32AndWait( const Cmd : string; const CmdShow : Integer) : Cardinal
Function WithinPastDays( const ANow, AThen : TDateTime; const ADays : Integer) : Boolean
Function WithinPastHours( const ANow, AThen : TDateTime; const AHours : Int64) : Boolean
Function WithinPastMilliSeconds( const ANow, AThen : TDateTime; const AMilliSeconds : Int64) : Boolean
Function WithinPastMinutes( const ANow, AThen : TDateTime; const AMinutes : Int64) : Boolean
Function WithinPastMonths( const ANow, AThen : TDateTime; const AMonths : Integer) : Boolean
Function WithinPastSeconds( const ANow, AThen : TDateTime; const ASeconds : Int64) : Boolean
Function WithinPastWeeks( const ANow, AThen : TDateTime; const AWeeks : Integer) : Boolean
Function WithinPastYears( const ANow, AThen : TDateTime; const AYears : Integer) : Boolean
Function WNetAddConnection( lpRemoteName, lpPassword, lpLocalName : PChar) : DWORD
Function WordToStr( const Value : Word) : String
Function WordGridFormatIdentToInt( const Ident : string; var Value : Longint) : Boolean
Function IntToWordGridFormatIdent( Value : Longint; var Ident : string) : Boolean
Procedure GetWordGridFormatValues( Proc : TGetStrProc)
Function WorkArea : Integer
Function WrapText( Line : string; MaxCol : Integer) : string;
Function WrapText( Line, BreakStr : string; BreakChars : TSysCharSet; MaxCol : Integer) : string;
Function Write( pv : Pointer; cb : Longint; pcbWritten : PLongint) : HResult
function Write(Buffer:String;Count:LongInt):LongInt
Function WriteClient( var Buffer, Count : Integer) : Integer
Function WriteFile( const AFile : string; const AEnableTransferFile : Boolean) : Cardinal
Function WriteHeaders( StatusCode : Integer; const ReasonString, Headers : string) : Boolean
Function WriteString( const AString : string) : Boolean
Function WStrGet(var S : AnyString; I : Integer) : WideChar;
Function wvsprintf( Output : PChar; Format : PChar; arglist : va_list) : Integer
Function wsprintf( Output : PChar; Format : PChar) : Integer
Function XmlDateTimeToStr( const XmlDateTime : string; const Format : string) : string
Function XmlTimeToStr( const XmlTime : string; const Format : string) : string
Function XorDecode( const Key, Source : string) : string
Function XorEncode( const Key, Source : string) : string
Function XorString( const Key, Src : ShortString) : ShortString
Function Yield : Bool
Function YearOf( const AValue : TDateTime) : Word
Function YearsBetween( const ANow, AThen : TDateTime) : Integer
Function YearSpan( const ANow, AThen : TDateTime) : Double
Function Yesterday : TDateTime
Function YesNoDialog(const ACaption, AMsg: string): boolean;
Function( const Name : string; Proc : TUserFunction)
Function using Special_Scholz from 3.8.5.0
Function TimeToFloat(value:Extended):Extended; // Normalstunden --> Industriestunden
Function FloatToTime(value:Extended):Extended; // Industriestunden --> Normalstunden
Function FloatToTime2Dec(value:Extended):Extended;
Function MinToStd(value:Extended):Extended;
Function MinToStdAsString(value:Extended):String;
Function RoundFloatToStr(zahl:Extended; decimals:integer):String;
Function RoundFloat(zahl:Extended; decimals:integer):Extended;
Function Round2Dec (zahl:Extended):Extended;
Function GetAngle(x,y:Extended):Double;
Function AddAngle(a1,a2:Double):Double;

*********************************************************************
unit uPSI_StText;
*********************************************************************
Function TextSeek( var F : TextFile; Target : LongInt) : Boolean
Function TextFileSize( var F : TextFile) : LongInt
Function TextPos( var F : TextFile) : LongInt
Function TextFlush( var F : TextFile) : Boolean

*********************************************************************
from  JvVCLUtils;
*********************************************************************
{ Windows resources (bitmaps and icons) VCL-oriented routines }
procedure DrawBitmapTransparent(Dest:TCanvas;DstX,DstY:Integer;Bitmap:TBitmap;TransparentColor:TColor);
procedure DrawBitmapRectTransparent(Dest: TCanvas;DstX,
DstY:Int;SrcRect:TRect;Bitmap:TBitmap;TransparColor:TColor);
procedure StretchBitmapRectTransparent(Dest: TCanvas; DstX, DstY, DstW,DstH: Integer; SrcRect: TRect;
Bitmap: TBitmap; TransparentColor: TColor);
function MakeBitmap(ResID: PChar): TBitmap;
function MakeBitmapID(ResID: Word): TBitmap;
function MakeModuleBitmap(Module: THandle; ResID: PChar): TBitmap;
function CreateTwoColorsBrushPattern(Color1, Color2: TColor): TBitmap;
function CreateDisabledBitmap_NewStyle(FOriginal: TBitmap; BackColor: TColor): TBitmap;
function CreateDisabledBitmapEx(FOriginal: TBitmap; OutlineColor, BackColor,
  HighlightColor, ShadowColor: TColor; DrawHighlight: Boolean): TBitmap;
function CreateDisabledBitmap(FOriginal: TBitmap; OutlineColor: TColor): TBitmap;
function ChangeBitmapColor(Bitmap: TBitmap; Color, NewColor: TColor): TBitmap;
procedure AssignBitmapCell(Source: TGraphic; Dest: TBitmap; Cols, Rows,Index: Integer);
{$IFDEF WIN32}
procedure ImageListDrawDisabled(Images: TImageList; Canvas: TCanvas;
  X, Y, Index: Integer; HighlightColor, GrayColor: TColor; DrawHighlight: Boolean);
{$ENDIF}
function MakeIcon(ResID: PChar): TIcon;
function MakeIconID(ResID: Word): TIcon;
function MakeModuleIcon(Module: THandle; ResID: PChar): TIcon;
function CreateBitmapFromIcon(Icon: TIcon; BackColor: TColor): TBitmap;
{$IFDEF WIN32}
function CreateIconFromBitmap(Bitmap: TBitmap; TransparentColor: TColor): TIcon;
{$ENDIF}
{ Service routines }
procedure NotImplemented;
procedure ResourceNotFound(ResID: PChar);
function PointInRect(const P: TPoint; const R: TRect): Boolean;
function PointInPolyRgn(const P: TPoint; const Points: array of TPoint): Boolean;
function PaletteColor(Color: TColor): Longint;
function WidthOf(R: TRect): Integer;
function HeightOf(R: TRect): Integer;
procedure PaintInverseRect(const RectOrg, RectEnd: TPoint);
procedure DrawInvertFrame(ScreenRect: TRect; Width: Integer);
procedure CopyParentImage(Control: TControl; Dest: TCanvas);
procedure Delay(MSecs: Longint);
procedure CenterControl(Control: TControl);
Function PaletteEntries( Palette : HPALETTE) : Integer
Function WindowClassName( Wnd : HWND) : string
Function ScreenWorkArea : TRect
Procedure MoveWindowOrg( DC : HDC; DX, DY : Integer)
Procedure SwitchToWindow( Wnd : HWND; Restore : Boolean)
Procedure ActivateWindow( Wnd : HWND)
Procedure ShowWinNoAnimate( Handle : HWND; CmdShow : Integer)
Procedure CenterWindow( Wnd : HWND)
Procedure ShadeRect( DC : HDC; const Rect : TRect)
Procedure KillMessage( Wnd : HWND; Msg : Cardinal)
Function DialogsToPixelsX( Dlgs : Word) : Word
Function DialogsToPixelsY( Dlgs : Word) : Word
Function PixelsToDialogsX( Pixs : Word) : Word
Function PixelsToDialogsY( Pixs : Word) : Word
{$IFDEF WIN32}
procedure ShowMDIClientEdge(ClientHandle: THandle; ShowEdge: Boolean);
function MakeVariant(const Values: array of Variant): Variant;
{$ENDIF}
function CreateRotatedFont(Font: TFont; Angle: Integer): HFONT;
function MsgBox(const Caption, Text: string; Flags: Integer): Integer;
function MsgDlg(const Msg:string; AType:TMsgDlgType; AButtons: TMsgDlgButtons; HelpCtx: Longint): Word;
{$IFDEF CBUILDER}
function FindPrevInstance(const MainFormClass: ShortString; const ATitle: string): HWND;
function ActivatePrevInstance(const MainFormClass: ShortString; const ATitle: string): Boolean;
{$ELSE}
function FindPrevInstance(const MainFormClass, ATitle: string): HWND;
function ActivatePrevInstance(const MainFormClass, ATitle: string): Boolean;
{$ENDIF CBUILDER}
function IsForegroundTask: Boolean;
procedure MergeForm(AControl: TWinControl; AForm: TForm; Align: TAlign; Show: Boolean);
function GetAveCharSize(Canvas: TCanvas): TPoint;
function MinimizeText(const Text: string; Canvas: TCanvas; MaxWidth: Integer): string;
procedure FreeUnusedOle;
procedure Beep;
function GetWindowsVersionJ: string;
function LoadDLL(const LibName: string): THandle;
function RegisterServer(const ModuleName: string): Boolean;
{$IFNDEF WIN32}
function IsLibrary: Boolean;
{$ENDIF}
{ Gradient filling routine }
type TFillDirection = (fdTopToBottom, fdBottomToTop, fdLeftToRight, fdRightToLeft);
procedure GradientFillRect(Canvas: TCanvas; ARect: TRect; StartColor, EndColor: TColor; Direction: TFillDirection; Colors: Byte);
{ String routines }
function GetEnvVar(const VarName: string): string;
function AnsiUpperFirstChar(const S: string): string;
function StringToPChar(var S: string): PChar;
function StrPAlloc(const S: string): PChar;
procedure SplitCommandLine(const CmdLine: string; var ExeName,Params: string);
function DropT(const S: string): string;
{ Memory routines }
function AllocMemo(Size: Longint): Pointer;
function ReallocMemo(fpBlock: Pointer; Size: Longint): Pointer;
procedure FreeMemo(var fpBlock: Pointer);
function GetMemoSize(fpBlock: Pointer): Longint;
function CompareMem(fpBlock1, fpBlock2: Pointer; Size: Cardinal): Boolean;
{$IFNDEF COMPILER5_UP}
procedure FreeAndNil(var Obj);
{$ENDIF}
// from PNGLoader
Function OptimizeForPNG(Image:TLinearBitmap;QuantizationSteps:Integer;TransparentColor:TColor):Integer
Procedure TransformRGB2LOCO( Image : TLinearBitmap)
Procedure TransformLOCO2RGB( Image : TLinearBitmap)
Procedure SortPalette( const Pal : TPalette; var ColorMap : TColorMap)
Function DrawButtonFace( Canvas : TCanvas; const Client : TRect; BevelWidth : Integer; Style : TButtonStyle; IsRounded, IsDown, IsFocused : Boolean) : TRect  //TButtons
  Function IsAnAllResult( const AModalResult : TModalResult) : Boolean
  Function InitWndProc( HWindow : HWnd; Message, WParam : Longint; LParam : Longint) : Longint
 AddConstantN('CTL3D_ALL','LongWord').SetUInt( $FFFF);
 //Procedure ChangeBiDiModeAlignment( var Alignment : TAlignment)
 //Function SendAppMessage( Msg : Cardinal; WParam, LParam : Longint) : Longint
 //Procedure MoveWindowOrg( DC : HDC; DX, DY : Integer)
 Procedure SetImeMode( hWnd : HWND; Mode : TImeMode)
 Procedure SetImeName( Name : TImeName)
 Function Win32NLSEnableIME( hWnd : HWND; Enable : Boolean) : Boolean
 Function Imm32GetContext( hWnd : HWND) : HIMC
 Function Imm32ReleaseContext( hWnd : HWND; hImc : HIMC) : Boolean
 Function Imm32GetConversionStatus( hImc : HIMC; var Conversion, Sentence : longword) : Boolean
 Function Imm32SetConversionStatus( hImc : HIMC; Conversion, Sentence : longword) : Boolean
 Function Imm32SetOpenStatus( hImc : HIMC; fOpen : Boolean) : Boolean
// Function Imm32SetCompositionWindow( hImc : HIMC; lpCompForm : PCOMPOSITIONFORM) : Boolean
 //Function Imm32SetCompositionFont( hImc : HIMC; lpLogfont : PLOGFONTA) : Boolean
 Function Imm32GetCompositionString(hImc:HIMC;dWord1:longword;lpBuf:string;dwBufLen:longint):Longint
 Function Imm32IsIME( hKl : longword) : Boolean
 Function Imm32NotifyIME( hImc : HIMC; dwAction, dwIndex, dwValue:longword):Boolean
 Procedure DragDone( Drop : Boolean)


//****************************************added  from jvjvclutils
function CanvasMaxTextHeight(Canvas: TCanvas): Integer;
function ReplaceComponentReference(This,NewReference:TComponent;var VarReference:TComponent):Boolean;
procedure DrawLine(Canvas: TCanvas; X, Y, X2, Y2: Integer);
function IsPositiveResult(Value: TModalResult): Boolean;
function IsNegativeResult(Value: TModalResult): Boolean;
function IsAbortResult(const Value: TModalResult): Boolean;
function StripAllFromResult(const Value: TModalResult): TModalResult;
// returns either BrightColor or DarkColor depending on the luminance of AColor
// This function gives the same result (AFAIK) as the function used in Windows to
// calculate the desktop icon text color based on the desktop background color
function SelectColorByLuminance(AColor, DarkColor, BrightColor: TColor): TColor;
type TJvHTMLCalcType = (htmlShow, htmlCalcWidth, htmlCalcHeight, htmlHyperLink);

procedure HTMLDrawTextEx(Canvas: TCanvas; Rect: TRect;
  const State: TOwnerDrawState; const Text: string; var Width: Integer;
  CalcType: TJvHTMLCalcType;  MouseX, MouseY: Integer; var MouseOnLink: Boolean;
  var LinkName: string; Scale: Integer = 100); overload;
procedure HTMLDrawTextEx(Canvas: TCanvas; Rect: TRect;
  const State: TOwnerDrawState; const Text: string; var Width, Height: Integer;
  CalcType: TJvHTMLCalcType;  MouseX, MouseY: Integer; var MouseOnLink: Boolean;
  var LinkName: string; Scale: Integer = 100); overload;
function HTMLDrawText(Canvas: TCanvas; Rect: TRect;
  const State: TOwnerDrawState; const Text: string; Scale: Integer = 100): string;
function HTMLDrawTextHL(Canvas: TCanvas; Rect: TRect;
  const State: TOwnerDrawState; const Text: string; MouseX, MouseY: Integer;
  Scale: Integer = 100): string;
function HTMLPlainText(const Text: string): string;
function HTMLTextExtent(Canvas: TCanvas; Rect: TRect;
  const State: TOwnerDrawState; const Text: string; Scale: Integer = 100): TSize;
function HTMLTextWidth(Canvas: TCanvas; Rect: TRect;
  const State: TOwnerDrawState; const Text: string; Scale: Integer = 100): Integer;
function HTMLTextHeight(Canvas: TCanvas; const Text: string; Scale: Integer = 100): Integer;
function HTMLPrepareText(const Text: string): string;

****************************************** uPSI_JvAppUtils;
Function GetDefaultSection( Component : TComponent) : string
Procedure GetDefaultIniData(Control:TControl; var IniFileName, Section : string; UseRegistry : Boolean)
Procedure GetDefaultIniData( Control : TControl; var IniFileName, Section : string)
Function GetDefaultIniName : string
 //'OnGetDefaultIniName','TOnGetDefaultIniName);
Function GetDefaultIniRegKey : string
Function FindForm( FormClass : TFormClass) : TForm
Function FindShowForm( FormClass : TFormClass; const Caption : string) : TForm
Function ShowDialog( FormClass : TFormClass) : Boolean
 //Function InstantiateForm( FormClass : TFormClass; var Reference) : TForm
Procedure SaveFormPlacement( Form : TForm; const IniFileName : string; UseRegistry : Boolean)
Procedure RestoreFormPlacement( Form : TForm; const IniFileName : string; UseRegistry : Boolean)
Procedure SaveMDIChildrenReg( MainForm : TForm; IniFile : TRegIniFile)
Procedure SaveFormPlacement( Form : TForm; const IniFileName : string)
Procedure RestoreFormPlacement( Form : TForm; const IniFileName : string)
Function GetUniqueFileNameInDir( const Path, FileNameMask : string) : string
Function StrToIniStr( const Str : string) : string
Function IniStrToStr( const Str : string) : string
Function IniReadString( IniFile : TObject; const Section, Ident, Default : string) : string
Procedure IniWriteString( IniFile : TObject; const Section, Ident, Value : string)
Function IniReadInteger( IniFile : TObject; const Section, Ident : string; Default : Longint) : Longint
Procedure IniWriteInteger( IniFile : TObject; const Section, Ident : string; Value : Longint)
Function IniReadBool( IniFile : TObject; const Section, Ident : string; Default : Boolean) : Boolean
Procedure IniWriteBool( IniFile : TObject; const Section, Ident : string; Value : Boolean)
Procedure IniReadSections( IniFile : TObject; Strings : TStrings)
Procedure IniEraseSection( IniFile : TObject; const Section : string)
Procedure IniDeleteKey( IniFile : TObject; const Section, Ident : string)
Procedure AppBroadcast( Msg, wParam : Longint; lParam : Longint)
Procedure AppBroadcast( Msg, wParam : Word; lParam : Longint)
Procedure AppTaskbarIcons( AppOnly : Boolean)
Procedure InternalSaveGridLayout( Grid : TCustomGrid; IniFile : TObject; const Section : string)
Procedure InternalRestoreGridLayout( Grid : TCustomGrid; IniFile : TObject; const Section : string)
Procedure InternalSaveMDIChildren( MainForm : TForm; IniFile : TObject)
Procedure InternalRestoreMDIChildren( MainForm : TForm; IniFile : TObject)
****************************************** uPSI_JvDBUtils;
Function CreateLocate( DataSet : TDataSet) : TJvLocateObject
Function IsDataSetEmpty( DataSet : TDataSet) : Boolean
Procedure RefreshQuery( Query : TDataSet)
Function DataSetSortedSearch(DataSet:TDataSet;const Value,FieldName:string;CaseInsensitive:Bool):Boolean
Function DataSetSectionName( DataSet : TDataSet) : string
Procedure InternalSaveFields( DataSet : TDataSet; IniFile : TObject; const Section : string)
Procedure InternalRestoreFields(DataSet:TDataSet;IniFile:TObject;const Section:string;RestoreVisible:Bool)
Function DataSetLocateThrough(DataSet:TDataSet; const KeyFields: string; const KeyValues: Variant;
Options: TLocateOptions) : Boolean
Procedure SaveFields( DataSet : TDataSet; IniFile : TIniFile)
Procedure RestoreFields( DataSet : TDataSet; IniFile : TIniFile; RestoreVisible : Boolean)
Procedure AssignRecord( Source, Dest : TDataSet; ByName : Boolean)
Function ConfirmDelete : Boolean
Procedure ConfirmDataSetCancel( DataSet : TDataSet)
Procedure CheckRequiredField( Field : TField)
Procedure CheckRequiredFields( const Fields : array of TField)
Function DateToSQL( Value : TDateTime) : string
Function FormatSQLDateRange( Date1, Date2 : TDateTime; const FieldName : string) : string
Function FormatSQLDateRangeEx( Date1, Date2 : TDateTime; const FieldName : string) : string
Function FormatSQLNumericRange(const FieldName:string;LowVal,HighVal,LowEmpty,
HighEmpty:Double;Inclusive:Bool):string
Function StrMaskSQL( const Value : string) : string
Function FormatSQLCondition(const FieldName,Operator,Val:string;FieldType:TFieldType;Exact:Bool):string
Function FormatAnsiSQLCondition(const FieldName,Operator,Val:string;FieldType:TFieldType;Exact:Bool):string
Procedure _DBError( const Msg : string)
 Const('TrueExpr','String '0=0
 Const('sdfStandard16','String '''"''mm''/''dd''/''yyyy''"''
 Const('sdfStandard32','String ''''''dd/mm/yyyy''''''
 Const('sdfOracle','String '"TO_DATE(''"dd/mm/yyyy"'', ''DD/MM/YYYY'')"
 Const('sdfInterbase','String '"CAST(''"mm"/"dd"/"yyyy"'' AS DATE)"
 Const('sdfMSSQL','String '"CONVERT(datetime, ''"mm"/"dd"/"yyyy"'', 103)"
 AddTypeS('Largeint', 'Longint
 addTypeS('TIFException', '(ErNoError, erCannotImport, erInvalidType, ErInternalError, '+
   'erInvalidHeader, erInvalidOpcode, erInvalidOpcodeParameter, erNoMainProc, erOutOfGlobalVarsRange, '+
   'erOutOfProcRange, ErOutOfRange, erOutOfStackRange, ErTypeMismatch, erUnexpectedEof, '+
   'erVersionError, ErDivideByZero, ErMathError,erCouldNotCallProc, erOutofRecordRange, '+
   'erOutOfMemory,erException,erNullPointerException,erNullVariantErrorerInterfaceNotSupportederError);
(*----------------------------------------------------------------------------*)
procedure SIRegister_JclIniFiles(CL: TPSPascalCompiler);
begin
 Function JIniReadBool( const FileName, Section, Line : string) : Boolean
 Function JIniReadInteger( const FileName, Section, Line : string) : Integer
 Function JIniReadString( const FileName, Section, Line : string) : string
 Procedure JIniWriteBool( const FileName, Section, Line : string; Value : Boolean)
 Procedure JIniWriteInteger( const FileName, Section, Line : string; Value : Integer)
 Procedure JIniWriteString( const FileName, Section, Line, Value : string)
 Procedure JIniReadStrings( IniFile : TCustomIniFile; const Section : string; Strings : TStrings)
 Procedure JIniWriteStrings( IniFile : TCustomIniFile; const Section : string; Strings : TStrings)
end;

(* === compile-time registration functions === *)
(*----------------------------------------------------------------------------*)
procedure SIRegister_JclDateTime(CL: TPSPascalCompiler);
begin
 'UnixTimeStart','LongInt'( 25569);
 Function JEncodeDate( const Year : Integer; Month, Day : Word) : TDateTime
 Procedure JDecodeDate( Date : TDateTime; var Year, Month, Day : Word);
 Procedure DecodeDate1( Date : TDateTime; var Year : Integer; var Month, Day : Word);
 Procedure DecodeDate2( Date : TDateTime; var Year, Month, Day : Integer);
 Function CenturyOfDate( const DateTime : TDateTime) : Integer
 Function CenturyBaseYear( const DateTime : TDateTime) : Integer
 Function DayOfDate( const DateTime : TDateTime) : Integer
 Function MonthOfDate( const DateTime : TDateTime) : Integer
 Function YearOfDate( const DateTime : TDateTime) : Integer
 Function JDayOfTheYear( const DateTime : TDateTime; var Year : Integer) : Integer;
 Function DayOfTheYear1( const DateTime : TDateTime) : Integer;
 Function DayOfTheYearToDateTime( const Year, Day : Integer) : TDateTime
 Function HourOfTime( const DateTime : TDateTime) : Integer
 Function MinuteOfTime( const DateTime : TDateTime) : Integer
 Function SecondOfTime( const DateTime : TDateTime) : Integer
 Function GetISOYearNumberOfDays( const Year : Word) : Word
 Function IsISOLongYear( const Year : Word) : Boolean;
 Function IsISOLongYear1( const DateTime : TDateTime) : Boolean;
 Function ISODayOfWeek( const DateTime : TDateTime) : Word
 Function JISOWeekNumber( DateTime : TDateTime; var YearOfWeekNumber, WeekDay : Integer) : Integer;
 Function ISOWeekNumber1( DateTime : TDateTime; var YearOfWeekNumber : Integer) : Integer;
 Function ISOWeekNumber2( DateTime : TDateTime) : Integer;
 Function ISOWeekToDateTime( const Year, Week, Day : Integer) : TDateTime
 Function JIsLeapYear( const Year : Integer) : Boolean;
 Function IsLeapYear1( const DateTime : TDateTime) : Boolean;
 Function JDaysInMonth( const DateTime : TDateTime) : Integer
 Function Make4DigitYear( Year, Pivot : Integer) : Integer
 Function JMakeYear4Digit( Year, WindowsillYear : Integer) : Integer
 Function JEasterSunday( const Year : Integer) : TDateTime // TDosDateTime', 'Integer
 Function JFormatDateTime( Form : string; DateTime : TDateTime) : string
 Function FATDatesEqual( const FileTime1, FileTime2 : Int64) : Boolean;
 Function FATDatesEqual1( const FileTime1, FileTime2 : TFileTime) : Boolean;
 Function HoursToMSecs( Hours : Integer) : Integer
 Function MinutesToMSecs( Minutes : Integer) : Integer
 Function SecondsToMSecs( Seconds : Integer) : Integer
 Function TimeOfDateTimeToSeconds( DateTime : TDateTime) : Integer
 Function TimeOfDateTimeToMSecs( DateTime : TDateTime) : Integer
 Function DateTimeToLocalDateTime( DateTime : TDateTime) : TDateTime
 Function LocalDateTimeToDateTime( DateTime : TDateTime) : TDateTime
 Function DateTimeToDosDateTime( const DateTime : TDateTime) : TDosDateTime
 Function JDateTimeToFileTime( DateTime : TDateTime) : TFileTime
 Function JDateTimeToSystemTime( DateTime : TDateTime) : TSystemTime;
 Procedure DateTimeToSystemTime1( DateTime : TDateTime; var SysTime : TSystemTime);
 Function LocalDateTimeToFileTime( DateTime : TDateTime) : FileTime
 Function DosDateTimeToDateTime( const DosTime : TDosDateTime) : TDateTime
 Function JDosDateTimeToFileTime( DosTime : TDosDateTime) : TFileTime;
 Procedure DosDateTimeToFileTime1( DTH, DTL : Word; FT : TFileTime);
 Function DosDateTimeToSystemTime( const DosTime : TDosDateTime) : TSystemTime
 Function DosDateTimeToStr( DateTime : Integer) : string
 Function JFileTimeToDateTime( const FileTime : TFileTime) : TDateTime
 Function FileTimeToLocalDateTime( const FileTime : TFileTime) : TDateTime
 Function JFileTimeToDosDateTime( const FileTime : TFileTime) : TDosDateTime;
 Procedure FileTimeToDosDateTime1( const FileTime : TFileTime; var Date, Time : Word);
 Function JFileTimeToSystemTime( const FileTime : TFileTime) : TSystemTime;
 Procedure FileTimeToSystemTime1( const FileTime : TFileTime; var ST : TSystemTime);
 Function FileTimeToStr( const FileTime : TFileTime) : string
 Function SystemTimeToDosDateTime( const SystemTime : TSystemTime) : TDosDateTime
 Function JSystemTimeToFileTime( const SystemTime : TSystemTime) : TFileTime;
 Procedure SystemTimeToFileTime1( const SystemTime : TSystemTime; FTime : TFileTime);
 Function SystemTimeToStr( const SystemTime : TSystemTime) : string
 Function CreationDateTimeOfFile( const Sr : TSearchRec) : TDateTime
 Function LastAccessDateTimeOfFile( const Sr : TSearchRec) : TDateTime
 Function LastWriteDateTimeOfFile( const Sr : TSearchRec) : TDateTime
  TJclUnixTime32', 'Longword
 Function JDateTimeToUnixTime( DateTime : TDateTime) : TJclUnixTime32
 Function JUnixTimeToDateTime( const UnixTime : TJclUnixTime32) : TDateTime
 Function FileTimeToUnixTime( const AValue : TFileTime) : TJclUnixTime32
 Function UnixTimeToFileTime( const AValue : TJclUnixTime32) : TFileTime
 Function JNullStamp : TTimeStamp
 Function JCompareTimeStamps( const Stamp1, Stamp2 : TTimeStamp) : Int64
 Function JEqualTimeStamps( const Stamp1, Stamp2 : TTimeStamp) : Boolean
 Function JIsNullTimeStamp( const Stamp : TTimeStamp) : Boolean
 Function TimeStampDOW( const Stamp : TTimeStamp) : Integer
 Function FirstWeekDay( const Year, Month : Integer; var DOW : Integer) : Integer;
 Function FirstWeekDay1( const Year, Month : Integer) : Integer;
 Function LastWeekDay( const Year, Month : Integer; var DOW : Integer) : Integer;
 Function LastWeekDay1( const Year, Month : Integer) : Integer;
 Function IndexedWeekDay( const Year, Month : Integer; Index : Integer) : Integer
 Function FirstWeekendDay( const Year, Month : Integer; var DOW : Integer) : Integer;
 Function FirstWeekendDay1( const Year, Month : Integer) : Integer;
 Function LastWeekendDay( const Year, Month : Integer; var DOW : Integer) : Integer;
 Function LastWeekendDay1( const Year, Month : Integer) : Integer;
 Function IndexedWeekendDay( const Year, Month : Integer; Index : Integer) : Integer
 Function FirstDayOfWeek( const Year, Month, DayOfWeek : Integer) : Integer
 Function LastDayOfWeek( const Year, Month, DayOfWeek : Integer) : Integer
 Function IndexedDayOfWeek( const Year, Month, DayOfWeek, Index : Integer) : Integer
  FindClass('TOBJECT'),'EJclDateTimeError
end;

procedure SIRegister_JclMiscel2(CL: TPSPascalCompiler);
begin
 Function SetDisplayResolution( const XRes, YRes : DWORD) : Longint
 Function CreateDOSProcessRedirected( const CommandLine, InputFile, OutputFile : string) : Boolean
 Function WinExec32( const Cmd : string; const CmdShow : Integer) : Boolean
 Function WinExec32AndWait( const Cmd : string; const CmdShow : Integer) : Cardinal
 Function WinExec32AndRedirectOutput(const Cmd: string; var Output: string; RawOutput:Boolean):Cardinal
  TJclKillLevel', '( klNormal, klNoSignal, klTimeOut )
 Function ExitWindows( ExitCode : Cardinal) : Boolean
 Function LogOffOS( KillLevel : TJclKillLevel) : Boolean
 Function PowerOffOS( KillLevel : TJclKillLevel) : Boolean
 Function ShutDownOS( KillLevel : TJclKillLevel) : Boolean
 Function RebootOS( KillLevel : TJclKillLevel) : Boolean
 Function HibernateOS( Force, DisableWakeEvents : Boolean) : Boolean
 Function SuspendOS( Force, DisableWakeEvents : Boolean) : Boolean
 Function ShutDownDialog( const DialogMessage:string;TimeOut:DWORD;Force,Reboot:Boolean):Boolean;;
 Function ShutDownDialog1(const MachineName,DialogMessage:string;TimeOut:DWORD;Force,Reboot:Bool):Bool;
 Function AbortShutDown : Boolean;
 Function AbortShutDown1( const MachineName : string) : Boolean;
  TJclAllowedPowerOperation', '( apoHibernate, apoShutdown, apoSuspend )
  TJclAllowedPowerOperations', 'set of TJclAllowedPowerOperation
 Function GetAllowedPowerOperations : TJclAllowedPowerOperations
  FindClass('TOBJECT'),'EJclCreateProcessError
 Procedure CreateProcAsUser( const UserDomain, UserName, PassWord, CommandLine : string)
 Procedure CreateProcAsUserEx(const UserDomain,UserName,Password,CommandLine:string;const
Environment:PChar);
  // with Add(EJclCreateProcessError) do
 end;


procedure SIRegister_JclAnsiStrings(CL: TPSPascalCompiler);
begin
  //'AnsiSigns','Set').SetSet(['-', '+']);
 'C1_UPPER','LongWord( $0001);
 'C1_LOWER','LongWord( $0002);
 'C1_DIGIT','LongWord').SetUInt( $0004);
 'C1_SPACE','LongWord').SetUInt( $0008);
 'C1_PUNCT','LongWord').SetUInt( $0010);
 'C1_CNTRL','LongWord').SetUInt( $0020);
 'C1_BLANK','LongWord').SetUInt( $0040);
 'C1_XDIGIT','LongWord').SetUInt( $0080);
 'C1_ALPHA','LongWord').SetUInt( $0100);
  AnsiChar', 'Char
 Function StrIsAlpha( const S : AnsiString) : Boolean
 Function StrIsAlphaNum( const S : AnsiString) : Boolean
 Function StrIsAlphaNumUnderscore( const S : AnsiString) : Boolean
 Function StrContainsChars(const S:AnsiString;Chars:TSysCharSet; CheckAll : Boolean) : Boolean
 Function StrConsistsOfNumberChars( const S : AnsiString) : Boolean
 Function StrIsDigit( const S : AnsiString) : Boolean
 Function StrIsSubset( const S : AnsiString; const ValidChars : TSysCharSet) : Boolean
 Function StrSame( const S1, S2 : AnsiString) : Boolean
 //Function StrCenter( const S : AnsiString; L : Integer; C : AnsiChar) : AnsiString
 Function StrCharPosLower( const S : AnsiString; CharPos : Integer) : AnsiString
 Function StrCharPosUpper( const S : AnsiString; CharPos : Integer) : AnsiString
 Function StrDoubleQuote( const S : AnsiString) : AnsiString
 Function StrEnsureNoPrefix( const Prefix, Text : AnsiString) : AnsiString
 Function StrEnsureNoSuffix( const Suffix, Text : AnsiString) : AnsiString
 Function StrEnsurePrefix( const Prefix, Text : AnsiString) : AnsiString
 Function StrEnsureSuffix( const Suffix, Text : AnsiString) : AnsiString
 Function StrEscapedToString( const S : AnsiString) : AnsiString
 Function JStrLower( const S : AnsiString) : AnsiString
 Procedure StrLowerInPlace( var S : AnsiString)
 ///Procedure StrLowerBuff( S : PAnsiChar)
 Procedure JStrMove( var Dest:AnsiString; const Source:AnsiString;const ToIndex,FromIndex,Count:Integer;
 Function StrPadLeft( const S : AnsiString; Len : Integer; C : AnsiChar) : AnsiString
 Function StrPadRight( const S : AnsiString; Len : Integer; C : AnsiChar) : AnsiString
 Function StrProper( const S : AnsiString) : AnsiString
 //Procedure StrProperBuff( S : PAnsiChar)
 Function StrQuote( const S : AnsiString; C : AnsiChar) : AnsiString
 Function StrRemoveChars( const S : AnsiString; const Chars : TSysCharSet) : AnsiString
 Function StrKeepChars( const S : AnsiString; const Chars : TSysCharSet) : AnsiString
 Procedure JStrReplace(var S:AnsiString; const Search, Replace : AnsiString; Flags : TReplaceFlags)
 Function StrReplaceChar( const S : AnsiString; const Source, Replace : AnsiChar) : AnsiString
 Function StrReplaceChars(const S:AnsiString;const Chars:TSysCharSet;Replace:AnsiChar):AnsiString
 Function StrReplaceButChars(const S:AnsiString;const Chars:TSysCharSet;Replace:AnsiChar):AnsiString;
 Function StrRepeat( const S : AnsiString; Count : Integer) : AnsiString
 Function StrRepeatLength( const S : AnsiString; const L : Integer) : AnsiString
 Function StrReverse( const S : AnsiString) : AnsiString
 Procedure StrReverseInPlace( var S : AnsiString)
 Function StrSingleQuote( const S : AnsiString) : AnsiString
 Function StrSmartCase( const S : AnsiString; Delimiters : TSysCharSet) : AnsiString
 Function StrStringToEscaped( const S : AnsiString) : AnsiString
 Function StrStripNonNumberChars( const S : AnsiString) : AnsiString
 Function StrToHex( const Source : AnsiString) : AnsiString
 Function StrTrimCharLeft( const S : AnsiString; C : AnsiChar) : AnsiString
 Function StrTrimCharsLeft( const S : AnsiString; const Chars : TSysCharSet) : AnsiString
 Function StrTrimCharRight( const S : AnsiString; C : AnsiChar) : AnsiString
 Function StrTrimCharsRight( const S : AnsiString; const Chars : TSysCharSet) : AnsiString
 Function StrTrimQuotes( const S : AnsiString) : AnsiString
 Function JStrUpper( const S : AnsiString) : AnsiString
 Procedure StrUpperInPlace( var S : AnsiString)
 //Procedure StrUpperBuff( S : PAnsiChar)
 Function StrOemToAnsi( const S : AnsiString) : AnsiString
 Function StrAnsiToOem( const S : AnsiString) : AnsiString
 Procedure StrAddRef( var S : AnsiString)
 Function StrAllocSize( const S : AnsiString) : Longint
 Procedure StrDecRef( var S : AnsiString)
 //Function StrLen( S : PAnsiChar) : Integer
 Function StrLength( const S : AnsiString) : Longint
 Function StrRefCount( const S : AnsiString) : Longint
 Procedure StrResetLength( var S : AnsiString)
 Function StrCharCount( const S : AnsiString; C : AnsiChar) : Integer
 Function StrCharsCount( const S : AnsiString; Chars : TSysCharSet) : Integer
 Function StrStrCount( const S, SubS : AnsiString) : Integer
 Function StrCompare( const S1, S2 : AnsiString) : Integer
 Function StrCompareRange( const S1, S2 : AnsiString; const Index, Count : Integer) : Integer
 //Function StrFillChar( const C : AnsiChar; Count : Integer) : AnsiString;
 Function StrFillChar1( const C : Char; Count : Integer) : AnsiString;
 Function StrFillChar(const C: Char; Count: Integer): string)');
 Function IntFillChar(const I: Integer; Count: Integer): string)');
 Function ByteFillChar(const B: Byte; Count: Integer): string)');
 Function ArrFillChar(const AC: Char; Count: Integer): TCharArray;');
 Function ArrByteFillChar(const AB: Char; Count: Integer): TByteArray;
 Function StrFind( const Substr, S : AnsiString; const Index : Integer) : Integer
 //Function StrHasPrefix( const S : AnsiString; const Prefixes : array of AnsiString) : Boolean
 Function StrIndex( const S : AnsiString; const List : array of AnsiString) : Integer
 Function StrILastPos( const SubStr, S : AnsiString) : Integer
 Function StrIPos( const SubStr, S : AnsiString) : Integer
 Function StrIsOneOf( const S : AnsiString; const List : array of AnsiString) : Boolean
 Function StrLastPos( const SubStr, S : AnsiString) : Integer
 Function StrMatch( const Substr, S : AnsiString; const Index : Integer) : Integer
 Function StrMatches( const Substr, S : AnsiString; const Index : Integer) : Boolean
 Function StrNIPos( const S, SubStr : AnsiString; N : Integer) : Integer
 Function StrNPos( const S, SubStr : AnsiString; N : Integer) : Integer
 Function StrPrefixIndex( const S : AnsiString; const Prefixes : array of AnsiString) : Integer
 Function StrSearch( const Substr, S : AnsiString; const Index : Integer) : Integer
 //Function StrAfter( const SubStr, S : AnsiString) : AnsiString
 //Function StrBefore( const SubStr, S : AnsiString) : AnsiString
 Function StrBetween( const S : AnsiString; const Start, Stop : AnsiChar) : AnsiString
 Function StrChopRight( const S : AnsiString; N : Integer) : AnsiString
 Function StrLeft( const S : AnsiString; Count : Integer) : AnsiString
 Function StrMid( const S : AnsiString; Start, Count : Integer) : AnsiString
 Function StrRestOf( const S : AnsiString; N : Integer) : AnsiString
 Function StrRight( const S : AnsiString; Count : Integer) : AnsiString
 Function CharEqualNoCase( const C1, C2 : AnsiChar) : Boolean
 Function CharIsAlpha( const C : AnsiChar) : Boolean
 Function CharIsAlphaNum( const C : AnsiChar) : Boolean
 Function CharIsBlank( const C : AnsiChar) : Boolean
```

```
 Function CharIsControl( const C : AnsiChar) : Boolean
 Function CharIsDelete( const C : AnsiChar) : Boolean
 Function CharIsDigit( const C : AnsiChar) : Boolean
 Function CharIsLower( const C : AnsiChar) : Boolean
 Function CharIsNumberChar( const C : AnsiChar) : Boolean
 Function CharIsPrintable( const C : AnsiChar) : Boolean
 Function CharIsPunctuation( const C : AnsiChar) : Boolean
 Function CharIsReturn( const C : AnsiChar) : Boolean
 Function CharIsSpace( const C : AnsiChar) : Boolean
 Function CharIsUpper( const C : AnsiChar) : Boolean
 Function CharIsWhiteSpace( const C : AnsiChar) : Boolean
 Function CharType( const C : AnsiChar) : Word
 Function CharHex( const C : AnsiChar) : Byte
 Function CharLower( const C : AnsiChar) : AnsiChar
 Function CharUpper( const C : AnsiChar) : AnsiChar
 Function CharToggleCase( const C : AnsiChar) : AnsiChar
 Function CharPos( const S : AnsiString; const C : AnsiChar; const Index : Integer) : Integer
 Function CharLastPos( const S : AnsiString; const C : AnsiChar; const Index : Integer) : Integer
 Function CharIPos( const S : AnsiString; C : AnsiChar; const Index : Integer) : Integer
 Function CharReplace( var S : AnsiString; const Search, Replace : AnsiChar) : Integer
 Procedure StrIToStrings( S, Sep : AnsiString; const List : TStrings; const AllowEmptyString : Boolean)
 Procedure StrToStrings( S, Sep : AnsiString; const List : TStrings; const AllowEmptyString : Boolean)
 Function StringsToStr(const List:TStrings;const Sep:AnsiString;const AllowEmptyString:Bool):AnsiString;
 Procedure TrimStrings( const List : TStrings; DeleteIfEmpty : Boolean)
 Procedure TrimStringsRight( const List : TStrings; DeleteIfEmpty : Boolean)
 Procedure TrimStringsLeft( const List : TStrings; DeleteIfEmpty : Boolean)
 Function AddStringToStrings(const S:AnsiString;Strings:TStrings; const Unique:Boolean):Boolean
 Function BooleanToStr( B : Boolean) : AnsiString
 Function FileToString( const FileName : AnsiString) : AnsiString
 Procedure StringToFile( const FileName, Contents : AnsiString; Append : Boolean)
 Function StrToken( var S : AnsiString; Separator : AnsiChar) : AnsiString
 Procedure StrTokens( const S : AnsiString; const List : TStrings)
 Procedure StrTokenToStrings( S : AnsiString; Separator : AnsiChar; const List : TStrings)
 //Function StrWord( var S : PAnsiChar; out Word : AnsiString) : Boolean
 Function StrToFloatSafe( const S : AnsiString) : Float
 Function StrToIntSafe( const S : AnsiString) : Integer
 Procedure StrNormIndex( const StrLen : Integer; var Index : Integer; var Count : Integer);
 Function ArrayOf( List : TStrings) : TDynStringArray;
  EJclStringError', 'EJclError
 function IsClass(Address: TObject): Boolean;
 function IsObject(Address: TObject): Boolean;
 // Console Utilities
 //function ReadKey: Char;
 function IntToStrZeroPad(Value, Count: Integer): AnsiString;
 function JclGUIDToString(const GUID: TGUID): string;
 function JclStringToGUID(const S: string): TGUID;

end;


*********************************************** uPSI_JvDBUtil;
Procedure ExecuteSQLScript(Base:TDataBase; const Script: string; const Commit:TCommit;OnProgress:TOnProgress; const UserData : Integer)
Function GetQueryResult( const DatabaseName, SQL : string) : Variant
Function GetStoredProcResult( const ADatabaseName, AStoredProcName : string; AParams : array of Variant; const AResultName : string) : Variant
//Function StrFieldDesc( Field : FLDDesc) : string
Function Var2Type( V : Variant; const VarType : Integer) : Variant
Procedure CopyRecord( DataSet : TDataSet)
//Procedure AddReference( Tbl : TTable; RefName : string; RefField : Word; MasterTable : string; MasterField : Word; ModOp, DelOp : RINTQual)
Procedure AddMasterPassword( Table : TTable; pswd : string)
Procedure PackTable( Table : TTable)
Procedure PackEncryptedTable( Table : TTable; pswd : string)
Function EncodeQuotes( const S : string) : string
Function Cmp( const S1, S2 : string) : Boolean
Function SubStr( const S : string; const Index : Integer; const Separator : string) : string
Function SubStrEnd( const S : string; const Index : Integer; const Separator : string) : string
Function ReplaceString( S : string; const OldPattern, NewPattern : string) : string
Procedure GetXYByPos( const S : string; const Pos : Integer; var X, Y : Integer)
*********************************************uPSI_JvJvBDEUtils;***************
//JvBDEUtils
Function CreateDbLocate : TJvLocateObject
 //Function CheckOpen( Status : DBIResult) : Boolean
Procedure FetchAllRecords( DataSet : TBDEDataSet)
Function TransActive( Database : TDatabase) : Boolean
Function AsyncQrySupported( Database : TDatabase) : Boolean
Function GetQuoteChar( Database : TDatabase) : string
Procedure ExecuteQuery( const DbName, QueryText : string)
Procedure ExecuteQueryEx( const SessName, DbName, QueryText : string)
Function FieldLogicMap( FldType : TFieldType) : Integer
Function FieldSubtypeMap( FldType : TFieldType) : Integer Value : string; Buffer : Pointer)
Function GetAliasPath( const AliasName : string) : string
Function IsDirectory( const DatabaseName : string) : Boolean
Function GetBdeDirectory : string
Function LoginToDatabase( Database : TDatabase; OnLogin : TDatabaseLoginEvent) : Boolean
Function DataSetFindValue( ADataSet : TBDEDataSet; const Value, FieldName : string) : Boolean
Function DataSetFindLike( ADataSet : TBDEDataSet; const Value, FieldName : string) : Boolean
Function DataSetRecNo( DataSet : TDataSet) : Longint
Function DataSetRecordCount( DataSet : TDataSet) : Longint
Function DataSetPositionStr( DataSet : TDataSet) : string
Procedure DataSetShowDeleted( DataSet : TBDEDataSet; Show : Boolean)
Function CurrentRecordDeleted( DataSet : TBDEDataSet) : Boolean
Function IsFilterApplicable( DataSet : TDataSet) : Boolean
Function IsBookmarkStable( DataSet : TBDEDataSet) : Boolean
Procedure SetIndex( Table : TTable; const IndexFieldNames : string)
Procedure RestoreIndex( Table : TTable)
Procedure DeleteRange( Table : TTable; IndexFields : array of const; FieldValues : array of const)
Procedure PackTable( Table : TTable)
Procedure ReindexTable( Table : TTable)
Procedure BdeFlushBuffers
Function GetNativeHandle( Database : TDatabase; Buffer : Pointer; BufSize : Integer) : Pointer
Procedure ToggleDebugLayer( Active : Boolean; const DebugFile : string)
Procedure DbNotSupported
Procedure ExportDataSet( Source : TBDEDataSet; DestTable : TTable; TableType : TTableType; const
AsciiCharSet : string; AsciiDelimited : Boolean; MaxRecordCount : Longint)
Procedure ExportDataSetEx( Source : TBDEDataSet; DestTable : TTable; TableType : TTableType; const
AsciiCharSet:string;AsciiDelimited:Boolean;AsciiDelimiter,AsciiSeparator:Char;MaxRecordCount:Longint);
Procedure
ImportDataSet(Source:TBDEDataSet;DestTable:TTable;MaxRecordCount:Longint;Mappings:TStrings;Mode:TBatchMode);
Procedure InitRSRUN(Database: TDatabase;const ConName:string; ConType:Integer;const ConServer:string);
*********************************************uPSI_JvDateUtil;
function CurrentYear: Word;
function IsLeapYear(AYear: Integer): Boolean;
function DaysPerMonth(AYear, AMonth: Integer): Integer;
function FirstDayOfPrevMonth: TDateTime;
function LastDayOfPrevMonth: TDateTime;
function FirstDayOfNextMonth: TDateTime;
function ExtractDay(ADate: TDateTime): Word;
function ExtractMonth(ADate: TDateTime): Word;
function ExtractYear(ADate: TDateTime): Word;
function IncDate(ADate: TDateTime; Days, Months, Years: Integer): TDateTime;
function IncDay(ADate: TDateTime; Delta: Integer): TDateTime;
function IncMonth(ADate: TDateTime; Delta: Integer): TDateTime;
function IncYear(ADate: TDateTime; Delta: Integer): TDateTime;
function ValidDate(ADate: TDateTime): Boolean;
procedure DateDiff(Date1, Date2: TDateTime; var Days, Months, Years: Word);
function MonthsBetween(Date1, Date2: TDateTime): Double;
function DaysInPeriod(Date1, Date2: TDateTime): Longint;
{ Count days between Date1 and Date2 + 1, so if Date1 = Date2 result = 1 }
function DaysBetween(Date1, Date2: TDateTime): Longint;
{ The same as previous but if Date2 < Date1 result = 0 }
function IncTime(ATime: TDateTime; Hours, Minutes, Seconds, MSecs: Integer): TDateTime;
function IncHour(ATime: TDateTime; Delta: Integer): TDateTime;
function IncMinute(ATime: TDateTime; Delta: Integer): TDateTime;
function IncSecond(ATime: TDateTime; Delta: Integer): TDateTime;
function IncMSec(ATime: TDateTime; Delta: Integer): TDateTime;
function CutTime(ADate: TDateTime): TDateTime; { Set time to 00:00:00:00 }
{ String to date conversions }
function GetDateOrder(const DateFormat: string): TDateOrder;
function MonthFromName(const S: string; MaxLen: Byte): Byte;
function StrToDateDef(const S: string; Default: TDateTime): TDateTime;
function StrToDateFmt(const DateFormat, S: string): TDateTime;
function StrToDateFmtDef(const DateFormat, S: string; Default: TDateTime): TDateTime;
function DefDateFormat(FourDigitYear: Boolean): string;
function DefDateMask(BlanksChar: Char; FourDigitYear: Boolean): string;
-----------------------------------------------------------------------------
********************************** JvUtils;******************************
{ GetWordOnPos returns Word from string, S, on the cursor position, P}
function GetWordOnPos(const S: string; const P: Integer): string;
{ GetWordOnPosEx work like GetWordOnPos function, also returns Word position in iBeg, iEnd variables }
function GetWordOnPosEx(const S: string; const P: Integer; var iBeg, iEnd: Integer): string;
{ SubStr returns substring from string, S, separated with Separator string}
function SubStr(const S: string; const Index: Integer; const Separator: string): string;
{ SubStrEnd same to previous function but Index numerated from the end of string }
function SubStrEnd(const S: string; const Index: Integer; const Separator: string): string;
{ SubWord returns next Word from string, P, and offsets Pointer to the end of Word, P2 }
function SubWord(P: PChar; var P2: PChar): string;
{ NumberByWord returns the text representation of
  the number, N, in normal russian language. Was typed from Monitor magazine }
function NumberByWord(const N: Longint): string;
//  function CurrencyByWord(Value : Currency) : string; GetLineByPos returns Line number, there
  //the symbol Pos is pointed. Lines separated with #13 symbol }
function GetLineByPos(const S: string; const Pos: Integer): Integer;
{ GetXYByPos is same to previous function, but returns X position in line too}
procedure GetXYByPos(const S: string; const Pos: Integer; var X, Y: Integer);
{ ReplaceString searches for all substrings, OldPattern,in a string, S, replaces them with NewPattern }
function ReplaceString(S: string; const OldPattern, NewPattern: string): string;
{ ConcatSep concatenate S and S2 strings with Separator. if S = '', separator don't included }
function ConcatSep(const S, S2, Separator: string): string;
{ ConcatLeftSep is same to previous function, but strings concatenate right to left }
function ConcatLeftSep(const S, S2, Separator: string): string;
{ MinimizeString truncs long string, S, and appends '...' symbols, if Length of S is more than MaxLen }
function MinimizeString(const S: string; const MaxLen: Integer): string;
{ Next 4 function for russian chars transliterating.
  This functions are needed because Oem2Ansi and Ansi2Oem functions sometimes works sucks }
procedure Dos2Win(var S: string);
procedure Win2Dos(var S: string);
```

```
function Dos2WinRes(const S: string): string;
function Win2DosRes(const S: string): string;
function Win2Koi(const S: string): string;
{ Spaces returns string consists on N space chars }
function Spaces(const N: Integer): string;
{ AddSpaces add spaces to string, S, if it Length is smaller than N }
function AddSpaces(const S: string; const N: Integer): string;
{ function LastDate for russian users only } { returns date relative to current date: '' }
function LastDate(const Dat: TDateTime): string;
{ CurrencyToStr format currency, Cur, using ffCurrency float format}
function CurrencyToStr(const Cur: currency): string;
{ Cmp compares two strings and returns True if they are equal. Case-insensitive.}
function Cmp(const S1, S2: string): Boolean;
{ StringCat add S2 string to S1 and returns this string }
function StringCat(var S1: string; S2: string): string;
{ HasChar returns True, if Char, Ch, contains in string, S }
function HasChar(const Ch: Char; const S: string): Boolean;
function HasAnyChar(const Chars: string; const S: string): Boolean;
function CharInSet(const Ch: Char; const SetOfChar: TSetOfChar): Boolean;
function CountOfChar(const Ch: Char; const S: string): Integer;
function DefStr(const S: string; Default: string): string;
{**** files routines}
{ GetWinDir returns Windows folder name }
function GetWinDir: TFileName;
function GetSysDir: String;
{ GetTempDir returns Windows temporary folder name }
function GetTempDir: string;
{ GenTempFileName returns temporary file name on drive, there FileName is placed }
function GenTempFileName(FileName: string): string;
{ GenTempFileNameExt same to previous function, but returning filename has given extension, FileExt }
function GenTempFileNameExt(FileName: string; const FileExt: string): string;
{ ClearDir clears folder Dir }
function ClearDir(const Dir: string): Boolean;
{ DeleteDir clears and than delete folder Dir }
function DeleteDir(const Dir: string): Boolean;
{ FileEquMask returns True if file, FileName, is compatible with given dos file mask, Mask }
function FileEquMask(FileName, Mask: TFileName): Boolean;
{ FileEquMasks returns True if file, FileName, is compatible with given Masks.
  Masks must be separated with comma (';') }
function FileEquMasks(FileName, Masks: TFileName): Boolean;
procedure DeleteFiles(const Folder: TFileName; const Masks: string);
{ LZFileExpand expand file, FileSource into FileDest.File must be compressed,using MS Compress program }
function LZFileExpand(const FileSource, FileDest: string): Boolean;
{ FileGetInfo fills SearchRec record for specified file attributes}
function FileGetInfo(FileName: TFileName; var SearchRec: TSearchRec): Boolean;
{ HasSubFolder returns True, if folder APath contains other folders }
function HasSubFolder(APath: TFileName): Boolean;
{ IsEmptyFolder returns True, if there are no files or folders in given folder, APath}
function IsEmptyFolder(APath: TFileName): Boolean;
{ AddSlash add slash Char to Dir parameter, if needed }
procedure AddSlash(var Dir: TFileName);
{ AddSlash returns string with added slash Char to Dir parameter, if needed }
function AddSlash2(const Dir: TFileName): string;
{ AddPath returns FileName with Path, if FileName not contain any path }
function AddPath(const FileName, Path: TFileName): TFileName;
function AddPaths(const PathList, Path: string): string;
function ParentPath(const Path: TFileName): TFileName;
function FindInPath(const FileName, PathList: string): TFileName;
function FindInPaths(const fileName,paths: String): String;
{$IFNDEF BCB1}
{ BrowseForFolder displays Browse For Folder dialog }
function BrowseForFolder(const Handle: HWND; const Title: string; var Folder: string): Boolean;
{$ENDIF BCB1}
Function BrowseForFolder(const ATitle: string; AllowCreate : Boolean; var ADirectory : string;
AHelpContext : THelpContext) : Boolean
Function BrowseForComputer(const ATitle : string; AllowCreate : Boolean; var ADirectory : string;
AHelpContext : THelpContext) : Boolean
Function BrowseDirectory(var AFolderName:string;const DlgText:string;AHelpContext:THelpContext):Boolean
Function BrowseComputer(var AComputerName:string;const DlgText:string;AHelpContext:THelpContext):Boolean

{ DeleteReadOnlyFile clears R/O file attribute and delete file }
function DeleteReadOnlyFile(const FileName: TFileName): Boolean;
{ HasParam returns True, if program running with specified parameter, Param }
function HasParam(const Param: string): Boolean;
function HasSwitch(const Param: string): Boolean;
function Switch(const Param: string): string;
{ ExePath returns ExtractFilePath(ParamStr(0)) }
function ExePath: TFileName;
function CopyDir(const SourceDir, DestDir: TFileName): Boolean;
function FileTimeToDateTime(const FT: TFileTime): TDateTime;
function MakeValidFileName(const FileName: TFileName; const ReplaceBadChar: Char): TFileName;
{**** Graphic routines }
{ TTFontSelected returns True, if True Type font is selected in specified device context }
function TTFontSelected(const DC: HDC): Boolean;
{ TrueInflateRect inflates rect in other method, than InflateRect API function }
function TrueInflateRect(const R: TRect; const I: Integer): TRect;
{**** Windows routines }
{ SetWindowTop put window to top without recreating window }
procedure SetWindowTop(const Handle: HWND; const Top: Boolean);
```

```
{**** other routines }
{ KeyPressed returns True, if Key VK is now pressed }
function KeyPressed(VK: Integer): Boolean;
procedure SwapInt(var Int1, Int2: Integer);
function IntPower(Base, Exponent: Integer): Integer;
function ChangeTopException(E: TObject): TObject;
function StrToBool(const S: string): Boolean;
{$IFNDEF COMPILER3_UP}
{ AnsiStrLIComp compares S1 to S2, without case-sensitivity, up to a maximum
  Length of MaxLen bytes. The compare operation is controlled by the
  current Windows locale. The return value is the same as for CompareStr. }
function AnsiStrLIComp(S1, S2: PChar; MaxLen: Cardinal): Integer;
function AnsiStrIComp(S1, S2: PChar): Integer;
{$ENDIF}
function Var2Type(V: Variant; const VarType: Integer): Variant;
function VarToInt(V: Variant): Integer;
function VarToFloat(V: Variant): Double;
{ following functions are not documented because they are don't work properly , so don't use them }
function ReplaceSokr1(S: string; const Word, Frase: string): string;
{ ReplaceSokr1 is full equal to ReplaceString function - only for compatibility - don't use }
{ GetSubStr is full equal to SubStr function - only for compatibility - don't use }
function GetSubStr(const S: string; const Index: Integer; const Separator: Char): string;
function GetParameter: string;
function GetLongFileName(FileName: string): string;
{* from  FileCtrl}
function DirectoryExists(const Name: string): Boolean;
procedure ForceDirectories(Dir: string);
{# from  FileCtrl}
function FileNewExt(const FileName, NewExt: TFileName): TFileName;
function GetComputerID: string;
function GetComputerName: string;
{**** string routines }
{ ReplaceAllSokr searches for all substrings, Words,in a string, S, and replaces them with Frases with the
same Index.Also see RAUtilsW.ReplaceSokr1 function }
function ReplaceAllSokr(S: string; Words, Frases: TStrings): string;
{ ReplaceSokr searches the Word in a string, S, on PosBeg position,
  in the list, Words, and if founds, replaces this Word with string from another list, Frases, with the
same Index, and then update NewSelStart variable }
function ReplaceSokr(S:string;PosBeg,Len:Integer;Words,Frases:TStrings;var NewSelStart:Integer): string;
{ CountOfLines calculates the lines count in a string,each line separated from another with CrLf sequence }
function CountOfLines(const S: string): Integer;
{ DeleteEmptyLines deletes all empty lines from strings, Ss. Lines contained only spaces also deletes. }
procedure DeleteEmptyLines(Ss: TStrings);
{ SQLAddWhere addes or modifies existing where-statement, where, to the strings, SQL.
  Note: If strings SQL allready contains where-statement, it must be started on begining of any line }
procedure SQLAddWhere(SQL: TStrings; const Where: string);
{**** files routines - }
{ ResSaveToFile save resource named as Name with Typ type into file FileName.
  Resource can be compressed using MS Compress program}
function ResSaveToFile(const Typ,Name: string; const Compressed:Boolean; const FileName: string): Boolean;
function ResSaveToFileEx(Inst:HINST;Typ,Name:PChar;const Compressed:Bool;const FileName:string): Bool
function ResSaveToString(Instance: HINST; const Typ, Name: string; var S: string): Boolean;
{ Execute executes other program and waiting for it terminating, then return its Exit Code }
function ExecuteJ(const CommandLine, WorkingDirectory: string): Integer;
{ IniReadSection read section, Section, from ini-file,
  IniFileName, into strings, Ss.This function reads ALL strings from specified section.
  Note: TIninFile.ReadSection function reads only strings with '=' symbol.}
function IniReadSection(const IniFileName: TFileName; const Section: string; Ss: TStrings): Boolean;
{ LoadTextFile load text file, FileName, into string }
function LoadTextFile(const FileName: TFileName): string;
procedure SaveTextFile(const FileName: TFileName; const Source: string);
{ ReadFolder reads files list from disk folder, Folder, that are equal  Mask, into strings, FileList}
function ReadFolder(const Folder, Mask: TFileName; FileList: TStrings): Integer;
function ReadFolders(const Folder: TFileName; FolderList: TStrings): Integer;
{$IFDEF COMPILER3_UP}
{ TargetFileName - if FileName is ShortCut returns filename ShortCut linked to }
function TargetFileName(const FileName: TFileName): TFileName;
{ return filename ShortCut linked to }
function ResolveLink(const hWnd: HWND; const LinkFile: TFileName; var FileName: TFileName): HRESULT;
{$ENDIF COMPILER3_UP}
{**** Graphic routines - }
{ LoadIcoToImage loads two icons from resource named NameRes,into two image lists ALarge and ASmall}
procedure LoadIcoToImage(ALarge, ASmall: TImageList; const NameRes: string);
{ RATextOut same with TCanvas.TextOut procedure, but can clipping drawing with rectangle, RClip. }
procedure RATextOut(Canvas: TCanvas; const R, RClip: TRect; const S: string);
{ RATextOutEx same with RATextOut function, but can calculate needed height for correct output }
function RATextOutEx(Canvas:TCanvas; const R,RClip:TRect;const S: string;const CalcHeight:Boolean):Integer;
{ RATextCalcHeight calculate needed height to correct output, using RATextOut or RATextOutEx functions }
function RATextCalcHeight(Canvas: TCanvas; const R: TRect; const S: string): Integer;
{ Cinema draws some visual effect }
procedure Cinema(Canvas: TCanvas; rS {Source}, rD {Dest}: TRect);
{ Roughed fills rect with special 3D pattern }
procedure Roughed(ACanvas: TCanvas; const ARect: TRect; const AVert: Boolean);
{ BitmapFromBitmap creates new small bitmap from part of source bitmap, SrcBitmap, with specified width
and height, AWidth, AHeight and placed on a specified Index, Index in the source bitmap }
function BitmapFromBitmap(SrcBitmap: TBitmap; const AWidth, AHeight, Index: Integer): TBitmap;
{ TextWidth calculate text with for writing using standard desktop font }
function TextWidth(AStr: string): Integer;
{ DefineCursor load cursor from resource, and return available cursor number, assigned to it }
```

```
function DefineCursor(Identifer: PChar): TCursor;
{**** other routines - }
{ FindFormByClass returns first form specified class, FormClass,owned by Application global variable }
function FindFormByClass(FormClass: TFormClass): TForm;
function FindFormByClassName(FormClassName: string): TForm;
{ FindByTag returns the control with specified class, ComponentClass, from WinContol.Controls property,
  having Tag property value, equaled to Tag parameter }
function FindByTag(WinControl:TWinControl;ComponentClass:TComponentClass;const Tag:Integer):TComponent;
{ ControlAtPos2 equal to TWinControl.ControlAtPos function, but works better }
function ControlAtPos2(Parent: TWinControl; X, Y: Integer): TControl;
{ RBTag searches WinControl.Controls for checked RadioButton and returns its Tag property value }
function RBTag(Parent: TWinControl): Integer;
{ AppMinimized returns True, if Application is minimized }
function AppMinimized: Boolean;
{ MessageBox is Application.MessageBox with string (not PChar) parameters.
  if Caption parameter = '', it replaced with Application.Title }
function MessageBoxJ(const Msg: string; Caption: string;const Flags: Integer): Integer;
function MsgDlg2(const Msg, ACaption: string; DlgType: TMsgDlgType;
  Buttons: TMsgDlgButtons; HelpContext: Integer; Control: TWinControl): Integer;
function MsgDlgDef(const Msg, ACaption: string; DlgType: TMsgDlgType;
  Buttons: TMsgDlgButtons; DefButton:TMsgDlgBtn; HelpContext: Integer;Control: TWinControl): Integer;
{ Delay stop program execution to MSec msec }
procedure Delay(MSec: Longword);
procedure CenterHor(Parent: TControl; MinLeft: Integer; Controls: array of TControl);
procedure EnableControls(Control: TWinControl; const Enable: Boolean);
procedure EnableMenuItems(MenuItem: TMenuItem; const Tag: Integer; const Enable: Boolean);
procedure ExpandWidth(Parent: TControl; MinWidth: Integer; Controls: array of TControl);
function PanelBorder(Panel: TCustomPanel): Integer;
function Pixels(Control: TControl; APixels: Integer): Integer;
procedure SetChildPropOrd(Owner: TComponent; PropName: string; Value: Longint);
procedure Error(const Msg: string);
procedure ItemHtDrawEx(Canvas: TCanvas; Rect: TRect;const State: TOwnerDrawState; const Text: string;
  const HideSelColor: Boolean; var PlainItem: string; var Width: Integer; CalcWidth: Boolean);
  {ex. Text parameter:'Item 1 <b>bold</b><i>italic ITALIC <c:Red>red<c:Green>green <c:blue>blue</i>'}
function ItemHtDraw(Canvas: TCanvas; Rect: TRect; const State: TOwnerDrawState; const Text: string;
  const HideSelColor: Boolean): string;
function ItemHtWidth(Canvas: TCanvas; Rect: TRect;const State: TOwnerDrawState; const Text: string;
  const HideSelColor: Boolean): Integer;
function ItemHtPlain(const Text: string): string;
{ ClearList - clears list of TObject }
procedure ClearList(List: TList);
procedure MemStreamToClipBoard(MemStream: TMemoryStream; const Format: Word);
procedure ClipBoardToMemStream(MemStream: TMemoryStream; const Format: Word);
{ RTTI support }
function GetPropType(Obj: TObject; const PropName: string): TTypeKind;
function GetPropStr(Obj: TObject; const PropName: string): string;
function GetPropOrd(Obj: TObject; const PropName: string): Integer;
function GetPropMethod(Obj: TObject; const PropName: string): TMethod;
procedure PrepareIniSection(SS: TStrings);
{ following functions are not documented because they are don't work properly, so don't use them }
{$IFDEF COMPILER2}
function CompareMem(P1, P2: Pointer; Length: Integer): Boolean; assembler;
{$ENDIF}

procedure SIRegister_JvBoxProcs(CL: TPSPascalCompiler);
begin
 Procedure BoxMoveSelectedItems( SrcList, DstList : TWinControl)
 Procedure BoxMoveAllItems( SrcList, DstList : TWinControl)
 Procedure BoxDragOver(List:TWinControl;Source:TObject;X,Y:Int;State:TDragState;var
Accept:Bool;Sorted:Bool;
 Procedure BoxMoveFocusedItem( List : TWinControl; DstIndex : Integer)
 Procedure BoxMoveSelected( List : TWinControl; Items : TStrings)
 Procedure BoxSetItem( List : TWinControl; Index : Integer)
 Function BoxGetFirstSelection( List : TWinControl) : Integer
 Function BoxCanDropItem( List : TWinControl; X, Y : Integer; var DragIndex : Integer) : Boolean
end;

procedure SIRegister_JvCsvParse(CL: TPSPascalCompiler);
begin
 Const('MaxInitStrNum','LongInt'( 9);
Function JvAnsiStrSplit( const InString : AnsiString; const SplitChar, QuoteChar:AnsiChar; var OutStrings
: array of AnsiString; MaxSplit : Integer) : Integer
Function JvStrSplit( const InString : string; const SplitChar, QuoteChar : Char; var OutStrings : array of
string; MaxSplit : Integer) : Integer
Function JvAnsiStrSplitStrings(const InStr:AnsiString;const SplitChar,
QuoteChar:AnsiChar;OutStrs:TStrings):Integer;
Function JvAnsiStrStrip( S : AnsiString) : AnsiString
Function JvStrStrip( S : string) : string
Function GetString( var Source : AnsiString; const Separator : AnsiString) : AnsiString
Function PadString( const S : AnsiString; Len : Integer; PadChar : AnsiChar) : AnsiString
Function BuildPathName( const PathName, FileName : AnsiString) : AnsiString
Function StrEatWhiteSpace( const S : string) : string
Function HexToAscii( const S : AnsiString) : AnsiString
Function AsciiToHex( const S : AnsiString) : AnsiString
Function StripQuotes( const S1 : AnsiString) : AnsiString
Function ValidNumericLiteral( S1 : PAnsiChar) : Boolean
Function ValidIntLiteral( S1 : PAnsiChar) : Boolean
Function ValidHexLiteral( S1 : PAnsiChar) : Boolean
Function HexPCharToInt( S1 : PAnsiChar) : Integer
Function ValidStringLiteral( S1 : PAnsiChar) : Boolean
Function StripPCharQuotes( S1 : PAnsiChar) : AnsiString
Function JvValidIdentifierAnsi( S1 : PAnsiChar) : Boolean
Function JvValidIdentifier( S1 : String) : Boolean
Function JvEndChar( X : AnsiChar) : Boolean
Procedure JvGetToken( S1, S2 : PAnsiChar)
Function IsExpressionKeyword( S1 : PAnsiChar) : Boolean
Function IsKeyword( S1 : PAnsiChar) : Boolean
Function JvValidVarReference( S1 : PAnsiChar) : Boolean
Function GetParenthesis( S1, S2 : PAnsiChar) : Boolean
Procedure JvGetVarReference( S1, S2, SIdx : PAnsiChar)
Procedure JvEatWhitespaceChars( S1 : PAnsiChar);
Procedure JvEatWhitespaceChars1( S1 : PWideChar);
Function GetTokenCount : Integer
Procedure ResetTokenCount
end;

procedure SIRegister_JvDBQueryParamsForm(CL: TPSPascalCompiler);
begin
  SIRegister_TJvQueryParamsDialog(CL);
 Function EditQueryParams( DataSet : TDataSet; List : TParams; AHelpContext : THelpContext) : Boolean
end;

********************************* JvStrUtil /  JvStrUtils;*****************************
function FindNotBlankCharPos(const S: string): Integer;
function AnsiChangeCase(const S: string): string;
function GetWordOnPos(const S: string; const P: Integer): string;
function GetWordOnPosEx(const S: string; const P: Integer; var iBeg, iEnd: Integer): string;
function Cmp(const S1, S2: string): Boolean;
{ Spaces returns string consists on N space chars }
function Spaces(const N: Integer): string;
{ HasChar returns True, if char, Ch, contains in string, S }
function HasChar(const Ch: Char; const S: string): Boolean;
function HasAnyChar(const Chars: string; const S: string): Boolean;
{ SubStr returns substring from string, S, separated with Separator string}
function SubStr(const S: string; const Index: Integer; const Separator: string): string;
{ SubStrEnd same to previous function but Index numerated from the end of string }
function SubStrEnd(const S: string; const Index: Integer; const Separator: string): string;
{ ReplaceString searches for all substrings, OldPattern, in a string, S, replaces them with NewPattern }
function ReplaceString(S: string; const OldPattern, NewPattern: string): string;
function CharInSet(const Ch: Char; const SetOfChar: TSetOfChar): Boolean;
{ GetXYByPos is same to previous function, but returns X position in line too}
procedure GetXYByPos(const S: string; const Pos: Integer; var X, Y: Integer);
{ AddSlash returns string with added slash char to Dir parameter, if needed }
function AddSlash2(const Dir: TFileName): string;
{ AddPath returns FileName with Path, if FileName not contain any path }
function AddPath(const FileName, Path: TFileName): TFileName;
{ ExePath returns ExtractFilePath(ParamStr(0)) }
function ExePath: TFileName;
function LoadTextFile(const FileName: TFileName): string;
procedure SaveTextFile(const FileName: TFileName; const Source: string);
{ ConcatSep concatenate S and S2 strings with Separator. if S = '', separator don't included }
function ConcatSep(const S, S2, Separator: string): string;
{ FileEquMask returns True if file, FileName, is compatible with given dos file mask, Mask }
function FileEquMask(FileName, Mask: TFileName): Boolean;
{ FileEquMasks returns True if file, FileName, is compatible with given Masks.
  Masks must be separated with comma (';') }
function FileEquMasks(FileName, Masks: TFileName): Boolean;
function StringEndsWith(const Str, SubStr: string): Boolean;
function ExtractFilePath2(const FileName: string): string;
function StrToOem(const AnsiStr: string): string;
{ StrToOem translates a string from the Windows character set into the OEM character set. }
function OemToAnsiStr(const OemStr: string): string;
{ OemToAnsiStr translates a string from the OEM character set into the Windows character set. }
function IsEmptyStr(const S: string; const EmptyChars: TCharSet): Boolean;
{ EmptyStr returns true if the given string contains only character from the EmptyChars. }
function ReplaceStr(const S, Srch, Replace: string): string;
{ Returns string with every occurrence of Srch string replaced with Replace string. }
function DelSpace(const S: string): string;
{ DelSpace return a string with all white spaces removed. }
function DelChars(const S: string; Chr: Char): string;
{ DelChars return a string with all Chr characters removed. }
function DelBSpace(const S: string): string;
{ DelBSpace trims leading spaces from the given string. }
function DelESpace(const S: string): string;
{ DelESpace trims trailing spaces from the given string. }
function DelRSpace(const S: string): string;
{ DelRSpace trims leading and trailing spaces from the given string. }
function DelSpace1(const S: string): string;
{ DelSpace1 return a string with all non-single white spaces removed. }
function Tab2Space(const S: string; Numb: Byte): string;
{ Tab2Space converts any tabulation character in the given string to the Numb spaces characters. }
function NPos(const C: string; S: string; N: Integer): Integer;
{ NPos searches for a N-th position of substring C in a given string. }
function MakeStr(C: Char; N: Integer): string;
function MS(C: Char; N: Integer): string;
{ MakeStr return a string of length N filled with character C. }
function AddChar(C: Char; const S: string; N: Integer): string;
{ AddChar return a string left-padded to length N with characters C. }
```

```
function AddCharR(C: Char; const S: string; N: Integer): string;
{ AddCharR return a string right-padded to length N with characters C. }
function LeftStr(const S: string; N: Integer): string;
{ LeftStr return a string right-padded to length N with blanks. }
function RightStr(const S: string; N: Integer): string;
{ RightStr return a string left-padded to length N with blanks. }
function CenterStr(const S: string; Len: Integer): string;
{ CenterStr centers the characters in the string based upon the Len specified. }
function CompStr(const S1, S2: string): Integer;
{CompStr compares S1 to S2, case-sensitivity. return val is -1 if S1 < S2,0 if S1 = S2,or 1 if S1>S2. }
function CompText(const S1, S2: string): Integer;
{ CompText compares S1 to S2, without case-sensitivity. The return value is the same as for CompStr. }
function Copy2Symb(const S: string; Symb: Char): string;
{ Copy2Symb returns a substring of a string S from begining to first character Symb. }
function Copy2SymbDel(var S: string; Symb: Char): string;
{ Copy2SymbDel returns a substr of string S from to first character Symb and removes substring from S. }
function Copy2Space(const S: string): string;
{ Copy2Symb returns a substring of a string S from begining to first white space. }
function Copy2SpaceDel(var S: string): string;
{ Copy2SpaceDel returns a substring of a string S from begining to first
  white space and removes this substring from S. }
function AnsiProperCase(const S: string; const WordDelims: TCharSet): string;
{ Returns string, with the first letter of each word in uppercase,
  all other letters in lowercase. Words are delimited by WordDelims. }
function WordCount(const S: string; const WordDelims: TCharSet): Integer;
{ WordCount given a set of word delimiters, returns number of words in S. }
function WordPosition(const N: Integer; const S: string; const WordDelims: TCharSet): Integer;
{ Given a set of word delimiters, returns start position of N'th word in S. }
function ExtractWord(N: Integer; const S: string; const WordDelims: TCharSet): string;
function ExtractWordPos(N:Integer; const S:string; const WordDelims:TCharSet;var Pos: Integer): string;
function ExtractDelimited(N: Integer; const S: string; const Delims: TCharSet): string;
{ ExtractWord, ExtractWordPos and ExtractDelimited given a set of word
  delimiters, return the N'th word in S. }
function ExtractSubstr(const S: string; var Pos: Integer; const Delims: TCharSet): string;
{ ExtractSubstr given set of word delimiters,returns the substring from S,that started from position Pos.}
function IsWordPresent(const W, S: string; const WordDelims: TCharSet): Boolean;
{ IsWordPresent given a set of word delimiters, returns True if word W is present in string S. }
function QuotedString(const S: string; Quote: Char): string;
{ QuotedString returns the given string as a quoted string, using the provided Quote character. }
function ExtractQuotedString(const S: string; Quote: Char): string;
{ ExtractQuotedString removes the Quote characters from the beginning and end of a quoted string,
   and reduces pairs of Quote characters within the quoted string to a single character. }
function FindPart(const HelpWilds, InputStr: string): Integer;
{ FindPart compares a string with '?' and another, returns the position of HelpWilds in InputStr. }
function IsWild(InputStr, Wilds: string; IgnoreCase: Boolean): Boolean;
{ IsWild compares InputString with WildCard string and returns True if corresponds. }
function XorString(const Key, Src: ShortString): ShortString;
function XorEncode(const Key, Source: string): string;
function XorDecode(const Key, Source: string): string;
{ ** Command line routines ** }
{$IFNDEF COMPILER4_UP}
function FindCmdLineSwitch(const Switch: string; SwitchChars: TCharSet;IgnoreCase: Boolean): Boolean;
{$ENDIF}
function GetCmdLineArg(const Switch: string; SwitchChars: TCharSet): string;
{ ** Numeric string handling routines ** }
function Numb2USA(const S: string): string;
{ Numb2USA converts numeric string S to USA-format. }
function Dec2Hex(N: Longint; A: Byte): string;
function D2H(N: Longint; A: Byte): string;
{ Dec2Hex converts the given value to a hexadecimal string representation
  with the minimum number of digits (A) specified. }
function Hex2Dec(const S: string): Longint;
function H2D(const S: string): Longint;
{ Hex2Dec converts the given hexadecimal string to the corresponding integer value. }
function Dec2Numb(N: Longint; A, B: Byte): string;
{ Dec2Numb converts the given value to a string representation with the
  base equal to B and with the minimum number of digits (A) specified. }
function Numb2Dec(S: string; B: Byte): Longint;
{ Numb2Dec converts the given B-based numeric string to the corresponding integer value. }
function IntToBin(Value: Longint; Digits, Spaces: Integer): string;
{ IntToBin converts given value to a bin string representation with min number of digits specified. }
function IntToRoman(Value: Longint): string;
{ IntToRoman converts the given value to a roman numeric string representation. }
function RomanToInt(const S: string): Longint;
{ RomanToInt converts the given string to an integer value. If the string
  doesn't contain a valid roman numeric value, the 0 value is returned. }
procedure I64ToCardinals(I: Int64; var LowPart, HighPart: Cardinal);
procedure CardinalsToI64(var I: Int64; const LowPart, HighPart: Cardinal);
********************************** JvFileUtil;**********************************
procedure CopyFileJ(const FileName, DestName: string; ProgressControl: TControl);
procedure CopyFileEx(const FileName,DestName:string;OverwriteReadOnly,ShellDialog:Boolean;ProgressControl:
TControl);
procedure MoveFile(const FileName, DestName: TFileName);
procedure MoveFileEx(const FileName, DestName: TFileName; ShellDialog: Boolean);
{$IFDEF COMPILER4_UP}
function GetFileSize(const FileName: string): Int64;
{$ELSE}
function GetFileSize(const FileName: string): Longint;
{$ENDIF}
```

```
function FileDateTime(const FileName: string): TDateTime;
function HasAttr(const FileName: string; Attr: Integer): Boolean;
function DeleteFiles(const FileMask: string): Boolean;
function DeleteFilesEx(const FileMasks: array of string): Boolean;
function ClearDir(const Path: string; Delete: Boolean): Boolean;
function NormalDir(const DirName: string): string;
function RemoveBackSlash(const DirName: string): string;
function ValidFileName(const FileName: string): Boolean;
function DirExists(Name: string): Boolean;
procedure ForceDirectories(Dir: string);
function FileLock(Handle: Integer; Offset, LockSize: Longint): Integer;
  {$IFDEF COMPILER4_UP} overload; {$ENDIF}
{$IFDEF COMPILER4_UP}
function FileLock(Handle: Integer; Offset, LockSize: Int64): Integer; overload;
{$ENDIF}
function FileUnlock(Handle: Integer; Offset, LockSize: Longint): Integer;
{$IFDEF COMPILER4_UP} overload; {$ENDIF}
{$IFDEF COMPILER4_UP}
function FileUnlock(Handle: Integer; Offset, LockSize: Int64): Integer; overload;
{$ENDIF}
function GetTempDir: string;
function GetWindowsDir: string;
function GetSystemDir: string;
function BrowseDirectory(var AFolderName:string; const DlgText:string; AHelpContext:THelpContext): Boolean;
{$IFDEF WIN32}
function BrowseComputer(var ComputerName:string; const DlgText:string; AHelpContext:THelpContext): Boolean;
function ShortToLongFileName(const ShortName: string): string;
function ShortToLongPath(const ShortName: string): string;
function LongToShortFileName(const LongName: string): string;
function LongToShortPath(const LongName: string): string;
procedure CreateFileLink(const FileName, DisplayName: string; Folder: Integer);
procedure DeleteFileLink(const DisplayName: string; Folder: Integer);
{$ENDIF WIN32}
{$IFNDEF COMPILER3_UP}
function IsPathDelimiter(const S: string; Index: Integer): Boolean;
{$ENDIF}
Function CreateCalculatorForm( AOwner : TComponent; AHelpContext : THelpContext) : TJvCalculatorForm
Function CreatePopupCalculator( AOwner : TComponent; ABiDiMode : TBiDiMode) : TWinControl
Function CreatePopupCalculator( AOwner : TComponent) : TWinControl
Procedure SetupPopupCalculator( PopupCalc : TWinControl; APrecision : Byte; ABeepOnError : Boolean)

//****************************procedure SIRegister_VarHlpr(CL: TPSPascalCompiler);
 Procedure VariantClear( var V : Variant)
 Procedure VariantArrayRedim( var V : Variant; High : Integer)
 Procedure VariantCast( const src : Variant; var dst : Variant; vt : Integer)
 Procedure VariantCpy( const src : Variant; var dst : Variant)
 Procedure VariantAdd( const src : Variant; var dst : Variant)
 Procedure VariantSub( const src : Variant; var dst : Variant)
 Procedure VariantMul( const src : Variant; var dst : Variant)
 Procedure VariantDiv( const src : Variant; var dst : Variant)
 Procedure VariantMod( const src : Variant; var dst : Variant)
 Procedure VariantAnd( const src : Variant; var dst : Variant)
 Procedure VariantOr( const src : Variant; var dst : Variant)
 Procedure VariantXor( const src : Variant; var dst : Variant)
 Procedure VariantShl( const src : Variant; var dst : Variant)
 Procedure VariantShr( const src : Variant; var dst : Variant)
 Function VariantAdd2( const V1 : Variant; const V2 : Variant) : Variant
 Function VariantSub2( const V1 : Variant; const V2 : Variant) : Variant
 Function VariantMul2( const V1 : Variant; const V2 : Variant) : Variant
 Function VariantDiv2( const V1 : Variant; const V2 : Variant) : Variant
 Function VariantMod2( const V1 : Variant; const V2 : Variant) : Variant
 Function VariantAnd2( const V1 : Variant; const V2 : Variant) : Variant
 Function VariantOr2( const V1 : Variant; const V2 : Variant) : Variant
 Function VariantXor2( const V1 : Variant; const V2 : Variant) : Variant
 Function VariantShl2( const V1 : Variant; const V2 : Variant) : Variant
 Function VariantShr2( const V1 : Variant; const V2 : Variant) : Variant
 Function VariantNot( const V1 : Variant) : Variant
 Function VariantNeg( const V1 : Variant) : Variant
 Function VariantGetElement( const V : Variant; i1 : integer) : Variant;
 Function VariantGetElement1( const V : Variant; i1, i2 : integer) : Variant;
 Function VariantGetElement2( const V : Variant; i1, i2, i3 : integer) : Variant;
 Function VariantGetElement3( const V : Variant; i1, i2, i3, i4 : integer) : Variant;
 Function VariantGetElement4( const V : Variant; i1, i2, i3, i4, i5 : integer) : Variant;
 Procedure VariantPutElement( var V : Variant; const data : Variant; i1 : integer);
 Procedure VariantPutElement1( var V : Variant; const data : Variant; i1, i2 : integer);
 Procedure VariantPutElement2( var V : Variant; const data : Variant; i1, i2, i3 : integer);
 Procedure VariantPutElement3( var V : Variant; const data : Variant; i1, i2, i3, i4 : integer);
 Procedure VariantPutElement4( var V : Variant; const data : Variant; i1, i2, i3, i4, i5 : integer);
end;

**************************************unit uPSI_JvgUtils;*************************************
function IsEven(I: Integer): Boolean;
function InchesToPixels(DC: HDC; Value: Single; IsHorizontal: Boolean): Integer;
function CentimetersToPixels(DC: HDC; Value: Single; IsHorizontal: Boolean): Integer;
procedure SwapInt(var I1, I2: Integer);
function Spaces(Count: Integer): string;
function DupStr(const Str: string; Count: Integer): string;
function DupChar(C: Char; Count: Integer): string;
procedure Msg(const AMsg: string);
```

```
function RectW(R: TRect): Integer;
function RectH(R: TRect): Integer;
function IncColor(AColor: Longint; AOffset: Byte): Longint;
function DecColor(AColor: Longint; AOffset: Byte): Longint;
function IsItAFilledBitmap(Bmp: TBitmap): Boolean;
procedure DrawTextInRectWithAlign(DC: HDC; R: TRect; const Text: string;
  HAlign: TglHorAlign; VAlign: TglVertAlign; Style: TglTextStyle; Fnt: TFont; Flags: UINT);
procedure DrawTextInRect(DC: HDC; R:TRect; const Text:string;Style:TglTextStyle;Fnt:TFont;Flags: UINT);
procedure ExtTextOutExt(DC: HDC; X, Y: Integer; R: TRect; const Text: string;
  Style: TglTextStyle; ADelineated, ASupress3D: Boolean; FontColor, DelinColor,HighlightColor,ShadowColor:
TColor; Illumination: TJvgIllumination; Gradient: TJvgGradient; Font: TFont);
procedure DrawBox(DC:HDC; var R:TRect; Style:TglBoxStyle;BackgrColor: Longint; ATransparent: Boolean);
function DrawBoxEx(DC: HDC; ARect: TRect; Borders: TglSides;
  BevelInner, BevelOuter: TPanelBevel; Bold:Boolean; BackgrColor: Longint;ATransparent: Boolean): TRect;
procedure GradientBox(DC: HDC; R: TRect; Gradient: TJvgGradient;PenStyle, PenWidth: Integer);
procedure ChangeBitmapColor(Bitmap: TBitmap; FromColor, ToColor: TColor);
procedure DrawBitmapExt(DC: HDC; { DC - background & result}
  SourceBitmap: TBitmap; R: TRect; X, Y: Integer; //...X,Y _in_ rect!
  BitmapOption: TglWallpaperOption; DrawState: TglDrawState;
  ATransparent: Boolean; TransparentColor: TColor; DisabledMaskColor: TColor);
procedure CreateBitmapExt(DC: HDC; { DC - background & result}
  SourceBitmap: TBitmap; R: TRect; X, Y: Integer; //...X,Y _in_ rect!
  BitmapOption: TglWallpaperOption; DrawState: TglDrawState;
  ATransparent: Boolean; TransparentColor: TColor; DisabledMaskColor: TColor);
procedure BringParentWindowToTop(Wnd: TWinControl);
function GetParentForm(Control: TControl): TForm;
procedure GetWindowImageFrom(Control:TWinControl;X,Y:Integer;ADrawSelf,ADrawChildWindows:Boolean;DC:HDC)
procedure GetWindowImage(Control: TWinControl; ADrawSelf, ADrawChildWindows: Boolean; DC: HDC);
procedure GetParentImageRect(Control: TControl; Rect: TRect; DC: HDC);
function CreateRotatedFont(F: TFont; Escapement: Integer): HFONT;
function FindMainWindow(const AWndClass, AWndTitle: string): THandle;
procedure CalcShadowAndHighlightColors(BaseColor: TColor; Colors: TJvgLabelColors);
function CalcMathString(AExpression: string): Single;
function IIF(AExpression: Boolean; IfTrue, IfFalse: Variant): Variant; overload;
function IIF(AExpression: Boolean; const IfTrue, IfFalse: string): string; overload;
function GetTransparentColor(Bitmap: TBitmap; AutoTrColor: TglAutoTransparentColor): TColor;
procedure TypeStringOnKeyboard(const S: string);
function NextStringGridCell( Grid: TStringGrid ): Boolean;
procedure DrawTextExtAligned(Canvas:TCanvas;const
Text:string;R:TRect;Alignment:TglAlignment;WordWrap:Boolean);
procedure LoadComponentFromTextFile(Component: TComponent; const FileName: string);
procedure SaveComponentToTextFile(Component: TComponent; const FileName: string);
function ComponentToString(Component: TComponent): string;
procedure StringToComponent(Component: TComponent; const Value: string);
function PlayWaveResource(const ResName: string): Boolean;
function UserName: string;
function ComputerName: string;
function CreateIniFileName: string;
function ExpandString(const Str: string; Len: Integer): string;
function Transliterate(const Str: string; RusToLat: Boolean): string;
function IsSmallFonts: Boolean;
function SystemColorDepth: Integer;
function GetFileTypeJ(const FileName: string): TglFileType;
Function GetFileType( hFile : THandle) : DWORD');
function FindControlAtPt(Control: TWinControl; Pt: TPoint; MinClass: TClass): TControl;
function StrPosExt(const Str1, Str2: PChar; Str2Len: DWORD): PChar;

{ **************************************Utility routines of unit classes}
function LineStart(Buffer, BufPos: PChar): PChar
function ExtractStrings(Separators, WhiteSpace: TSysCharSet; Content: PChar;'+
  'Strings: TStrings): Integer
 Function TestStreamFormat( Stream : TStream) : TStreamOriginalFormat
 Procedure RegisterClass( AClass : TPersistentClass)
 Procedure RegisterClasses( AClasses : array of TPersistentClass)
 Procedure RegisterClassAlias( AClass : TPersistentClass; const Alias : string)
 Procedure UnRegisterClass( AClass : TPersistentClass)
 Procedure UnRegisterClasses( AClasses : array of TPersistentClass)
 Procedure UnRegisterModuleClasses( Module : HMODULE)
 Function FindGlobalComponent( const Name : string) : TComponent
 Function IsUniqueGlobalComponentName( const Name : string) : Boolean
 Function InitInheritedComponent( Instance : TComponent; RootAncestor : TClass) : Boolean
 Function InitComponentRes( const ResName : string; Instance : TComponent) : Boolean
 Function ReadComponentRes( const ResName : string; Instance : TComponent) : TComponent
 Function ReadComponentResEx( HInstance : THandle; const ResName : string) : TComponent
 Function ReadComponentResFile( const FileName : string; Instance : TComponent) : TComponent
 Procedure WriteComponentResFile( const FileName : string; Instance : TComponent)
 Procedure GlobalFixupReferences
 Procedure GetFixupReferenceNames( Root : TComponent; Names : TStrings)
 Procedure GetFixupInstanceNames(Root: TComponent; const ReferenceRootName string; Names : TStrings)
 Procedure RedirectFixupReferences( Root : TComponent; const OldRootName, NewRootName : string)
 Procedure RemoveFixupReferences( Root : TComponent; const RootName : string)
 Procedure RemoveFixups( Instance : TPersistent)
 Function FindNestedComponent( Root : TComponent; const NamePath : string) : TComponent
 Procedure BeginGlobalLoading
 Procedure NotifyGlobalLoading
 Procedure EndGlobalLoading
 Function GetUltimateOwner1( ACollection : TCollection) : TPersistent;
 Function GetUltimateOwner( APersistent : TPersistent) : TPersistent;
 // AddTypeS('TWndMethod', 'Procedure (var Message : TMessage)
 //Function MakeObjectInstance( Method : TWndMethod) : Pointer
 Procedure FreeObjectInstance( ObjectInstance : Pointer)
// Function AllocateHWnd( Method : TWndMethod) : HWND
 Procedure DeallocateHWnd( Wnd : HWND)
 Function AncestorIsValid( Ancestor : TPersistent; Root, RootAncestor : TComponent) : Boolean
****************************************unit uPSI_SqlTimSt and DB;**************************************
Procedure VarSQLTimeStampCreate4( var aDest : Variant; const ASQLTimeStamp : TSQLTimeStamp);
Function VarSQLTimeStampCreate3: Variant;
Function VarSQLTimeStampCreate2( const AValue : string) : Variant;
Function VarSQLTimeStampCreate1( const AValue : TDateTime) : Variant;
Function VarSQLTimeStampCreate( const ASQLTimeStamp : TSQLTimeStamp) : Variant;
Function VarSQLTimeStamp : TVarType
Function VarIsSQLTimeStamp( const aValue : Variant) : Boolean;
Function LocalToUTC( var TZInfo : TTimeZone; var Value : TSQLTimeStamp) : TSQLTimeStamp //beta
Function UTCToLocal( var TZInfo : TTimeZone; var Value : TSQLTimeStamp) : TSQLTimeStamp //beta
Function VarToSQLTimeStamp( const aValue : Variant) : TSQLTimeStamp
Function SQLTimeStampToStr( const Format : string; DateTime : TSQLTimeStamp) : string
Function SQLDayOfWeek( const DateTime : TSQLTimeStamp) : integer
Function DateTimeToSQLTimeStamp( const DateTime : TDateTime) : TSQLTimeStamp
Function SQLTimeStampToDateTime( const DateTime : TSQLTimeStamp) : TDateTime
Function TryStrToSQLTimeStamp( const S : string; var TimeStamp : TSQLTimeStamp) : Boolean
Function StrToSQLTimeStamp( const S : string) : TSQLTimeStamp
Procedure CheckSqlTimeStamp( const ASQLTimeStamp : TSQLTimeStamp)
Function ExtractFieldName( const Fields : string; var Pos : Integer) : string;
Function ExtractFieldName( const Fields : WideString; var Pos : Integer) : WideString;
 //'Procedure RegisterFields( const FieldClasses : array of TFieldClass)
Procedure DatabaseError( const Message : WideString; Component : TComponent)
Procedure DatabaseErrorFmt(const Message:WIdeString; const Args:array of const;Component:TComponent)
Procedure DisposeMem( var Buffer, Size : Integer)
Function BuffersEqual( Buf1, Buf2 : Pointer; Size : Integer) : Boolean
Function GetFieldProperty(DataSet:TDataSet; Control:TComponent; const FieldName: WideString): TField
Function VarTypeToDataType( VarType : Integer) : TFieldType
***********************************************unit JvStrings;*****************************************
{template functions}
function ReplaceFirst(const SourceStr, FindStr, ReplaceStr: string): string;
function ReplaceLast(const SourceStr, FindStr, ReplaceStr: string): string;
function InsertLastBlock(var SourceStr: string; BlockStr: string): Boolean;
function RemoveMasterBlocks(const SourceStr: string): string;
function RemoveFields(const SourceStr: string): string;
{http functions}
function URLEncode(const Value: AnsiString): AnsiString; // Converts string To A URLEncoded string
function URLDecode(const Value: AnsiString): AnsiString; // Converts string From A URLEncoded string
{set functions}
procedure SplitSet(AText: string; AList: TStringList);
function JoinSet(AList: TStringList): string;
function FirstOfSet(const AText: string): string;
function LastOfSet(const AText: string): string;
function CountOfSet(const AText: string): Integer;
function SetRotateRight(const AText: string): string;
function SetRotateLeft(const AText: string): string;
function SetPick(const AText: string; AIndex: Integer): string;
function SetSort(const AText: string): string;
function SetUnion(const Set1, Set2: string): string;
function SetIntersect(const Set1, Set2: string): string;
function SetExclude(const Set1, Set2: string): string;
{replace any <,> etc by &lt; &gt;}
function XMLSafe(const AText: string): string;
{simple hash, Result can be used in Encrypt}
function Hash(const AText: string): Integer;
{ Base64 encode and decode a string }
function B64Encode(const S: AnsiString): AnsiString;
function B64Decode(const S: AnsiString): AnsiString;
{Basic encryption from a Borland Example}
function Encrypt(const InString: AnsiString; StartKey, MultKey, AddKey: Integer): AnsiString;
function Decrypt(const InString: AnsiString; StartKey, MultKey, AddKey: Integer): AnsiString;
{Using Encrypt and Decrypt in combination with B64Encode and B64Decode}
function EncryptB64(const InString: AnsiString; StartKey, MultKey, AddKey: Integer): AnsiString;
function DecryptB64(const InString: AnsiString; StartKey, MultKey, AddKey: Integer): AnsiString;
procedure CSVToTags(Src, Dst: TStringList);
// converts a csv list to a tagged string list
procedure TagsToCSV(Src, Dst: TStringList);
// converts a tagged string list to a csv list
// only fieldnames from the first record are scanned ib the other records
procedure ListSelect(Src, Dst: TStringList; const AKey, AValue: string);
{selects akey=avalue from Src and returns recordset in Dst}
procedure ListFilter(Src: TStringList; const AKey, AValue: string);
{filters Src for akey=avalue}
procedure ListOrderBy(Src: TStringList; const AKey: string; Numeric: Boolean);
{orders a tagged Src list by akey}
function PosStr(const FindString, SourceString: string;
  StartPos: Integer = 1): Integer;
{ PosStr searches the first occurrence of a substring FindString in a string
  given by SourceString with case sensitivity (upper and lower case characters
  are differed). This function returns the index value of the first character
  of a specified substring from which it occurs in a given string starting with
  StartPos character index. If a specified substring is not found Q_PosStr
  returns zero. author of algorithm is Peter Morris (UK) (Faststrings unit from www.torry.ru). }
function PosStrLast(const FindString, SourceString: string): Integer;
{finds the last occurance}
```

```
function LastPosChar(const FindChar: Char; SourceString: string): Integer;
function PosText(const FindString, SourceString: string; StartPos: Integer = 1): Integer;
{ PosText searches the first occurrence of a substring FindString in a string
  given by SourceString without case sensitivity (upper and lower case characters are not differed). This
function returns the index value of the first character of a specified substring from which it occurs in a
given string starting with Start
function PosTextLast(const FindString, SourceString: string): Integer;
{finds the last occurance}
function NameValuesToXML(const AText: string): string;
{$IFDEF MSWINDOWS}
procedure LoadResourceFile(AFile: string; MemStream: TMemoryStream);
{$ENDIF MSWINDOWS}
procedure DirFiles(const ADir, AMask: string; AFileList: TStringList);
procedure RecurseDirFiles(const ADir: string; var AFileList: TStringList);
procedure RecurseDirProgs(const ADir: string; var AFileList: TStringList);
procedure SaveString(const AFile, AText: string);
Procedure SaveStringasFile( const AFile, AText : string)
function LoadStringJ(const AFile: string): string;
Function LoadStringofFile( const AFile : string) : string
Procedure SaveStringtoFile( const AFile, AText : string)
Function LoadStringfromFile( const AFile : string) : string
function HexToColor(const AText: string): TColor;
function UppercaseHTMLTags(const AText: string): string;
function LowercaseHTMLTags(const AText: string): string;
procedure GetHTMLAnchors(const AFile: string; AList: TStringList);
function RelativePath(const ASrc, ADst: string): string;
function GetToken(var Start: Integer; const SourceText: string): string;
function PosNonSpace(Start: Integer; const SourceText: string): Integer;
function PosEscaped(Start: Integer; const SourceText, FindText: string; EscapeChar: Char): Integer;
function DeleteEscaped(const SourceText: string; EscapeChar: Char): string;
function BeginOfAttribute(Start: Integer; const SourceText: string): Integer;
// parses the beginning of an attribute: space + alpha character
function ParseAttribute(var Start:Integer; const SourceText:string; var AName,AValue:string): Boolean;
//parses a name="value" attrib from Start; returns 0 when not found or else the position behind attribute
procedure ParseAttributes(const SourceText: string; Attributes: TStrings);
// parses all name=value attributes to the attributes TStringList
function HasStrValue(const AText, AName: string; var AValue: string): Boolean;
// checks if a name="value" pair exists and returns any value
function GetStrValue(const AText, AName, ADefault: string): string;
// retrieves string value from a line like:
//  name="jan verhoeven" email="jan1 dott verhoeven att wxs dott nl"
// returns ADefault when not found
function GetHTMLColorValue(const AText, AName: string; ADefault: TColor): TColor;
// same for a color
function GetIntValue(const AText, AName: string; ADefault: Integer): Integer;
// same for an Integer
function GetFloatValue(const AText, AName: string; ADefault: Extended): Extended;
// same for a float
function GetBoolValue(const AText, AName: string): Boolean;
// same for Boolean but without default
function GetValue(const AText, AName: string): string;
//retrieves string value from a line like: name="jan verhoeven" email="jan1 verhoeven att wxs dott nl"
procedure SetValue(var AText: string; const AName, AValue: string);
// sets a string value in a line
procedure DeleteValue(var AText: string; const AName: string);
// deletes a AName="value" pair from AText
procedure GetNames(AText: string; AList: TStringList);
// get a list of names from a string with name="value" pairs
function GetHTMLColor(AColor: TColor): string;
// converts a color value to the HTML hex value
function BackPosStr(Start: Integer; const FindString, SourceString: string): Integer;
// finds a string backward case sensitive
function BackPosText(Start: Integer; const FindString, SourceString: string): Integer;
// finds a string backward case insensitive
function PosRangeStr(Start: Integer; const HeadString, TailString, SourceString: string;
  var RangeBegin: Integer; var RangeEnd: Integer): Boolean;
// finds a text range, e.g. <TD>....</TD> case sensitive
function PosRangeText(Start: Integer; const HeadString, TailString, SourceString: string;
  var RangeBegin: Integer; var RangeEnd: Integer): Boolean;
// finds a text range, e.g. <TD>....</td> case insensitive
function BackPosRangeStr(Start: Integer; const HeadString, TailString, SourceString: string;
  var RangeBegin: Integer; var RangeEnd: Integer): Boolean;
// finds a text range backward, e.g. <TD>....</TD> case sensitive
function BackPosRangeText(Start: Integer; const HeadString, TailString, SourceString: string;
  var RangeBegin: Integer; var RangeEnd: Integer): Boolean;
// finds a text range backward, e.g. <TD>....</td> case insensitive
function PosTag(Start: Integer; SourceString: string; var RangeBegin: Integer;
  var RangeEnd: Integer): Boolean;
// finds a HTML or XML tag:  <....>
function InnerTag(Start: Integer; const HeadString, TailString, SourceString: string;
  var RangeBegin: Integer; var RangeEnd: Integer): Boolean;
// finds the innertext between opening and closing tags
function Easter(NYear: Integer): TDateTime;
// returns the easter date of a year.
function GetWeekNumber(Today: TDateTime): string;
//gets a datecode. Returns year and weeknumber in format: YYWW
function ParseNumber(const S: string): Integer;
// parse number returns the last position, starting from 1
function ParseDate(const S: string): Integer;
```

```
// parse a SQL style data string from positions 1,
// starts and ends with #

*****************************************unit JvJCLUtils;*******************************************

function VarIsInt(Value: Variant): Boolean;
 // VarIsInt returns VarIsOrdinal-[varBoolean]
{ PosIdx returns the index of the first appearance of SubStr in Str. The search starts at index "Index". }
function PosIdx(const SubStr, S: string; Index: Integer = 0): Integer;
function PosIdxW(const SubStr, S: WideString; Index: Integer = 0): Integer;
function PosLastCharIdx(Ch: Char; const S: string; Index: Integer = 0): Integer;
{ GetWordOnPos returns Word from string, S, on the cursor position, P}
function GetWordOnPos(const S: string; const P: Integer): string;
function GetWordOnPosW(const S: WideString; const P: Integer): WideString;
function GetWordOnPos2(const S: string; P: Integer; var iBeg, iEnd: Integer): string;
function GetWordOnPos2W(const S: WideString; P: Integer; var iBeg, iEnd: Integer): WideString;
{ GetWordOnPosEx working like GetWordOnPos function, but
  also returns Word position in iBeg, iEnd variables }
function GetWordOnPosEx(const S: string; const P: Integer; var iBeg, iEnd: Integer): string;
function GetWordOnPosExW(const S: WideString; const P: Integer; var iBeg, iEnd: Integer): WideString;
function GetNextWordPosEx(const Text: string; StartIndex: Integer; var iBeg, iEnd: Integer): string;
function GetNextWordPosExW(const Text:WideString;StartIndex:Integer; var iBeg,iEnd:Integer):WideString;
procedure GetEndPosCaret(const Text: string; CaretX, CaretY: Integer; var X, Y: Integer);
{ GetEndPosCaret returns the caret position of the last char. For the position
  after the last char of Text you must add 1 to the returned X value. }
procedure GetEndPosCaretW(const Text: WideString;CaretX,CaretY:Integer;var X,Y:Integer);
{ GetEndPosCaret returns the caret position of the last char. For the position
  after the last char of Text you must add 1 to the returned X value. }
{ SubStrBySeparator returns substring from string, S, separated with Separator string}
function SubStrBySeparator(const S:string;const Index:Integer;const Separator:string;StartIndex:Int=1):string;
function SubStrBySeparatorW(const S:WideString;const Index:Int;const Separator:WideString;StartIndex:Int:WideString;
{ SubStrEnd same to previous function but Index numerated from the end of string }
function SubStrEnd(const S: string; const Index: Integer; const Separator: string): string;
{ SubWord returns next Word from string, P, and offsets Pointer to the end of Word, P2 }
function SubWord(P: PChar; var P2: PChar): string;
function CurrencyByWord(Value: Currency): string;
{ GetLineByPos returns the Line number, there the symbol Pos is pointed. Lines separated with #13 symbol }
function GetLineByPos(const S: string; const Pos: Integer): Integer;
{ GetXYByPos is same as GetLineByPos, but returns X position in line as well}
procedure GetXYByPos(const S: string; const Pos: Integer; var X, Y: Integer);
procedure GetXYByPosW(const S: WideString; const Pos: Integer; var X, Y: Integer);
{ ReplaceString searches for all substrings, OldPattern,
  in a string, S, and replaces them with NewPattern }
function ReplaceString(S: string; const OldPattern,NewPattern: string; StartIndex:Integer = 1):string;
function ReplaceStringW(S: WideString; const OldPattern,NewPattern: WideString;StartIndex:Integer=1):WideString;
{ ConcatSep concatenate S1 and S2 strings with Separator. if S = '' then separator not included }
function ConcatSep(const S1, S2, Separator: string): string; {$IFDEF SUPPORTS_INLINE} inline; {$ENDIF SUPPORTS_INLINE}
{ ConcatLeftSep is same to previous function, but   strings concatenate right to left }
function ConcatLeftSep(const S1, S2, Separator: string): string; {$IFDEF SUPPORTS_INLINE} inline; {$ENDIF SUPPORTS_INLINE}

{ Next 4 function for russian chars transliterating.
  This functions are needed because Oem2Ansi and Ansi2Oem functions sometimes suck }
procedure Dos2Win(var S: AnsiString);
procedure Win2Dos(var S: AnsiString);
function Dos2WinRes(const S: AnsiString): AnsiString; inline; {$ENDIF SUPPORTS_INLINE}
function Win2DosRes(const S: AnsiString): AnsiString; inline; {$ENDIF SUPPORTS_INLINE}
function Win2Koi(const S: AnsiString): AnsiString;
{ FillString fills the string Buffer with Count Chars }
procedure FillString(var Buffer: string; Count: Integer; const Value: Char); overload;
procedure FillString(var Buffer: string; StartIndex, Count: Integer; const Value: Char); overload;
{ MoveString copies Count Chars from Source to Dest }
procedure MoveString(const Source: string; var Dest: string; Count: Integer); {$IFDEF SUPPORTS_INLINE} inline; {$ENDIF SUPPORTS_INLINE} overload;
procedure MoveString(const Source: string; SrcStartIdx: Integer; var Dest: string;
  DstStartIdx: Integer;Count: Integer);{$IFDEF SUPPORTS_INLINE} inline; {$ENDIF SUPPORTS_INLINE} overload;
{ FillWideChar fills Buffer with Count WideChars (2 Bytes) }
procedure FillWideChar(var Buffer; Count: Integer; const Value: WideChar);
{ MoveWideChar copies Count WideChars from Source to Dest }
procedure MoveWideChar(const Source; var Dest;Count:Integer);{$IFDEF SUPPORTS_INLINE}inline;{$ENDIF SUPPORTS_INLINE}
{ FillNativeChar fills Buffer with Count NativeChars }
procedure FillNativeChar(var Buffer; Count: Integer; const Value: Char); // D2009 internal error {$IFDEF SUPPORTS_INLINE} inline; {$ENDIF SUPPORTS_INLINE}
{ MoveWideChar copies Count WideChars from Source to Dest }
procedure MoveNativeChar(const Source; var Dest; Count: Integer); // D2009 internal error {$IFDEF SUPPORTS_INLINE} inline; {$ENDIF SUPPORTS_INLINE}
{ IsSubString() compares the sub string to the string. Indices are 1th based. }
function IsSubString(const S: string; StartIndex: Integer; const SubStr: string): Boolean;
{ Spaces returns string consists on N space chars }
function Spaces(const N: Integer): string;
{ AddSpaces adds spaces to string S, if its Length is smaller than N }
function AddSpaces(const S: string; const N: Integer): string;
function SpacesW(const N: Integer): WideString;
function AddSpacesW(const S: WideString; const N: Integer): WideString;
```

```
{ function LastDateRUS for russian users only }
{ returns date relative to current date: 'äâà äíÿ íàçàä' }
function LastDateRUS(const Dat: TDateTime): string;
{ CurrencyToStr format Currency, Cur, using ffCurrency float format}
function CurrencyToStr(const Cur: Currency): string;
{ HasChar returns True, if Char, Ch, contains in string, S }
function HasChar(const Ch: Char; const S: string): Boolean;
function HasCharW(const Ch: WideChar; const S: WideString): Boolean; inline; {$ENDIF SUPPORTS_INLINE}
function HasAnyChar(const Chars: string; const S: string): Boolean;
{$IFNDEF COMPILER12_UP}
function CharInSet(const Ch: AnsiChar;const SetOfChar:TSysCharSet):Boolean;inline;{$ENDIF SUPPORTS_INLINE}
{$ENDIF ~COMPILER12_UP}
function CharInSetW(const Ch: WideChar;const SetOfChar: TSysCharSet):Boolean;inline; {$ENDIF SUPPORTS_INLINE}
function CountOfChar(const Ch: Char; const S: string): Integer;
function DefStr(const S: string; Default: string): string; {$IFDEF SUPPORTS_INLINE} inline; {$ENDIF SUPPORTS_INLINE}
{ StrLICompW2 is a faster replacement for JclUnicode.StrLICompW }
function StrLICompW2(S1, S2: PWideChar; MaxLen: Integer): Integer;
function StrPosW(S, SubStr: PWideChar): PWideChar;
function StrLenW(S: PWideChar): Integer;
function TrimW(const S: WideString):WideString;{$IFDEF SUPPORTS_INLINE} inline;{$ENDIF SUPPORTS_INLINE}
function TrimLeftW(const S: WideString): WideString; {$IFDEF SUPPORTS_INLINE} inline; {$ENDIF SUPPORTS_INLINE}
function TrimRightW(const S: WideString): WideString; inline; {$ENDIF SUPPORTS_INLINE}
TPixelFormat', '( pfDevice, pf1bit, pf4bit, pf8bit, pf24bit )
TMappingMethod', '( mmHistogram, mmQuantize, mmTrunc784, mmTrunc666, mmTripel, mmGrayscale )
 Function GetBitmapPixelFormat( Bitmap : TBitmap) : TPixelFormat
Function GetPaletteBitmapFormat( Bitmap : TBitmap) : TPixelFormat
Procedure SetBitmapPixelFormat( Bitmap: TBitmap; PixelFormat: TPixelFormat; Method: TMappingMethod)
Function BitmapToMemoryStream(Bitmap:TBitmap;PixelFormat:TPixelFormat;Method:TMappingMethod):TMemoryStream;
Procedure GrayscaleBitmap( Bitmap : TBitmap)
Function BitmapToMemory( Bitmap : TBitmap; Colors : Integer) : TStream
Procedure SaveBitmapToFile( const Filename : string; Bitmap : TBitmap; Colors : Integer)
Function ScreenPixelFormat : TPixelFormat
Function ScreenColorCount : Integer
Procedure TileImage( Canvas : TCanvas; Rect : TRect; Image : TGraphic)
Function ZoomImage( ImageW, ImageH, MaxW, MaxH : Integer; Stretch : Boolean) : TPoint
//  SIRegister_TJvGradient(CL);

{***************************************** files routines}
procedure SetDelimitedText(List: TStrings; const Text: string; Delimiter: Char);
const
  {$IFDEF MSWINDOWS}
  DefaultCaseSensitivity = False;
  {$ENDIF MSWINDOWS}
  {$IFDEF UNIX}
  DefaultCaseSensitivity = True;
  {$ENDIF UNIX}
{ GenTempFileName returns temporary file name on
  drive, there FileName is placed }
function GenTempFileName(FileName: string): string;
{ GenTempFileNameExt same to previous function, but
  returning filename has given extension, FileExt }
function GenTempFileNameExt(FileName: string; const FileExt: string): string;
{ ClearDir clears folder Dir }
function ClearDir(const Dir: string): Boolean;
{ DeleteDir clears and than delete folder Dir }
function DeleteDir(const Dir: string): Boolean;
{ FileEquMask returns True if file, FileName, is compatible with given dos file mask, Mask }
function FileEquMask(FileName, Mask: TFileName; CaseSensitive: Boolean=DefaultCaseSensitivity): Boolean;
{ FileEquMasks returns True if file, FileName, is compatible with given Masks.
  Masks must be separated with SepPath (MSW: ';' / UNIX: ':') }
function FileEquMasks(FileName, Masks: TFileName;
  CaseSensitive: Boolean = DefaultCaseSensitivity): Boolean;
function DeleteFiles(const Folder: TFileName; const Masks: string): Boolean;
{$IFDEF MSWINDOWS}
{ LZFileExpand expand file, FileSource,
  into FileDest. Given file must be compressed, using MS Compress program }
function LZFileExpand(const FileSource, FileDest: string): Boolean;
{$ENDIF MSWINDOWS}
{ FileGetInfo fills SearchRec record for specified file attributes}
function FileGetInfo(FileName: TFileName; var SearchRec: TSearchRec): Boolean;
{ HasSubFolder returns True, if folder APath contains other folders }
function HasSubFolder(APath: TFileName): Boolean;
{ IsEmptyFolder returns True, if there are no files or
  folders in given folder, APath}
function IsEmptyFolder(APath: TFileName): Boolean;
{ AddSlash returns string with added slash Char to Dir parameter, if needed }
function AddSlash(const Dir: TFileName): string; {$IFDEF SUPPORTS_INLINE} inline; {$ENDIF SUPPORTS_INLINE}
{ AddPath returns FileName with Path, if FileName not contain any path }
function AddPath(const FileName, Path: TFileName): TFileName;
function AddPaths(const PathList, Path: string): string;
function ParentPath(const Path: TFileName): TFileName;
function FindInPath(const FileName, PathList: string): TFileName;
{ DeleteReadOnlyFile clears R/O file attribute and delete file }
function DeleteReadOnlyFile(const FileName: TFileName): Boolean;
{ HasParam returns True, if program running with specified parameter, Param }
function HasParam(const Param: string): Boolean;
function HasSwitch(const Param: string): Boolean;
function Switch(const Param: string): string;
{ ExePath returns ExtractFilePath(ParamStr(0)) }
function ExePath: TFileName; {$IFDEF SUPPORTS_INLINE} inline; {$ENDIF SUPPORTS_INLINE}
function CopyDir(const SourceDir, DestDir: TFileName): Boolean;
//function FileTimeToDateTime(const FT: TFileTime): TDateTime;
procedure FileTimeToDosDateTimeDWord(const FT: TFileTime; out Dft: DWORD);
function MakeValidFileName(const FileName: TFileName; ReplaceBadChar: Char): TFileName;
{**** Graphic routines }
{ IsTTFontSelected returns True, if True Type font is selected in specified device context }
function IsTTFontSelected(const DC: HDC): Boolean;
function KeyPressed(VK: Integer): Boolean;
Function isKeypressed: boolean;  //true if key on memo2 (shell output) is pressed
{ TrueInflateRect inflates rect in other method, than InflateRect API function }
function TrueInflateRect(const R: TRect; const I: Integer): TRect;
{**** Color routines }
procedure RGBToHSV(R, G, B: Integer; var H, S, V: Integer);
function RGBToBGR(Value: Cardinal): Cardinal;
//function ColorToPrettyName(Value: TColor): string;
//function PrettyNameToColor(const Value: string): TColor;
{**** other routines }
procedure SwapInt(var Int1, Int2: Integer); {$IFDEF SUPPORTS_INLINE} inline; {$ENDIF SUPPORTS_INLINE}
function IntPower(Base, Exponent: Integer): Integer;
function ChangeTopException(E: TObject): TObject; // Linux version writes error message to ErrOutput
function StrToBool(const S: string): Boolean;
function Var2Type(V: Variant; const DestVarType: Integer): Variant;
function VarToInt(V: Variant): Integer;
function VarToFloat(V: Variant): Double;
{ following functions are not documented because they not work properly sometimes, so do not use them }
// (rom) ReplaceStrings1, GetSubStr removed
function GetLongFileName(const FileName: string): string;
function FileNewExt(const FileName, NewExt: TFileName): TFileName;
function GetParameter: string;
function GetComputerID: string;
function GetComputerName: string;
{**** string routines }
{ ReplaceAllStrings searches for all substrings, Words,
  in a string, S, and replaces them with Frases with the same Index. }
function ReplaceAllStrings(const S: string; Words, Frases: TStrings): string;
{ ReplaceStrings searches the Word in a string, S, on PosBeg position,
  in the list, Words, and if founds, replaces this Word with string from another list,Frases, with the
same Index, and then update NewSelStart variable }
function ReplaceStrings(const S:string;PosBeg,Len:Int;Words,Frases:TStrings;var NewSelStart:Int):string;
{ CountOfLines calculates the lines count in a string, S,
  each line must be separated from another with CrLf sequence }
function CountOfLines(const S: string): Integer;
{ DeleteLines deletes all lines from strings which in the words,  words.
  The word of will be deleted from strings. }
procedure DeleteOfLines(Ss: TStrings; const Words: array of string);
{ DeleteEmptyLines deletes all empty lines from strings, Ss.
  Lines contained only spaces also deletes. }
procedure DeleteEmptyLines(Ss: TStrings);
{ SQLAddWhere addes or modifies existing where-statement, where,
  to the strings, SQL. Note: If strings SQL allready contains where-statement,
  it must be started on the begining of any line }
procedure SQLAddWhere(SQL: TStrings; const Where: string);
{**** files routines - }
{$IFDEF MSWINDOWS}
{ ResSaveToFile save resource named as Name with Typ type into file FileName.
  Resource can be compressed using MS Compress program}
function ResSaveToFile(const Typ,Name:string; const Compressed: Boolean; const FileName: string): Boolean;
function ResSaveToFileEx(Instance:HINST;Typ,Name:PChar;const Compressed:Boolean;const FileName:string):
Boolean;
function ResSaveToString(Instance: HINST; const Typ, Name: string; var S: string): Boolean;
{$ENDIF MSWINDOWS}
{ IniReadSection read section, Section, from ini-file,
  IniFileName, into strings, Ss.This function reads ALL strings from specified section.
  Note: TIninFile.ReadSection function reads only strings with '=' symbol.}
function IniReadSection(const IniFileName: TFileName; const Section: string; Ss: TStrings): Boolean;
{ LoadTextFile load text file, FileName, into string }
function LoadTextFile(const FileName: TFileName): string;
procedure SaveTextFile(const FileName: TFileName; const Source: string);
{ ReadFolder reads files list from disk folder,Folder,that are equal to mask, Mask,into strings, FileList}
function ReadFolder(const Folder, Mask: TFileName; FileList: TStrings): Integer;
function ReadFolders(const Folder: TFileName; FolderList: TStrings): Integer;
{ RATextOut same with TCanvas.TextOut procedure, but can clipping drawing with rectangle, RClip. }
procedure RATextOut(Canvas: TCanvas; const R, RClip: TRect; const S: string);
{ RATextOutEx same with RATextOut function, but can calculate needed height for correct output }
function RATextOutEx(Canvas:TCanvas;const R,RClip:TRect; const S:string;const CalcHeight:Boolean):Integer;
{ RATextCalcHeight calculate needed height for
  correct output, using RATextOut or RATextOutEx functions }
function RATextCalcHeight(Canvas: TCanvas; const R: TRect; const S: string): Integer;
{ Cinema draws some visual effect }
procedure Cinema(Canvas: TCanvas; rS {Source}, rD {Dest}: TRect);
{ Roughed fills rect with special 3D pattern }
procedure Roughed(ACanvas: TCanvas; const ARect: TRect; const AVert: Boolean);
{ BitmapFromBitmap creates new small bitmap from part of source bitmap, SrcBitmap, with specified width
and height, AWidth, AHeight and placed on a specified Index, Index in the  source bitmap }
function BitmapFromBitmap(SrcBitmap: TBitmap; const AWidth, AHeight, Index: Integer): TBitmap;
```

```
{ TextWidth calculate text with for writing using standard desktop font }
function TextWidth(const AStr: string): Integer;
{ TextHeight calculate text height for writing using standard desktop font }
function TextHeight(const AStr: string): Integer;
procedure SetChildPropOrd(Owner: TComponent; const PropName: string; Value: Longint);
procedure Error(const Msg: string);
procedure ItemHtDrawEx(Canvas: TCanvas; Rect: TRect; const State: TOwnerDrawState; const Text: string;
  const HideSelColor: Boolean; var PlainItem: string; var Width: Integer; CalcWidth: Boolean);
{example Text parameter:'Item 1<b>bold</b><i>italic ITALIC <c:Red>red<c:Green>green<c:blue>blue</i>' }
function ItemHtDraw(Canvas: TCanvas; Rect: TRect;
  const State: TOwnerDrawState; const Text: string; const HideSelColor: Boolean): string;
function ItemHtWidth(Canvas: TCanvas; Rect: TRect;
  const State: TOwnerDrawState; const Text: string; const HideSelColor: Boolean): Integer;
function ItemHtPlain(const Text: string): string;
{ ClearList - clears list of TObject }
procedure ClearList(List: TList);
procedure MemStreamToClipBoard(MemStream: TMemoryStream; const Format: Word);
procedure ClipBoardToMemStream(MemStream: TMemoryStream; const Format: Word);
{ RTTI support }
function GetPropType(Obj: TObject; const PropName: string): TTypeKind;
function GetPropStr(Obj: TObject; const PropName: string): string;
function GetPropOrd(Obj: TObject; const PropName: string): Integer;
function GetPropMethod(Obj: TObject; const PropName: string): TMethod;
procedure PrepareIniSection(Ss: TStrings);
{ following functions are not documented because they are don't work properly, so don't use them }
// (rom) from JvBandWindows to make it obsolete
function PointL(const X, Y: Longint): TPointL; {$IFDEF SUPPORTS_INLINE} inline; {$ENDIF SUPPORTS_INLINE}
// (rom) from JvBandUtils to make it obsolete
function iif(const Test: Boolean; const ATrue, AFalse: Variant): Variant;
procedure CopyIconToClipboard(Icon: TIcon; BackColor: TColor);
function CreateIconFromClipboard: TIcon;
{ begin JvIconClipboardUtils } { Icon clipboard routines }
function CF_ICON: Word;
procedure AssignClipboardIcon(Icon: TIcon);
{ Real-size icons support routines (32-bit only) }
procedure GetIconSize(Icon: HICON; var W, H: Integer);
function CreateRealSizeIcon(Icon: TIcon): HICON;
procedure DrawRealSizeIcon(Canvas: TCanvas; Icon: TIcon; X, Y: Integer);
{end JvIconClipboardUtils }
function CreateScreenCompatibleDC: HDC;
function InvalidateRect(hWnd: HWND; const lpRect: TRect; bErase: BOOL): BOOL; overload; {$IFDEF SUPPORTS_INLINE} inline; {$ENDIF}
function InvalidateRect(hWnd: HWND; lpRect: PRect; bErase: BOOL): BOOL; overload;
{ begin JvRLE } // (rom) changed API for inclusion in JCL
procedure RleCompressTo(InStream, OutStream: TStream);
procedure RleDecompressTo(InStream, OutStream: TStream);
procedure RleCompress(Stream: TStream);
procedure RleDecompress(Stream: TStream);
{ end JvRLE } { begin JvDateUtil }
function CurrentYear: Word; {$IFDEF SUPPORTS_INLINE} inline; {$ENDIF SUPPORTS_INLINE}
function IsLeapYear(AYear: Integer): Boolean;
function DaysInAMonth(const AYear, AMonth: Word): Word;
function DaysPerMonth(AYear, AMonth: Integer): Integer;
function FirstDayOfPrevMonth: TDateTime;
function LastDayOfPrevMonth: TDateTime;
function FirstDayOfNextMonth: TDateTime;
function ExtractDay(ADate: TDateTime): Word;
function ExtractMonth(ADate: TDateTime): Word;
function ExtractYear(ADate: TDateTime): Word;
function IncDate(ADate: TDateTime; Days, Months, Years: Integer): TDateTime;
function IncDay(ADate: TDateTime;Delta:Integer):TDateTime; inline; {$ENDIF SUPPORTS_INLINE}
function IncMonth(ADate: TDateTime; Delta: Integer): TDateTime;
function IncYear(ADate: TDateTime; Delta: Integer): TDateTime;
function ValidDate(ADate: TDateTime): Boolean;
procedure DateDiff(Date1, Date2: TDateTime; var Days, Months, Years: Word);
function MonthsBetween(Date1, Date2: TDateTime): Double;
function DaysInPeriod(Date1, Date2: TDateTime): Longint;
{ Count days between Date1 and Date2 + 1, so if Date1 = Date2 result = 1 }
function DaysBetween(Date1, Date2: TDateTime): Longint;
{ The same as previous but if Date2 < Date1 result = 0 }
function IncTime(ATime: TDateTime; Hours, Minutes, Seconds, MSecs: Integer): TDateTime;
function IncHour(ATime: TDateTime; Delta: Integer): TDateTime;
function IncMinute(ATime: TDateTime; Delta: Integer): TDateTime;
function IncSecond(ATime: TDateTime; Delta: Integer): TDateTime;
function IncMSec(ATime: TDateTime; Delta: Integer): TDateTime;
function CutTime(ADate: TDateTime): TDateTime;  { Set time to 00:00:00:00 }
{ String to date conversions }
function GetDateOrder(const DateFormat: string): TDateOrder;
function MonthFromName(const S: string; MaxLen: Byte): Byte;
function StrToDateDef(const S: string; Default: TDateTime): TDateTime;
function StrToDateFmt(const DateFormat, S: string): TDateTime;
function StrToDateFmtDef(const DateFormat, S: string; Default: TDateTime): TDateTime;
//function DefDateFormat(AFourDigitYear: Boolean): string;
//function DefDateMask(BlanksChar: Char; AFourDigitYear: Boolean): string;
function FormatLongDate(Value: TDateTime): string;
function FormatLongDateTime(Value: TDateTime): string;
{ end JvDateUtil }
function BufToBinStr(Buf: Pointer; BufSize: Integer): string;
function BinStrToBuf(Value: string; Buf: Pointer; BufSize: Integer): Integer;
```

```
{ begin JvStrUtils } { ** Common string handling routines ** }
{$IFDEF UNIX}
function iconversion(InP: PAnsiChar; OutP: Pointer; InBytes, OutBytes: Cardinal;
  const ToCode, FromCode: AnsiString): Boolean;
function iconvString(const S, ToCode, FromCode: AnsiString): string;
function iconvWideString(const S: WideString; const ToCode, FromCode: AnsiString): WideString;
function OemStrToAnsi(const S: AnsiString): AnsiString;
function AnsiStrToOem(const S: AnsiString): AnsiString;
{$ENDIF UNIX}
function StrToOem(const AnsiStr: AnsiString): AnsiString;
{ StrToOem translates a string from the Windows character set into the OEM character set. }
function OemToAnsiStr(const OemStr: AnsiString): AnsiString;
{ OemToAnsiStr translates a string from the OEM character set into the Windows character set. }
function IsEmptyStr(const S: string; const EmptyChars: TSysCharSet): Boolean;
{ EmptyStr returns True if the given string contains only character from the EmptyChars. }
function ReplaceStr(const S, Srch, Replace: string): string;
{ Returns string with every occurrence of Srch string replaced with Replace string. }
function DelSpace(const S: string): string;
{ DelSpace return a string with all white spaces removed. }
function DelChars(const S: string; Chr: Char): string;
{ DelChars return a string with all Chr characters removed. }
function DelBSpace(const S: string): string;
{ DelBSpace trims leading spaces from the given string. }
function DelESpace(const S: string): string;
{ DelESpace trims trailing spaces from the given string. }
function DelRSpace(const S: string): string;
{ DelRSpace trims leading and trailing spaces from the given string. }
function DelSpace1(const S: string): string;
{ DelSpace1 return a string with all non-single white spaces removed. }
function Tab2Space(const S: string; Numb: Byte): string;
{ Tab2Space converts any tabulation character in the given string to the
  Numb spaces characters. }
function NPos(const C: string; S: string; N: Integer): Integer;
{ NPos searches for a N-th position of substring C in a given string. }
function MakeStr(C: Char; N: Integer): string; overload;
{$IFNDEF COMPILER12_UP}
function MakeStr(C: WideChar; N: Integer): WideString; overload;
{$ENDIF !COMPILER12_UP}
function MS(C: Char; N: Integer): string; {$IFDEF SUPPORTS_INLINE} inline; {$ENDIF SUPPORTS_INLINE}
{ MakeStr return a string of length N filled with character C. }
function AddChar(C: Char; const S: string; N: Integer): string;
{ AddChar return a string left-padded to length N with characters C. }
function AddCharR(C: Char; const S: string; N: Integer): string;
{ AddCharR return a string right-padded to length N with characters C. }
function LeftStr(const S: string; N: Integer): string;
{ LeftStr return a string right-padded to length N with blanks. }
function RightStr(const S: string; N: Integer): string;
{ RightStr return a string left-padded to length N with blanks. }
function CenterStr(const S: string; Len: Integer): string;
{ CenterStr centers the characters in the string based upon the Len specified. }
function CompStr(const S1, S2: string): Integer; {$IFDEF SUPPORTS_INLINE} inline; {$ENDIF SUPPORTS_INLINE}
{ CompStr compares S1 to S2, with case-sensitivity. The return value is
  -1 if S1 < S2, 0 if S1 = S2, or 1 if S1 > S2. }
function CompText(const S1, S2: string):Integer; {$IFDEF SUPPORTS_INLINE} inline; {$ENDIF SUPPORTS_INLINE}
{ CompText compares S1 to S2, without case-sensitivity. The return value is the same as for CompStr. }
function Copy2Symb(const S: string; Symb: Char): string;
{ Copy2Symb returns a substring of a string S from begining to first character Symb. }
function Copy2SymbDel(var S: string; Symb: Char): string;
{ Copy2SymbDel returns a substring of a string S from begining to first
  character Symb and removes this substring from S. }
function Copy2Space(const S: string): string;
{ Copy2Symb returns a substring of a string S from begining to first white space. }
function Copy2SpaceDel(var S: string): string;
{ Copy2SpaceDel returns a substring of a string S from begining to first
  white space and removes this substring from S. }
function AnsiProperCase(const S: string; const WordDelims: TSysCharSet): string;
{ Returns string, with the first letter of each word in uppercase,
  all other letters in lowercase. Words are delimited by WordDelims. }
function WordCount(const S: string; const WordDelims: TSysCharSet): Integer;
{ WordCount given a set of word delimiters, returns number of words in S. }
function WordPosition(const N: Integer; const S: string; const WordDelims: TSysCharSet): Integer;
{ Given a set of word delimiters, returns start position of N'th word in S. }
function ExtractWord(N: Integer; const S: string; const WordDelims: TSysCharSet): string;
function ExtractWordPos(N: Integer;const S: string;const WordDelims:TSysCharSet;var Pos: Integer): string;
function ExtractDelimited(N: Integer; const S: string;const Delims: TSysCharSet): string;
{ ExtractWord, ExtractWordPos and ExtractDelimited given a set of word
  delimiters, return the N'th word in S. }
function ExtractSubstr(const S: string; var Pos: Integer;const Delims: TSysCharSet): string;
{ ExtractSubstr given a set of word delimiters, returns the substring from S,
  that started from position Pos. }
function IsWordPresent(const W, S: string; const WordDelims: TSysCharSet): Boolean;
{ IsWordPresent given a set of word delimiters, returns True if word W is present in string S. }
function QuotedString(const S: string; Quote: Char): string;
{ QuotedString returns the given string as a quoted string, using the provided Quote character. }
function ExtractQuotedString(const S: string; Quote: Char): string;
{ ExtractQuotedString removes the Quote characters from the beginning and
  end of a quoted string, and reduces pairs of Quote characters within quoted string to single character.}
function FindPart(const HelpWilds, InputStr: string): Integer;
{ FindPart compares a string with '?' and another, returns the position of HelpWilds in InputStr. }
```

```
function IsWild(InputStr, Wilds: string; IgnoreCase: Boolean): Boolean;
{ IsWild compares InputString with WildCard string and returns True if corresponds. }
function XorString(const Key, Src: ShortString): ShortString;
function XorEncode(const Key, Source: string): string;
function XorDecode(const Key, Source: string): string;
{ ** Command line routines ** }
function GetCmdLineArg(const Switch: string; ASwitchChars: TSysCharSet): string;
{ ** Numeric string handling routines ** }
function Numb2USA(const S: string): string;
{ Numb2USA converts numeric string S to USA-format. }
function Dec2Hex(N: Longint; A: Byte): string; {$IFDEF SUPPORTS_INLINE} inline; {$ENDIF SUPPORTS_INLINE}
{ Dec2Hex converts the given value to a hexadecimal string representation
  with the minimum number of digits (A) specified. }
function Hex2Dec(const S: string): Longint;
{ Hex2Dec converts the given hexadecimal string to the corresponding integer value. }
function Dec2Numb(N: Int64; A, B: Byte): string;
{ Dec2Numb converts the given value to a string representation with the
  base equal to B and with the minimum number of digits (A) specified. }
function Numb2Dec(S: string; B: Byte): Int64;
{ Numb2Dec converts the given B-based numeric string to the corresponding
  integer value. }
function IntToBin(Value: Longint; Digits, Spaces: Integer): string;
{ IntToBin converts the given value to a binary string representation
  with the minimum number of digits specified. }
function IntToRoman(Value: Longint): string;
{ IntToRoman converts the given value to a roman numeric string representation. }
function RomanToInt(const S: string): Longint;
{ RomanToInt converts the given string to an integer value. If the string
  doesn't contain a valid roman numeric value, the 0 value is returned. }
function FindNotBlankCharPos(const S: string): Integer;
function FindNotBlankCharPosW(const S: WideString): Integer;
function AnsiChangeCase(const S: string): string;
function WideChangeCase(const S: string): string;
function StartsText(const SubStr, S: string): Boolean;
function EndsText(const SubStr, S: string): Boolean;
function DequotedStr(const S: string; QuoteChar: Char = ''''): string;
function AnsiDequotedStr(const S: string; AQuote: Char): string; //follow Delphi 2009's "Ansi" prefix
{end JvStrUtils}
{$IFDEF UNIX}
function GetTempFileName(const Prefix: AnsiString): AnsiString;
{$ENDIF UNIX}
{ begin JvFileUtil }
function FileDateTime(const FileName: string): TDateTime;
function HasAttr(const FileName: string; Attr: Integer): Boolean;
function DeleteFilesEx(const FileMasks: array of string): Boolean;
function NormalDir(const DirName: string): string;
function RemoveBackSlash(const DirName: string): string; // only for Windows/DOS Paths
function ValidFileName(const FileName: string): Boolean;
{$IFDEF MSWINDOWS}
function FileLock(Handle: Integer; Offset, LockSize: Longint): Integer; overload;
function FileLock(Handle: Integer; Offset, LockSize: Int64): Integer; overload;
function FileUnlock(Handle: Integer; Offset, LockSize: Longint): Integer; overload;
function FileUnlock(Handle: Integer; Offset, LockSize: Int64): Integer; overload;
{$ENDIF MSWINDOWS}
function GetWindowsDir: string;
function GetSystemDir: string;
function ShortToLongFileName(const ShortName: string): string;
function LongToShortFileName(const LongName: string): string;
function ShortToLongPath(const ShortName: string): string;
function LongToShortPath(const LongName: string): string;
{$IFDEF MSWINDOWS}
procedure CreateFileLink(const FileName, DisplayName: string; Folder: Integer);
procedure DeleteFileLink(const DisplayName: string; Folder: Integer);
{$ENDIF MSWINDOWS}
{ end JvFileUtil }
// Works like PtInRect but includes all edges in comparision
function PtInRectInclusive(R: TRect; Pt: TPoint): Boolean;
// Works like PtInRect but excludes all edges from comparision
function PtInRectExclusive(R: TRect; Pt: TPoint): Boolean;
function FourDigitYear: Boolean; {$IFDEF SUPPORTS_DEPRECATED} deprecated; {$ENDIF}
function IsFourDigitYear: Boolean;
{ moved from JvJVCLUTils }
//Open an object with the shell (url or something like that)
function OpenObject(const Value: string): Boolean; overload;
function OpenObject(Value: PChar): Boolean; overload;
{$IFDEF MSWINDOWS}
//Raise the last Exception
procedure RaiseLastWin32; overload;
procedure RaiseLastWin32(const Text: string); overload;
//Raise the last Exception with a small comment from your part { GetFileVersion returns the most
significant 32 bits of a file's binary version number. Typically, this includes the major and minor
version placed together in one 32-bit Integer. I
function GetFileVersion(const AFileName: string): Cardinal;
{$EXTERNALSYM GetFileVersion}
//Get version of Shell.dll
function GetShellVersion: Cardinal;
{$EXTERNALSYM GetShellVersion}
// CD functions  on HW
procedure OpenCdDrive;
procedure CloseCdDrive;
// returns True if Drive is accessible
function DiskInDrive(Drive: Char): Boolean;
{$ENDIF MSWINDOWS}
//Same as linux function ;)
procedure PError(const Text: string);
// execute a program without waiting
procedure Exec(const FileName, Parameters, Directory: string);
// execute a program and wait for it to finish
function ExecuteAndWait(CmdLine:string;const WorkingDirectory:string;Visibility:Integer=SW_SHOW): Int;
// returns True if this is the first instance of the program that is running
function FirstInstance(const ATitle: string): Boolean;
// restores a window based on it's classname and Caption. Either can be left empty
// to widen the search
procedure RestoreOtherInstance(const MainFormClassName, MainFormCaption: string);
// manipulate the traybar and start button
procedure HideTraybar;
procedure ShowTraybar;
procedure ShowStartButton(Visible: Boolean = True);
// (rom) SC_MONITORPOWER is documented as Win 95 only(rom) better do some testing set monitor functions
procedure MonitorOn;
procedure MonitorOff;
procedure LowPower;
// send a key to the window named AppName
function SendKey(const AppName: string; Key: Char): Boolean;
{$IFDEF MSWINDOWS}
// returns a list of all win currently visible, the Objects property is filled with their window handle
procedure GetVisibleWindows(List: TStrings);
Function GetVisibleWindowsF( List : TStrings):TStrings');
// associates an extension to a specific program
procedure AssociateExtension(const IconPath, ProgramName, Path, Extension: string);
procedure AddToRecentDocs(const FileName: string);
function GetRecentDocs: TStringList;
{$ENDIF MSWINDOWS}
function CharIsMoney(const Ch: Char): Boolean;
//function StrToCurrDef(const Str: string; Def: Currency): Currency;
function IntToExtended(I: Integer): Extended;
{ GetChangedText works out the new text given the current cursor pos & the key pressed
  It is not very useful in other contexts, but in this unit as it is needed in both MemoEx and TypedEdit }
function GetChangedText(const Text: string; SelStart, SelLength: Integer; Key: Char): string;
function MakeYear4Digit(Year, Pivot: Integer): Integer;
//function StrIsInteger(const S: string): Boolean;
function StrIsFloatMoney(const Ps: string): Boolean;
function StrIsDateTime(const Ps: string): Boolean;
function PreformatDateString(Ps: string): string;
function BooleanToInteger(const B: Boolean): Integer;
function StringToBoolean(const Ps: string): Boolean;
function SafeStrToDateTime(const Ps: string): TDateTime;
function SafeStrToDate(const Ps: string): TDateTime;
function SafeStrToTime(const Ps: string): TDateTime;
function StrDelete(const psSub, psMain: string): string;
  { returns the fractional value of pcValue}
function TimeOnly(pcValue: TDateTime): TTime;
{ returns the integral value of pcValue }
function DateOnly(pcValue: TDateTime): TDate;
type TdtKind = (dtkDateOnly, dtkTimeOnly, dtkDateTime);
const { TDateTime value used to signify Null value}
  NullEquivalentDate: TDateTime = 0.0;
function DateIsNull(const pdtValue: TDateTime; const pdtKind: TdtKind): Boolean;
// Replacement for Win32Check to avoid platform specific warnings in D6
function OSCheck(RetVal: Boolean): Boolean;
{ Shortens a fully qualified Path name so that it can be drawn with a specified length limit.
  Same as FileCtrl.MinimizeName in functionality (but not implementation). Included here to
  not be forced to use FileCtrl unnecessarily }
function MinimizeFileName(const FileName: string; Canvas: TCanvas; MaxLen: Integer): string;
function MinimizeText(const Text: string; Canvas: TCanvas; MaxWidth: Integer): string;
{ MinimizeString trunactes long string, S, and appends'...'symbols,if Length of S is more than MaxLen }
function MinimizeString(const S: string; const MaxLen: Integer): string;
procedure RunDll32Internal(Wnd:THandle; const DLLName,FuncName,CmdLine:string;CmdShow:Integer=
SW_SHOWDEFAULT);
{ GetDLLVersion loads DLLName, gets a pointer to DLLVersion function and calls it, returning major and
minor version values from the function. Returns False if DLL not loaded or if GetDLLVersion couldn't be
found.}
function GetDLLVersion(const DLLName: string; var pdwMajor, pdwMinor: Integer): Boolean;
{$ENDIF MSWINDOWS}
procedure ResourceNotFound(ResID: PChar);
function EmptyRect: TRect;
function RectWidth(R: TRect): Integer;
function RectHeight(R: TRect): Integer;
function CompareRect(const R1, R2: TRect): Boolean;
procedure RectNormalize(var R: TRect);
function RectIsSquare(const R: TRect): Boolean;
function RectSquare(var ARect: TRect; AMaxSize: Integer = -1): Boolean;
//If AMaxSize = -1 ,then auto calc Square's max size
{$IFDEF MSWINDOWS}
procedure FreeUnusedOle;
function GetWindowsVersion: string;
function LoadDLL(const LibName: string): THandle;
function RegisterServer(const ModuleName: string): Boolean;
function UnregisterServer(const ModuleName: string): Boolean;
{$ENDIF MSWINDOWS}
{ String routines }
function GetEnvVar(const VarName: string): string;
function AnsiUpperFirstChar(const S: string): string; //follow Delphi 2009's example with "Ansi" prefix
function StringToPChar(var S: string): PChar;
function StrPAlloc(const S: string): PChar;
procedure SplitCommandLine(const CmdLine: string; var ExeName, Params: string);
function DropT(const S: string): string;
{ Memory routines }
function AllocMemo(Size: Longint): Pointer;
function ReallocMemo(fpBlock: Pointer; Size: Longint): Pointer;
procedure FreeMemo(var fpBlock: Pointer);
function GetMemoSize(fpBlock: Pointer): Longint;
function CompareMem(fpBlock1, fpBlock2: Pointer; Size: Cardinal): Boolean;
{ Manipulate huge pointers routines }
procedure HugeInc(var HugePtr: Pointer; Amount: Longint);
procedure HugeDec(var HugePtr: Pointer; Amount: Longint);
function HugeOffset(HugePtr: Pointer; Amount: Longint): Pointer;
procedure HugeMove(Base: Pointer; Dst, Src, Size: Longint);
procedure HMemCpy(DstPtr, SrcPtr: Pointer; Amount: Longint);
function WindowClassName(Wnd: THandle): string;
procedure SwitchToWindow(Wnd: THandle; Restore: Boolean);
procedure ActivateWindow(Wnd: THandle);
procedure ShowWinNoAnimate(Handle: THandle; CmdShow: Integer);
procedure KillMessage(Wnd: THandle; Msg: Cardinal);
{ SetWindowTop put window to top without recreating window }
procedure SetWindowTop(const Handle: THandle; const Top: Boolean);
procedure CenterWindow(Wnd: THandle);
function MakeVariant(const Values: array of Variant): Variant;
{ Convert dialog units to pixels and backwards }
{$IFDEF MSWINDOWS}
function DialogUnitsToPixelsX(DlgUnits: Word): Word;
function DialogUnitsToPixelsY(DlgUnits: Word): Word;
function PixelsToDialogUnitsX(PixUnits: Word): Word;
function PixelsToDialogUnitsY(PixUnits: Word): Word;
{$ENDIF MSWINDOWS}
function GetUniqueFileNameInDir(const Path, FileNameMask: string): string;
{$IFDEF BCB}
function FindPrevInstance(const MainFormClass: ShortString;const ATitle: string): THandle;
function ActivatePrevInstance(const MainFormClass: ShortString;const ATitle: string): Boolean;
{$ELSE}
function FindPrevInstance(const MainFormClass, ATitle: string): THandle;
function ActivatePrevInstance(const MainFormClass, ATitle: string): Boolean;
{$ENDIF BCB}
{$IFDEF MSWINDOWS}
{ BrowseForFolderNative displays Browse For Folder dialog }
function BrowseForFolderNative(const Handle: THandle; const Title: string; var Folder: string): Boolean;
{$ENDIF MSWINDOWS}
procedure AntiAlias(Clip: TBitmap);
procedure AntiAliasRect(Clip: TBitmap; XOrigin, YOrigin,XFinal, YFinal: Integer);
procedure CopyRectDIBits(ACanvas: TCanvas; const DestRect: TRect;
  ABitmap: TBitmap; const SourceRect: TRect);
function IsTrueType(const FontName: string): Boolean;
// Removes all non-numeric characters from AValue and returns the resulting string
function TextToValText(const AValue: string): string;
Function ExecRegExpr( const ARegExpr, AInputStr : RegExprString) : boolean
Procedure SplitRegExpr( const ARegExpr, AInputStr : RegExprString; APieces : TStrings)
Function ReplaceRegExpr(const ARegExpr,AInputStr,
AReplaceStr:RegExprString;AUseSubstitution:bool):RegExprString;
Function QuoteRegExprMetaChars( const AStr : RegExprString) : RegExprString
Function RegExprSubExpressions(const ARegExpr:string; ASubExprs:TStrings; AExtendedSyntax : boolean) :

**********************************************************unit uPSI_JvTFUtils;
 Function JExtractYear( ADate : TDateTime) : Word
 Function JExtractMonth( ADate : TDateTime) : Word
 Function JExtractDay( ADate : TDateTime) : Word
 Function ExtractHours( ATime : TDateTime) : Word
 Function ExtractMins( ATime : TDateTime) : Word
 Function ExtractSecs( ATime : TDateTime) : Word
 Function ExtractMSecs( ATime : TDateTime) : Word
 Function FirstOfMonth( ADate : TDateTime) : TDateTime
 Function GetDayOfNthDOW( Year, Month, DOW, N : Word) : Word
 Function GetWeeksInMonth( Year, Month : Word; StartOfWeek : Integer) : Word
 Procedure IncBorlDOW( var BorlDOW : Integer; N : Integer)
 Procedure IncDOW( var DOW : TTFDayOfWeek; N : Integer)
 Procedure IncDays( var ADate : TDateTime; N : Integer)
 Procedure IncWeeks( var ADate : TDateTime; N : Integer)
 Procedure IncMonths( var ADate : TDateTime; N : Integer)
 Procedure IncYears( var ADate : TDateTime; N : Integer)
 Function EndOfMonth( ADate : TDateTime) : TDateTime
 Function IsFirstOfMonth( ADate : TDateTime) : Boolean
 Function IsEndOfMonth( ADate : TDateTime) : Boolean
 Procedure EnsureMonth( Month : Word)
 Procedure EnsureDOW( DOW : Word)
 Function EqualDates( D1, D2 : TDateTime) : Boolean
 Function Lesser( N1, N2 : Integer) : Integer
 Function Greater( N1, N2 : Integer) : Integer
 Function GetDivLength( TotalLength, DivCount, DivNum : Integer) : Integer
 Function GetDivNum( TotalLength, DivCount, X : Integer) : Integer
 Function GetDivStart( TotalLength, DivCount, DivNum : Integer) : Integer
 Function DOWToBorl( ADOW : TTFDayOfWeek) : Integer
 Function BorlToDOW( BorlDOW : Integer) : TTFDayOfWeek
 Function DateToDOW( ADate : TDateTime) : TTFDayOfWeek
 Procedure CalcTextPos( HostRect : TRect; var TextLeft, TextTop:Integer; var TextBounds : TRect;
AFont:TFont;AAngle: Integer; HAlign : TAlignment; VAlign : TJvTFVAlignment; ATxt : string)
 Procedure DrawAngleText( ACanvas : TCanvas; HostRect : TRect; var TextBounds : TRect; AAngle : Integer;
HAlign : TAlignment; VAlign : TJvTFVAlignment; ATxt : string)
 Function JRectWidth( ARect : TRect) : Integer
 Function JRectHeight( ARect : TRect) : Integer
 Function JEmptyRect : TRect
 Function IsClassByName( Obj : TObject; ClassName : string) : Boolean

procedure SIRegister_MSysUtils(CL: TPSPascalCompiler);
begin
 Procedure HideTaskBarButton( hWindow : HWND)
 Function msLoadStr( ID : Integer) : String
 Function msFormat( fmt : String; params : array of const) : String
 Function msFileExists( const FileName : String) : Boolean
 Function msIntToStr( Int : Int64) : String
 Function msStrPas( const Str : PChar) : String
 Function msRenameFile( const OldName, NewName : String) : Boolean
 Function CutFileName( s : String) : String
 Function GetVersionInfo( var VersionString : String) : DWORD
 Function FormatTime( t : Cardinal) : String
 Function msCreateDir( const Dir : string) : Boolean
 Function SetAutoRun( NeedAutoRun : Boolean; AppName : String) : Boolean
 Function SetTreeViewStyle( const hTV : HWND; dwNewStyle : dword) : DWORD
 Function msStrLen( Str : PChar) : Integer
 Function msDirectoryExists( const Directory : String) : Boolean
 Function GetFolder( hWnd : hWnd; RootDir : Integer; Caption : String) : String
 Function SetBlendWindow( hWnd : HWND; AlphaBlend : Byte) : LongBool
 Function EditWindowProc( hWnd : HWND; Msg : UINT; wParam : WPARAM; lParam : LPARAM) : LRESULT
 Procedure SetEditWndProc( hWnd : HWND; ptr : TObject)
 Function GetTextFromFile( Filename : String) : string
 Function IsTopMost( hWnd : HWND) : Bool // 'LWA_ALPHA','LongWord').SetUInt( $00000002);
 Function msStrToIntDef( const s : String; const i : Integer) : Integer
 Function msStrToInt( s : String) : Integer
 Function GetItemText( hDlg : THandle; ID : DWORD) : String
end;

procedure SIRegister_ESBMaths2(CL: TPSPascalCompiler);
begin
  //TDynFloatArray', 'array of Extended
  TDynLWordArray', 'array of LongWord
  TDynLIntArray', 'array of LongInt
  TDynFloatMatrix', 'array of TDynFloatArray
  TDynLWordMatrix', 'array of TDynLWordArray
  TDynLIntMatrix', 'array of TDynLIntArray
 Function SquareAll( const X : TDynFloatArray) : TDynFloatArray
 Function InverseAll( const X : TDynFloatArray) : TDynFloatArray
 Function LnAll( const X : TDynFloatArray) : TDynFloatArray
 Function Log10All( const X : TDynFloatArray) : TDynFloatArray
 Function LinearTransform( const X : TDynFloatArray; Offset, Scale : Extended) : TDynFloatArray
 Function AddVectors( const X, Y : TDynFloatArray) : TDynFloatArray
 Function SubVectors( const X, Y : TDynFloatArray) : TDynFloatArray
 Function MultVectors( const X, Y : TDynFloatArray) : TDynFloatArray
 Function DotProduct( const X, Y : TDynFloatArray) : Extended
 Function MNorm( const X : TDynFloatArray) : Extended
 Function MatrixIsRectangular( const X : TDynFloatMatrix) : Boolean
 Procedure MatrixDimensions(const X:TDynFloatMatrix;var Rows,Columns:LongWord;var Rectangular:Boolean;
 Function MatrixIsSquare( const X : TDynFloatMatrix) : Boolean
 Function MatricesSameDimensions( const X, Y : TDynFloatMatrix) : Boolean
 Function AddMatrices( const X, Y : TDynFloatMatrix) : TDynFloatMatrix
 Procedure AddToMatrix( var X : TDynFloatMatrix; const Y : TDynFloatMatrix)
 Function SubtractMatrices( const X, Y : TDynFloatMatrix) : TDynFloatMatrix
 Procedure SubtractFromMatrix( var X : TDynFloatMatrix; const Y : TDynFloatMatrix)
 Function MultiplyMatrixByConst( const X : TDynFloatMatrix; const K : Extended) : TDynFloatMatrix
 Procedure MultiplyMatrixByConst2( var X : TDynFloatMatrix; const K : Extended);
 Function MultiplyMatrices( const X, Y : TDynFloatMatrix) : TDynFloatMatrix;
 Function TransposeMatrix( const X : TDynFloatMatrix) : TDynFloatMatrix;
 Function GrandMean( const X : TDynFloatMatrix; var N : LongWord) : Extended
end;

procedure SIRegister_ESBMaths(CL: TPSPascalCompiler);
begin
 'ESBMinSingle','Single').setExtended( 1.5e-45);
 'ESBMaxSingle','Single').setExtended( 3.4e+38);
 'ESBMinDouble','Double').setExtended( 5.0e-324);
 'ESBMaxDouble','Double').setExtended( 1.7e+308);
 'ESBMinExtended','Extended').setExtended( 3.6e-4951);
 'ESBMaxExtended','Extended').setExtended( 1.1e+4932);
 'ESBMinCurrency','Currency').SetExtended( - 922337203685477.5807);
 'ESBMaxCurrency','Currency').SetExtended( 922337203685477.5807);
 'ESBSqrt2','Extended').setExtended( 1.4142135623730950488);
 'ESBSqrt3','Extended').setExtended( 1.7320508075688772935);
 'ESBSqrt5','Extended').setExtended( 2.2360679774997896964);
 'ESBSqrt10','Extended').setExtended( 3.1622776601683793320);
```

```
'ESBSqrtPi','Extended').setExtended( 1.7724538509055160272 9);
'ESBCbrt2','Extended').setExtended( 1.2599210498948731648);
'ESBCbrt3','Extended').setExtended( 1.4422495703074083823);
'ESBCbrt10','Extended').setExtended( 2.1544346900318837219);
'ESBCbrt100','Extended').setExtended( 4.6415888336127788924);
'ESBCbrtPi','Extended').setExtended( 1.4645918875615232630);
'ESBInvSqrt2','Extended').setExtended( 0.70710678118654752440);
'ESBInvSqrt3','Extended').setExtended( 0.57735026918962576451);
'ESBInvSqrt5','Extended').setExtended( 0.44721359549995793928);
'ESBInvSqrtPi','Extended').setExtended( 0.56418958354775628695);
'ESBInvCbrtPi','Extended').setExtended( 0.68278406325529568147);
'ESBe','Extended').setExtended( 2.7182818284590452354);
'ESBe2','Extended').setExtended( 7.3890560989306502272);
'ESBePi','Extended').setExtended( 23.140692632779269006);
'ESBePiOn2','Extended').setExtended( 4.8104773809653516555);
'ESBePiOn4','Extended').setExtended( 2.1932800507380154566);
'ESBLn2','Extended').setExtended( 0.69314718055994530942);
'ESBLn10','Extended').setExtended( 2.30258509299404568402);
'ESBLnPi','Extended').setExtended( 1.14472988584940017414);
'ESBLog10Base2','Extended').setExtended( 3.3219280948873623478);
'ESBLog2Base10','Extended').setExtended( 0.30102999566398119521);
'ESBLog3Base10','Extended').setExtended( 0.47712125471966243730);
'ESBLogPiBase10','Extended').setExtended( 0.4971498726941339);
'ESBLogEBase10','Extended').setExtended( 0.43429448190325182765);
'ESBPi','Extended').setExtended( 3.1415926535897932385);
'ESBInvPi','Extended').setExtended( 3.1830988618379067154e-1);
'ESBTwoPi','Extended').setExtended( 6.2831853071795864769);
'ESBThreePi','Extended').setExtended( 9.4247779607693797153);
'ESBPi2','Extended').setExtended( 9.8696044010893586188);
'ESBPiToE','Extended').setExtended( 22.459157718361045473);
'ESBPiOn2','Extended').setExtended( 1.5707963267948966192);
'ESBPiOn3','Extended').setExtended( 1.0471975511965977462);
'ESBPiOn4','Extended').setExtended( 0.7853981633974483096);
'ESBThreePiOn2','Extended').setExtended( 4.7123889803846898577);
'ESBFourPiOn3','Extended').setExtended( 4.1887902047863909846);
'ESBTwoToPower63','Extended').setExtended( 9223372036854775808.0);
'ESBOneRadian','Extended').setExtended( 57.295779513082320877);
'ESBOneDegree','Extended').setExtended( 1.7453292519943295769E-2);
'ESBOneMinute','Extended').setExtended( 2.9088820866572159615E-4);
'ESBOneSecond','Extended').setExtended( 4.8481368110953599359E-6);
'ESBGamma','Extended').setExtended( 0.57721566490153286061);
'ESBLnRt2Pi','Extended').setExtended( 9.189385332046727E-1);
 //LongWord', 'Cardinal
 TBitList', 'Word
Function UMul( const Num1, Num2 : LongWord) : LongWord
Function UMulDiv2p32( const Num1, Num2 : LongWord) : LongWord
Function UMulDiv( const Num1, Num2, Divisor : LongWord) : LongWord
Function UMulMod( const Num1, Num2, Modulus : LongWord) : LongWord
Function SameFloat( const X1, X2 : Extended) : Boolean
Function FloatIsZero( const X : Extended) : Boolean
Function FloatIsPositive( const X : Extended) : Boolean
Function FloatIsNegative( const X : Extended) : Boolean
Procedure IncLim( var B : Byte; const Limit : Byte)
Procedure IncLimSI( var B : ShortInt; const Limit : ShortInt)
Procedure IncLimW( var B : Word; const Limit : Word)
Procedure IncLimI( var B : Integer; const Limit : Integer)
Procedure IncLimL( var B : LongInt; const Limit : LongInt)
Procedure DecLim( var B : Byte; const Limit : Byte)
Procedure DecLimSI( var B : ShortInt; const Limit : ShortInt)
Procedure DecLimW( var B : Word; const Limit : Word)
Procedure DecLimI( var B : Integer; const Limit : Integer)
Procedure DecLimL( var B : LongInt; const Limit : LongInt)
Function MaxB( const B1, B2 : Byte) : Byte
Function MinB( const B1, B2 : Byte) : Byte
Function MaxSI( const B1, B2 : ShortInt) : ShortInt
Function MinSI( const B1, B2 : ShortInt) : ShortInt
Function MaxW( const B1, B2 : Word) : Word
Function MinW( const B1, B2 : Word) : Word
Function esbMaxI( const B1, B2 : Integer) : Integer
Function esbMinI( const B1, B2 : Integer) : Integer
Function MaxL( const B1, B2 : LongInt) : LongInt
Function MinL( const B1, B2 : LongInt) : LongInt
Procedure SwapB( var B1, B2 : Byte)
Procedure SwapSI( var B1, B2 : ShortInt)
Procedure SwapW( var B1, B2 : Word)
Procedure SwapI( var B1, B2 : SmallInt)
Procedure SwapL( var B1, B2 : LongInt)
Procedure SwapI32( var B1, B2 : Integer)
Procedure SwapC( var B1, B2 : LongWord)
Procedure SwapInt64( var X, Y : Int64)
Function esbSign( const B : LongInt) : ShortInt
Function Max4Word( const X1, X2, X3, X4 : Word) : Word
Function Min4Word( const X1, X2, X3, X4 : Word) : Word
Function Max3Word( const X1, X2, X3 : Word) : Word
Function Min3Word( const X1, X2, X3 : Word) : Word
Function MaxBArray( const B : array of Byte) : Byte
Function MaxWArray( const B : array of Word) : Word
Function MaxSIArray( const B : array of ShortInt) : ShortInt
Function MaxIArray( const B : array of Integer) : Integer
Function MaxLArray( const B : array of LongInt) : LongInt
Function MinBArray( const B : array of Byte) : Byte
Function MinWArray( const B : array of Word) : Word
Function MinSIArray( const B : array of ShortInt) : ShortInt
Function MinIArray( const B : array of Integer) : Integer
Function MinLArray( const B : array of LongInt) : LongInt
Function SumBArray( const B : array of Byte) : Byte
Function SumBArray2( const B : array of Byte) : Word
Function SumSIArray( const B : array of ShortInt) : ShortInt
Function SumSIArray2( const B : array of ShortInt) : Integer
Function SumWArray( const B : array of Word) : Word
Function SumWArray2( const B : array of Word) : LongInt
Function SumIArray( const B : array of Integer) : Integer
Function SumLArray( const B : array of LongInt) : LongInt
Function SumLWArray( const B : array of LongWord) : LongWord
Function ESBDigits( const X : LongWord) : Byte
Function BitsHighest( const X : LongWord) : Integer
Function ESBBitsNeeded( const X : LongWord) : Integer
Function esbGCD( const X, Y : LongWord) : LongWord
Function esbLCM( const X, Y : LongInt) : Int64
//Function esbLCM( const X, Y : LongInt) : LongInt
Function RelativePrime( const X, Y : LongWord) : Boolean
Function Get87ControlWord : TBitList
Procedure Set87ControlWord( const CWord : TBitList)
Procedure SwapExt( var X, Y : Extended)
Procedure SwapDbl( var X, Y : Double)
Procedure SwapSing( var X, Y : Single)
Function esbSgn( const X : Extended) : ShortInt
Function Distance( const X1, Y1, X2, Y2 : Extended) : Extended
Function ExtMod( const X, Y : Extended) : Extended
Function ExtRem( const X, Y : Extended) : Extended
Function CompMOD( const X, Y : Comp) : Comp
Procedure Polar2XY( const Rho, Theta : Extended; var X, Y : Extended)
Procedure XY2Polar( const X, Y : Extended; var Rho, Theta : Extended)
Function DMS2Extended( const Degs, Mins, Secs : Extended) : Extended
Procedure Extended2DMS( const X : Extended; var Degs, Mins, Secs : Extended)
Function MaxExt( const X, Y : Extended) : Extended
Function MinExt( const X, Y : Extended) : Extended
Function MaxEArray( const B : array of Extended) : Extended
Function MinEArray( const B : array of Extended) : Extended
Function MaxSArray( const B : array of Single) : Single
Function MinSArray( const B : array of Single) : Single
Function MaxCArray( const B : array of Comp) : Comp
Function MinCArray( const B : array of Comp) : Comp
Function SumSArray( const B : array of Single) : Single
Function SumEArray( const B : array of Extended) : Extended
Function SumSqEArray( const B : array of Extended) : Extended
Function SumSqDiffEArray( const B : array of Extended; Diff : Extended) : Extended
Function SumXYEArray( const X, Y : array of Extended) : Extended
Function SumCArray( const B : array of Comp) : Comp
Function FactorialX( A : LongWord) : Extended
Function PermutationX( N, R : LongWord) : Extended
Function esbBinomialCoeff( N, R : LongWord) : Extended
Function IsPositiveEArray( const X : array of Extended) : Boolean
Function esbGeometricMean( const X : array of Extended) : Extended
Function esbHarmonicMean( const X : array of Extended) : Extended
Function ESBMean( const X : array of Extended) : Extended
Function esbSampleVariance( const X : array of Extended) : Extended
Function esbPopulationVariance( const X : array of Extended) : Extended
Procedure esbSampleVarianceAndMean( const X : array of Extended; var Variance, Mean : Extended)
Procedure esbPopulationVarianceAndMean( const X : array of Extended; var Variance, Mean : Extended)
Function GetMedian( const SortedX : array of Extended) : Extended
Function GetMode( const SortedX : array of Extended; var Mode : Extended) : Boolean
Procedure GetQuartiles( const SortedX : array of Extended; var Q1, Q3 : Extended)
Function ESBMagnitude( const X : Extended) : Integer
Function ESBTan( Angle : Extended) : Extended
Function ESBCot( Angle : Extended) : Extended
Function ESBCosec( const Angle : Extended) : Extended
Function ESBSec( const Angle : Extended) : Extended
Function ESBArcTan( X, Y : Extended) : Extended
Procedure ESBSinCos( Angle : Extended; var SinX, CosX : Extended)
Function ESBArcCos( const X : Extended) : Extended
Function ESBArcSin( const X : Extended) : Extended
Function ESBArcSec( const X : Extended) : Extended
Function ESBArcCosec( const X : Extended) : Extended
Function ESBLog10( const X : Extended) : Extended
Function ESBLog2( const X : Extended) : Extended
Function ESBLogBase( const X, Base : Extended) : Extended
Function Pow2( const X : Extended) : Extended
Function IntPow( const Base : Extended; const Exponent : LongWord) : Extended
Function ESBIntPower( const X : Extended; const N : LongInt) : Extended
Function XtoY( const X, Y : Extended) : Extended
Function esbTenToY( const Y : Extended) : Extended
Function esbTwoToY( const Y : Extended) : Extended
Function LogXtoBaseY( const X, Y : Extended) : Extended
Function esbISqrt( const I : LongWord) : Longword
Function ILog2( const I : LongWord) : LongWord
Function IGreatestPowerOf2( const N : LongWord) : LongWord
Function ESBArCosh( X : Extended) : Extended
```

```
 Function ESBArSinh( X : Extended) : Extended
 Function ESBArTanh( X : Extended) : Extended
 Function ESBCosh( X : Extended) : Extended
 Function ESBSinh( X : Extended) : Extended
 Function ESBTanh( X : Extended) : Extended
 Function InverseGamma( const X : Extended) : Extended
 Function esbGamma( const X : Extended) : Extended
 Function esbLnGamma( const X : Extended) : Extended
 Function esbBeta( const X, Y : Extended) : Extended
 Function IncompleteBeta( X : Extended; P, Q : Extended) : Extended
end;

********************************Integer Huge Cardinal Utils***************************
Function Add_uint64_WithCarry( x, y : uint64; var Carry : Boolean) : uint64
Function Add_uint32_WithCarry( x, y : uint32; var Carry : Boolean) : uint32
Function Subtract_uint64_WithBorrow( x, y : uint64; var Borrow : Boolean) : uint64
Function Subtract_uint32_WithBorrow( x, y : uint32; var Borrow : Boolean) : uint32
Function BitCount_8( Value : byte) : integer
Function BitCount_16( Value : uint16) : integer
Function BitCount_32( Value : uint32) : integer
Function BitCount_64( Value : uint64) : integer
Function CountSetBits_64( Value : uint64) : integer TPrimalityTestNoticeProc',
 Procedure ( CountPrimalityTests : integer)
 Function gcd( a, b : THugeCardinal) : THugeCardinal
 Function lcm( a, b : THugeCardinal) : THugeCardinal
 Function isCoPrime( a, b : THugeCardinal) : boolean
 Function isProbablyPrime(p: THugeCardinal;OnProgress: TProgress; var wasAborted: boolean): boolean
 Function hasSmallFactor( p : THugeCardinal) : boolean
 //Function GeneratePrime( NumBits : integer; OnProgress : TProgress; OnPrimalityTest:
TPrimalityTestNoticeProc; PassCount: integer; Pool1:TMemoryStreamPool;var Prime: THugeCardinal; var
NumbersTested: integer) : boolean
 Function Inverse( Prime, Modulus : THugeCardinal) : THugeCardinal
 Const('StandardExponent','LongInt'( 65537);
//Procedure Compute_RSA_Fundamentals_2Factors(RequiredBitLengthOfN:integer;Fixed_e:uint64;var N,e,d,
Totient:TProgress;OnPrimalityTest:TPrimalityTestNoticeProc;GeneratePrimePassCount:int;Pool1:TMemoryStreamPo
Numbers
 Function Validate_RSA_Fundamentals( var N, e, d, Totient : THugeCardinal) : boolean')

procedure SIRegister_xrtl_math_Integer(CL: TPSPascalCompiler);
begin
  AddTypeS('TXRTLInteger', 'array of Integer
  AddClassN(FindClass('TOBJECT'),'EXRTLMathException
  (FindClass('TOBJECT'),'EXRTLExtendInvalidArgument
  AddClassN(FindClass('TOBJECT'),'EXRTLDivisionByZero
  AddClassN(FindClass('TOBJECT'),'EXRTLExpInvalidArgument
  AddClassN(FindClass('TOBJECT'),'EXRTLInvalidRadix
  AddClassN(FindClass('TOBJECT'),'EXRTLInvalidRadixDigit
  AddClassN(FindClass('TOBJECT'),'EXRTLRootInvalidArgument
  'BitsPerByte','LongInt'( 8);
  BitsPerDigit','LongInt'( 32);
  SignBitMask','LongWord( $80000000);
 Function XRTLAdjustBits( const ABits : Integer) : Integer
 Function XRTLLength( const AInteger : TXRTLInteger) : Integer
 Function XRTLDataBits( const AInteger : TXRTLInteger) : Integer
 Procedure XRTLBitPosition( const BitIndex : Integer; var Index, Mask : Integer)
 Procedure XRTLBitSet( var AInteger : TXRTLInteger; const BitIndex : Integer)
 Procedure XRTLBitReset( var AInteger : TXRTLInteger; const BitIndex : Integer)
 Function XRTLBitGet( const AInteger : TXRTLInteger; const BitIndex : Integer) : Integer
 Function XRTLBitGetBool( const AInteger : TXRTLInteger; const BitIndex : Integer) : Boolean
 Function XRTLExtend(const AInteger:TXRTLInteger;ADataBits:Integer;Sign:Int;var AResult:TXRTLInteger):Int;
 Function XRTLZeroExtend(const AInteger:TXRTLInteger;ADataBits:Integer; var AResult:TXRTLInteger):Integer;
 Function XRTLSignExtend(const AInteger:TXRTLInteger; ADataBits:Integer;var AResult:TXRTLInteger):Integer;
 Function XRTLSignStrip(const AInteger:TXRTLInteger;var AResult:TXRTLInteger;const AMinDataBits:Int):Int;
 Procedure XRTLNot( const AInteger : TXRTLInteger; var AResult : TXRTLInteger)
 Procedure XRTLOr( const AInteger1, AInteger2 : TXRTLInteger; var AResult : TXRTLInteger)
 Procedure XRTLAnd( const AInteger1, AInteger2 : TXRTLInteger; var AResult : TXRTLInteger)
 Procedure XRTLXor( const AInteger1, AInteger2 : TXRTLInteger; var AResult : TXRTLInteger)
 Function XRTLSign( const AInteger : TXRTLInteger) : Integer
 Procedure XRTLZero( var AInteger : TXRTLInteger)
 Procedure XRTLOne( var AInteger : TXRTLInteger)
 Procedure XRTLMOne( var AInteger : TXRTLInteger)
 Procedure XRTLTwo( var AInteger : TXRTLInteger)
 Function XRTLNeg( const AInteger : TXRTLInteger; var AResult : TXRTLInteger) : Integer
 Function XRTLAbs( const AInteger : TXRTLInteger; var AResult : TXRTLInteger) : Integer
 Procedure XRTLFullSum( const A, B, C : Integer; var Sum, Carry : Integer)
 Function XRTLAdd( const AInteger1, AInteger2 : TXRTLInteger; var AResult : TXRTLInteger) : Integer;
 Function XRTLAdd1(const AInteger1:TXRTLInteger;const AInteger2:Int64;var AResult:TXRTLInteger):Integer;
 Function XRTLSub( const AInteger1, AInteger2 : TXRTLInteger; var AResult : TXRTLInteger) : Integer;
 Function XRTLSub1( const AInteger1:TXRTLInteger;const AInteger2:Int64;var AResult:TXRTLInteger):Integer;
 Function XRTLCompare( const AInteger1, AInteger2 : TXRTLInteger) : Integer;
 Function XRTLCompare1( const AInteger1 : TXRTLInteger; const AInteger2 : Int64) : Integer;
 Function XRTLUMul( const AInteger1, AInteger2 : TXRTLInteger; var AResult : TXRTLInteger) : Integer
 Function XRTLMulAdd(const AInteger1,AInteger2,AInteger3:TXRTLInteger; var AResult:TXRTLInteger):Integer
 Function XRTLMul( const AInteger1, AInteger2 : TXRTLInteger; var AResult : TXRTLInteger) : Integer
 Procedure XRTLDivMod( const AInteger1, AInteger2 : TXRTLInteger; var QResult, RResult : TXRTLInteger)
 Procedure XRTLSqr( const AInteger : TXRTLInteger; var AResult : TXRTLInteger)
 Procedure XRTLSqrt( const AInteger : TXRTLInteger; var AResult : TXRTLInteger)
 Procedure XRTLRoot( const AInteger1, AInteger2 : TXRTLInteger; var AResult : TXRTLInteger)
 Procedure XRTLRootApprox(const AInteger1,AInteger2:TXRTLInteger;var ALowApproxResult,
AHighApproxResult:TXRTLInteger)
```

```
 Procedure XRTLURootApprox(const AInteger1,AInteger2:TXRTLInteger;var ALowApproxResult, AHighApproxResult:TXRTLInteger);
 Procedure XRTLExp( const AInteger1, AInteger2 : TXRTLInteger; var AResult : TXRTLInteger)
 Procedure XRTLExpMod( const AInteger1, AInteger2, AInteger3 : TXRTLInteger;var AResult: TXRTLInteger)
 Procedure XRTLSLBL(const AInteger: TXRTLInteger; const BitCount:Integer; var AResult: TXRTLInteger)
 Procedure XRTLSABL(const AInteger: TXRTLInteger; const BitCount:Integer; var AResult: TXRTLInteger)
 Procedure XRTLRCBL(const AInteger: TXRTLInteger; const BitCount:Integer; var AResult: TXRTLInteger)
 Procedure XRTLSLDL(const AInteger:TXRTLInteger;const DigitCount:Integer; var AResult:TXRTLInteger)
 Procedure XRTLSADL(const AInteger: TXRTLInteger; const DigitCount:Integer;var AResult: TXRTLInteger)
 Procedure XRTLRCDL(const AInteger:TXRTLInteger; const DigitCount:Integer; var AResult: TXRTLInteger)
 Procedure XRTLSLBR(const AInteger: TXRTLInteger; const BitCount:Integer; var AResult: TXRTLInteger)
 Procedure XRTLSABR(const AInteger: TXRTLInteger; const BitCount:Integer; var AResult: TXRTLInteger)
 Procedure XRTLRCBR(const AInteger: TXRTLInteger; const BitCount:Integer; var AResult: TXRTLInteger)
 Procedure XRTLSLDR(const AInteger:TXRTLInteger; const DigitCount:Integer; var AResult: TXRTLInteger)
 Procedure XRTLSADR(const AInteger: TXRTLInteger; const DigitCount:Integer;var AResult: TXRTLInteger)
 Procedure XRTLRCDR(const AInteger: TXRTLInteger;const DigitCount:Integer;var AResult: TXRTLInteger)
 Function XRTLToHex( const AInteger : TXRTLInteger; Digits : Integer) : string
 Function XRTLToBin( const AInteger : TXRTLInteger; Digits : Integer) : string
 Function XRTLToString( const AInteger : TXRTLInteger; Radix : Integer; Digits : Integer) : string
 Procedure XRTLFromHex( const Value : string; var AResult : TXRTLInteger)
 Procedure XRTLFromBin( const Value : string; var AResult : TXRTLInteger)
 Procedure XRTLFromString( const Value : string; var AResult : TXRTLInteger; Radix : Integer)
 Procedure XRTLAssign( const AInteger : TXRTLInteger; var AResult : TXRTLInteger);
 Procedure XRTLAssign1( const Value : Integer; var AResult : TXRTLInteger);
 Procedure XRTLAssign2( const Value : Int64; var AResult : TXRTLInteger);
 Procedure XRTLAppend( const ALow, AHigh : TXRTLInteger; var AResult : TXRTLInteger)
 Procedure XRTLSplit(const AInteger: TXRTLInteger; var ALow,AHigh: TXRTLInteger;LowDigits: Integer)
 Function XRTLGetMSBitIndex( const AInteger : TXRTLInteger) : Integer
 Procedure XRTLMinMax(const AInteger1, AInteger2 : TXRTLInteger;var AMinResult,AMaxResult: TXRTLInteger)
 Procedure XRTLMin( const AInteger1, AInteger2 : TXRTLInteger; var AResult : TXRTLInteger);
 Procedure XRTLMin1(const AInteger1: TXRTLInteger;const AInteger2:Integer;var AResult : TXRTLInteger);
 Procedure XRTLMax( const AInteger1, AInteger2 : TXRTLInteger; var AResult : TXRTLInteger);
 Procedure XRTLMax1(const AInteger1:TXRTLInteger; const AInteger2:Integer; var AResult:TXRTLInteger);
 Procedure XRTLGCD( const AInteger1, AInteger2 : TXRTLInteger; var AResult : TXRTLInteger)
 Procedure XRTLSwap( var AInteger1, AInteger2 : TXRTLInteger)
 Procedure XRTLFactorial( const AInteger : TXRTLInteger; var AResult : TXRTLInteger)
 Procedure XRTLFactorialMod( const AInteger1, AInteger2 : TXRTLInteger; var AResult : TXRTLInteger)
end;

procedure SIRegister_JvXPCoreUtils(CL: TPSPascalCompiler);
begin
 Function JvXPMethodsEqual( const Method1, Method2 : TMethod) : Boolean
 Procedure JvXPDrawLine( const ACanvas : TCanvas; const X1, Y1, X2, Y2 : Integer)
 Procedure JvXPCreateGradientRect( const AWidth, AHeight : Integer; const StartColor, EndColor : TColor; const Colors:TJvXPGradientColors;const Style:TJvXPGradientStyle;const Dithered:Boolean;var Bitmap:TBitmap);
 Procedure JvXPAdjustBoundRect(const BorderWidth:Byte; const ShowBoundLines:Boolean; const BoundLines:TJvXPBoundLines; var Rect : TRect)
 Procedure JvXPDrawBoundLines(const ACan:TCanvas;const BoundLines:TJvXPBoundLines;const AColor:TColor;const Rect:TRect);
 Procedure JvXPConvertToGray2( Bitmap : TBitmap)
 Procedure JvXPRenderText( const AParent : TControl; const ACanvas : TCanvas; ACaption : TCaption; const AFont : TFont; const AEnabled, AShowAccelChar : Boolean; var ARect : TRect; AFlags : Integer)
 Procedure JvXPFrame3D(const ACanvas:TCanvas;const Rect:TRect;const TopColor,BottomColor:TColor;const Swapped:Boolean);
 Procedure JvXPColorizeBitmap( Bitmap : TBitmap; const AColor : TColor)
 Procedure JvXPSetDrawFlags(const AAlignment: TAlignment;const AWordWrap: Boolean; var Flags : Integer)
 Procedure JvXPPlaceText( const AParent: TControl; const ACanvas : TCanvas; const AText : TCaption; const AFont : TFont; const AEnabled,AShowAccelChar:Boolean;const AAlignment:TAlignment;const AWordWrap:Boolean;var Rect:TRect)
end;


procedure SIRegister_uwinstr(CL: TPSPascalCompiler);
begin
 Function StrDec( S : String) : String
 Function uIsNumeric( var S : String; var X : Float) : Boolean
 Function ReadNumFromEdit( Edit : TEdit) : Float
 Procedure WriteNumToFile( var F : Text; X : Float)
end;

procedure SIRegister_utexplot(CL: TPSPascalCompiler);
begin
 Function TeX_InitGraphics( FileName : String; PgWidth, PgHeight : Integer; Header : Boolean) : Boolean
 Procedure TeX_SetWindow( X1, X2, Y1, Y2 : Integer; GraphBorder : Boolean)
 Procedure TeX_LeaveGraphics( Footer : Boolean)
 Procedure TeX_SetOxScale( Scale : TScale; OxMin, OxMax, OxStep : Float)
 Procedure TeX_SetOyScale( Scale : TScale; OyMin, OyMax, OyStep : Float)
 Procedure TeX_SetGraphTitle( Title : String)
 Procedure TeX_SetOxTitle( Title : String)
 Procedure TeX_SetOyTitle( Title : String)
 Procedure TeX_PlotOxAxis
 Procedure TeX_PlotOyAxis
 Procedure TeX_PlotGrid( Grid : TGrid)
 Procedure TeX_WriteGraphTitle
 Function TeX_SetMaxCurv( NCurv : Byte) : Boolean
 Procedure TeX_SetPointParam( CurvIndex, Symbol, Size : Integer)
 Procedure TeX_SetLineParam( CurvIndex, Style : Integer; Width : Float; Smooth : Boolean)
 Procedure TeX_SetCurvLegend( CurvIndex : Integer; Legend : String)
 Procedure TeX_SetCurvStep( CurvIndex, Step : Integer)
 Procedure TeX_PlotCurve( X, Y : TVector; Lb, Ub, CurvIndex : Integer)
 Procedure TeX_PlotCurveWithErrorBars( X, Y, S : TVector; Ns, Lb, Ub, CurvIndex : Integer)
 Procedure TeX_PlotFunc( Func : TFunc; X1, X2 : Float; Npt : Integer; CurvIndex : Integer)
 Procedure TeX_WriteLegend( NCurv : Integer; ShowPoints, ShowLines : Boolean)
 Procedure TeX_ConRec( Nx, Ny, Nc : Integer; X, Y, Z : TVector; F : TMatrix)
 Function Xcm( X : Float) : Float
 Function Ycm( Y : Float) : Float
end;

*----------------------------------------------------------------------------*)
procedure SIRegister_VarRecUtils(CL: TPSPascalCompiler);
begin
  TConstArray', 'array of TVarRec
 Function CopyVarRec( const Item : TVarRec) : TVarRec
 Function CreateConstArray( const Elements : array of const) : TConstArray
 Procedure FinalizeVarRec( var Item : TVarRec)
 Procedure FinalizeConstArray( var Arr : TConstArray)
end;

procedure SIRegister_StStrS(CL: TPSPascalCompiler);
begin
 Function HexBS( B : Byte) : ShortString
 Function HexWS( W : Word) : ShortString
 Function HexLS( L : LongInt) : ShortString
 Function HexPtrS( P : Pointer) : ShortString
 Function BinaryBS( B : Byte) : ShortString
 Function BinaryWS( W : Word) : ShortString
 Function BinaryLS( L : LongInt) : ShortString
 Function OctalBS( B : Byte) : ShortString
 Function OctalWS( W : Word) : ShortString
 Function OctalLS( L : LongInt) : ShortString
 Function Str2Int16S( const S : ShortString; var I : SmallInt) : Boolean
 Function Str2WordS( const S : ShortString; var I : Word) : Boolean
 Function Str2LongS( const S : ShortString; var I : LongInt) : Boolean
 Function Str2RealS( const S : ShortString; var R : Double) : Boolean
 Function Str2RealS( const S : ShortString; var R : Real) : Boolean
 Function Str2ExtS( const S : ShortString; var R : Extended) : Boolean
 Function Long2StrS( L : LongInt) : ShortString
 Function Real2StrS( R : Double; Width : Byte; Places : ShortInt) : ShortString
 Function Ext2StrS( R : Extended; Width : Byte; Places : ShortInt) : ShortString
 Function ValPrepS( const S : ShortString) : ShortString
 Function CharStrS( C : AnsiChar; Len : Cardinal) : ShortString
 Function PadChS( const S : ShortString; C : AnsiChar; Len : Cardinal) : ShortString
 Function PadS( const S : ShortString; Len : Cardinal) : ShortString
 Function LeftPadChS( const S : ShortString; C : AnsiChar; Len : Cardinal) : ShortString
 Function LeftPadS( const S : ShortString; Len : Cardinal) : ShortString
 Function TrimLeadS( const S : ShortString) : ShortString
 Function TrimTrailS( const S : ShortString) : ShortString
 Function TrimS( const S : ShortString) : ShortString
 Function TrimSpacesS( const S : ShortString) : ShortString
 Function CenterChS( const S : ShortString; C : AnsiChar; Len : Cardinal) : ShortString
 Function CenterS( const S : ShortString; Len : Cardinal) : ShortString
 Function EntabS( const S : ShortString; TabSize : Byte) : ShortString
 Function DetabS( const S : ShortString; TabSize : Byte) : ShortString
 Function ScrambleS( const S, Key : ShortString) : ShortString
 Function SubstituteS( const S, FromStr, ToStr : ShortString) : ShortString
 Function FilterS( const S, Filters : ShortString) : ShortString
 Function CharExistsS( const S : ShortString; C : AnsiChar) : Boolean
 Function CharCountS( const S : ShortString; C : AnsiChar) : Byte
 Function WordCountS( const S, WordDelims : ShortString) : Cardinal
 Function WordPositionS( N : Cardinal; const S, WordDelims : ShortString; var Pos : Cardinal) : Boolean
 Function ExtractWordS( N : Cardinal; const S, WordDelims : ShortString) : ShortString
 Function AsciiCountS( const S, WordDelims : ShortString; Quote : AnsiChar) : Cardinal
 Function AsciiPositionS(N:Cardinal;const S,WordDelims:ShortString;Quote:AnsiChar;var Pos:Cardinal):Boolean
 Function ExtractAsciiS(N:Cardinal;const S,WordDelims:ShortString;Quote:AnsiChar): ShortString
 Procedure WordWrapS(const InSt: ShortString; var OutSt,Overlap: ShortString;
Margin:Cardinal;PadToMargin:Boolean)
 Function CompStringS( const S1, S2 : ShortString) : Integer
 Function CompUCStringS( const S1, S2 : ShortString) : Integer
 Function SoundexS( const S : ShortString) : ShortString
 Function MakeLetterSetS( const S : ShortString) : Longint
 Procedure BMMakeTableS( const MatchString : ShortString; var BT : BTable)
 Function BMSearchS(var Buffer,BufLength:Card;var BT:BTable;const MatchString:ShortString;var
Pos:Cardinal):Bool;
 Function BMSearchUCS(var Buffer,BufLength:Card;var BT:BTable;const MatchString:ShortString;var
Pos:Cardinal):Bool;
 Function DefaultExtensionS( const Name, Ext : ShortString) : ShortString
 Function ForceExtensionS( const Name, Ext : ShortString) : ShortString
 Function JustFilenameS( const PathName : ShortString) : ShortString
 Function JustNameS( const PathName : ShortString) : ShortString
 Function JustExtensionS( const Name : ShortString) : ShortString
 Function JustPathnameS( const PathName : ShortString) : ShortString
 Function AddBackSlashS( const DirName : ShortString) : ShortString
 Function CleanPathNameS( const PathName : ShortString) : ShortString
 Function HasExtensionS( const Name : ShortString; var DotPos : Cardinal) : Boolean
 Function CommaizeS( L : LongInt) : ShortString
 Function CommaizeChS( L : Longint; Ch : AnsiChar) : ShortString
 Function FloatFormS(const Mask:ShortString;R:TstFloat;const LtCurr,RtCurr:ShortString;Sep,
DecPt:Char):ShortString;
```

```
 Function LongIntFormS(const Mask:ShortString;L:LongInt;const LtCurr, RtCurr:ShortString;Sep:AnsiChar):ShortString;
 Function StrChPosS( const P : ShortString; C : AnsiChar; var Pos : Cardinal) : Boolean
 Function StrStPosS( const P, S : ShortString; var Pos : Cardinal) : Boolean
 Function StrStCopyS( const S : ShortString; Pos, Count : Cardinal) : ShortString
 Function StrChInsertS( const S : ShortString; C : AnsiChar; Pos : Cardinal) : ShortString
 Function StrStInsertS( const S1, S2 : ShortString; Pos : Cardinal) : ShortString
 Function StrChDeleteS( const S : ShortString; Pos : Cardinal) : ShortString
 Function StrStDeleteS( const S : ShortString; Pos, Count : Cardinal) : ShortString
 Function ContainsOnlyS( const S, Chars : ShortString; var BadPos : Cardinal) : Boolean
 Function ContainsOtherThanS( const S, Chars : ShortString; var BadPos : Cardinal) : Boolean
 Function CopyLeftS( const S : ShortString; Len : Cardinal) : ShortString
 Function CopyMidS( const S : ShortString; First, Len : Cardinal) : ShortString
 Function CopyRightS( const S : ShortString; First : Cardinal) : ShortString
 Function CopyRightAbsS( const S : ShortString; NumChars : Cardinal) : ShortString
 Function CopyFromNthWordS(const S,WordDelims:string;const AWord:String;N:Card;var SubString:ShortString):Bool;
 Function DeleteFromNthWordS(const S,WordDelims:String;AWord:ShortString;N:Card;var SubStr:ShortString):Bool;
 Function CopyFromToWordS(const S,WordDelims,Word1,Word2:ShortString;N1,N2:Card;var SubString:ShortString):Bool;
 Function DeleteFromToWordS(const S,WordDelims,Wrd1,Wrd2:ShortString;N1,N2:Card;var SubString:ShortString):Bool;
 Function CopyWithinS( const S, Delimiter : ShortString; Strip : Boolean) : ShortString
 Function DeleteWithinS( const S, Delimiter : ShortString) : ShortString
 Function ExtractTokensS(const S, Delims:ShortString;QuoteChar:AnsiChar;AllowNulls:Boolean;Tokens:TStrings):Cardinal
 Function IsChAlphaS( C : Char) : Boolean
 Function IsChNumericS( C : Char; const Numbers : ShortString) : Boolean
 Function IsChAlphaNumericS( C : Char; const Numbers : ShortString) : Boolean
 Function IsStrAlphaS( const S : Shortstring) : Boolean
 Function IsStrNumericS( const S, Numbers : ShortString) : Boolean
 Function IsStrAlphaNumericS( const S, Numbers : ShortString) : Boolean
 Function LastWordS( const S, WordDelims, AWord : ShortString; var Position : Cardinal) : Boolean
 Function LastWordAbsS( const S, WordDelims : ShortString; var Position : Cardinal) : Boolean
 Function LastStringS( const S, AString : ShortString; var Position : Cardinal) : Boolean
 Function LeftTrimCharsS( const S, Chars : ShortString) : ShortString
 Function KeepCharsS( const S, Chars : ShortString) : ShortString
 Function RepeatStringS(const RepeatString:ShortString;var Repetitions: Cardinal; MaxLen : Cardinal):ShortString;
 Function ReplaceStringS(const S,OldString,NewString:ShortString;N:Cardinal;var Replacements:Cardinal):ShortString;
 Function ReplaceStringAllS(const S,OldString,NewString:ShortString;var Replacements:Cardinal):ShortString;
 Function ReplaceWordS(const S,WordDelims,OldWord,NewW:SString;N:Cardinal;var Replacements:Cardinal):ShortString
 Function ReplaceWordAllS(const S,WordDelims,OldWord,NewWord:ShortString;var Replacements:Cardinal):ShortString
 Function RightTrimCharsS( const S, Chars : ShortString) : ShortString
 Function StrWithinS(const S,SearchStr: ShortString;Start:Cardinal;var Position:Cardinal):boolean
 Function TrimCharsS( const S, Chars : ShortString) : ShortString
 Function WordPosS(const S,WordDelims,AWord:ShortString;N:Cardinal; var Position: Cardinal):Boolean
end;


********unit uPSI_StUtils; from Systools4*********************************************************
Function SignL( L : LongInt) : Integer
Function SignF( F : Extended) : Integer
Function MinWord( A, B : Word) : Word
Function MidWord( W1, W2, W3 : Word) : Word
Function MaxWord( A, B : Word) : Word
Function MinLong( A, B : LongInt) : LongInt
Function MidLong( L1, L2, L3 : LongInt) : LongInt
Function MaxLong( A, B : LongInt) : LongInt
Function MinFloat( F1, F2 : Extended) : Extended
Function MidFloat( F1, F2, F3 : Extended) : Extended
Function MaxFloat( F1, F2 : Extended) : Extended
Function MakeInteger16( H, L : Byte) : SmallInt
Function MakeWordS( H, L : Byte) : Word
Function SwapNibble( B : Byte) : Byte
Function SwapWord( L : LongInt) : LongInt
Procedure SetFlag( var Flags : Word; FlagMask : Word)
Procedure ClearFlag( var Flags : Word; FlagMask : Word)
Function FlagIsSet( Flags, FlagMask : Word) : Boolean
Procedure SetByteFlag( var Flags : Byte; FlagMask : Byte)
Procedure ClearByteFlag( var Flags : Byte; FlagMask : Byte)
Function ByteFlagIsSet( Flags, FlagMask : Byte) : Boolean
Procedure SetLongFlag( var Flags : LongInt; FlagMask : LongInt)
Procedure ClearLongFlag( var Flags : LongInt; FlagMask : LongInt)
Function LongFlagIsSet( Flags, FlagMask : LongInt) : Boolean
Procedure ExchangeBytes( var I, J : Byte)
Procedure ExchangeWords( var I, J : Word)
Procedure ExchangeLongInts( var I, J : LongInt)
Procedure ExchangeStructs( var I, J, Size : Cardinal)
Procedure FillWord( var Dest, Count : Cardinal; Filler : Word)
Procedure FillStruct( var Dest, Count : Cardinal; var Filler, FillerSize : Cardinal)
Function AddWordToPtr( P : ___Pointer; W : Word) : ___Pointer
//******************uPSI_StFIN;*********************************************************
Function AccruedInterestMaturity(Issue,Maturity:TStDate;Rate,Par:Extended;Basis: TStBasis): Extended
Function AccruedInterestPeriodic(Issue,Settlement,Maturity:TStDate;Rate, Par:Extended;Frequency:TStFrequency; Basis : TStBasis) : Extended
Function BondDuration( Settlement,Maturity:TStDate;Rate, Yield:Ext;Frequency:TStFrequency;Basis:TStBasis):Extended;
Function BondPrice(Settlement,Maturity:TStDate;Rate,Yield, Redemption:Ext;Freq:TStFrequency;Basis:TStBasis): Extended
Function CumulativeInterest( Rate : Extended; NPeriods : Integer; PV : Extended; StartPeriod, EndPeriod : Integer; Frequency : TStFrequency; Timing : TStPaymentTime) : Extended
Function CumulativePrincipal( Rate : Extended; NPeriods : Integer; PV : Extended; StartPeriod, EndPeriod : Integer; Frequency : TStFrequency; Timing : TStPaymentTime) : Extended
Function DayCount( Day1, Day2 : TStDate; Basis : TStBasis) : LongInt
Function DecliningBalance( Cost, Salvage : Extended; Life, Period, Month : Integer) : Extended
Function DiscountRate(Settlement,Maturity:TStDate; Price,Redemption:Extended;Basis:TStBasis): Extended;
Function DollarToDecimal( FracDollar : Extended; Fraction : Integer) : Extended
Function DollarToDecimalText( DecDollar : Extended) : string
Function DollarToFraction( DecDollar : Extended; Fraction : Integer) : Extended
Function DollarToFractionStr( FracDollar : Extended; Fraction : Integer) : string
Function EffectiveInterestRate( NominalRate : Extended; Frequency : TStFrequency) : Extended
Function FutureValueS(Rate:Extended;NPeriods:Int;Pmt, PV:Extended;Freq:TStFrequency;Timing:TStPaymentTime):Extended;
Function FutureValueSchedule( Principal : Extended; const Schedule : array of Double) : Extended
Function FutureValueSchedule16( Principal : Extended; const Schedule, NRates : Integer) : Extended
Function InterestRateS(NPeriods:Int;Pmt,PV, FV:Extended;Freq:TStFrequency;Timing:TStPaymentTime;Guess:Extended):Extend;
Function InternalRateOfReturn( const Values : array of Double; Guess : Extended) : Extended
Function InternalRateOfReturn16( const Values, NValues : Integer; Guess : Extended) : Extended
Function IsCardValid( const S : string) : Boolean
Function ModifiedDuration(Settlement,Maturity:TStDate;Rate, Yield:Extended;Freq:TStFrequency;Basis:TStBasis):Extended;
Function ModifiedIRR(const Values: array of Double; FinanceRate,ReinvestRate: Extended) : Extended
Function ModifiedIRR16(const Values,NValues:Integer; FinanceRate,ReinvestRate: Extended) : Extended
Function NetPresentValueS( Rate : Extended; const Values : array of Double) : Extended
Function NetPresentValue16( Rate : Extended; const Values, NValues : Integer) : Extended
Function NominalInterestRate( EffectRate : Extended; Frequency : TStFrequency) : Extended
Function NonperiodicIRR(const Values:array of Double;const Dates:array of TStDate;Guess:Extended):Extended;
Function NonperiodicNPV(Rate:Extended;const Values: array of Double;const Dates:array of TStDate):Extended;
Function Payment( Rate : Extended; NPeriods : Integer; PV, FV : Extended; Frequency : TStFrequency; Timing : TStPaymentTime): Extended
Function Periods(Rate:Extended;Pmt,PV,FV:Extended;Frequency:TStFrequency;Timing:TStPaymentTime):Integer;
Function PresentValueS( Rate: Extended; NPeriods: Integer; Pmt, FV: Extended; Frequency: TStFrequency; Timing: TStPaymentTime): Extended
Function ReceivedAtMaturity(Settlement,Maturity:TStDate;Invest,Discount:Extended;Basis:TStBasis):Extended;
Function RoundToDecimal( Value : Extended; Places : Integer; Bankers : Boolean) : Extended
Function TBillEquivYield( Settlement, Maturity : TStDate; Discount : Extended) : Extended
Function TBillPrice( Settlement, Maturity : TStDate; Discount : Extended) : Extended
Function TBillYield( Settlement, Maturity : TStDate; Price : Extended) : Extended
Function VariableDecliningBalance( Cost, Salvage : Extended; Life : Integer; StartPeriod, EndPeriod, Factor : Extended; NoSwitch : boolean) : Extended
Function YieldDiscounted(Settlement,Maturity:TStDate;Price,Redemption:Extended;Basis:TStBasis):Extended;
Function YieldPeriodic(Settlement,Maturity:TStDate;Rate,Price, Redemption:Extended;Freq:TStFrequency;Basis:TStBasis):Extended
Function YieldMaturity(Issue,Settlement,Maturity:TStDate;Rate,Price:Extended;Basis:TStBasis):Extended;

//*******************************************unit uPSI_StAstroP;
Procedure PlanetsPos( JD : Double; var PA : TStPlanetsArray)
//*****unit unit uPSI_StStat; Statistic Package of SysTools*****************************************
Function AveDev( const Data : array of Double) : Double
Function AveDev16( const Data, NData : Integer) : Double
Function Confidence( Alpha, StandardDev : Double; Size : LongInt) : Double
Function Correlation( const Data1, Data2 : array of Double) : Double
Function Correlation16( const Data1, Data2, NData : Integer) : Double
Function Covariance( const Data1, Data2 : array of Double) : Double
Function Covariance16( const Data1, Data2, NData : Integer) : Double
Function DevSq( const Data : array of Double) : Double
Function DevSq16( const Data, NData : Integer) : Double
Procedure Frequency(const Data:array of Double;const Bins:array of Double;var Counts: array of LongInt);
 //Procedure Frequency16( const Data, NData : Integer; const Bins, NBins : Integer; var Counts)
Function GeometricMeanS( const Data : array of Double) : Double
Function GeometricMean16( const Data, NData : Integer) : Double
Function HarmonicMeanS( const Data : array of Double) : Double
Function HarmonicMean16( const Data, NData : Integer) : Double
Function Largest( const Data : array of Double; K : Integer) : Double
Function Largest16( const Data, NData : Integer; K : Integer) : Double
Function MedianS( const Data : array of Double) : Double
Function Median16( const Data, NData : Integer) : Double
Function Mode( const Data : array of Double) : Double
Function Mode16( const Data, NData : Integer) : Double
Function Percentile( const Data : array of Double; K : Double) : Double
Function Percentile16( const Data, NData : Integer; K : Double) : Double
Function PercentRank( const Data : array of Double; X : Double) : Double
Function PercentRank16( const Data, NData : Integer; X : Double) : Double
Function Permutations( Number, NumberChosen : Integer) : Extended
Function Combinations( Number, NumberChosen : Integer) : Extended
Function FactorialS( N : Integer) : Extended
Function Rank( Number : Double; const Data : array of Double; Ascending : Boolean) : Integer
Function Rank16( Number : Double; const Data, NData : Integer; Ascending : Boolean) : Integer
Function Smallest( const Data : array of Double; K : Integer) : Double
Function Smallest16( const Data, NData : Integer; K : Integer) : Double
Function TrimMean( const Data : array of Double; Percent : Double) : Double
Function TrimMean16( const Data, NData : Integer; Percent : Double) : Double
 AddTypeS('TStLinEst', 'record B0 : Double; B1 : double; seB0 : double; seB'
  +'1 : Double; R2 : Double; sigma :Double; SSr: double; SSe: Double; F0 : Double; df : Integer;end
Procedure LinEst(const KnownY:array of Double;const KnownX:array of Double;var
LF:TStLinEst;ErrorStats:Bool;
Procedure LogEst(const KnownY:array of Double;const KnownX:array of Double;var
LF:TStLinEst;ErrorStats:Bool;
Function Forecast(X: Double; const KnownY: array of Double; const KnownX: array of Double) : Double
Function ForecastExponential(X:Double;const KnownY:array of Double;const KnownX:array of Double):Double;
Function Intercept( const KnownY : array of Double; const KnownX : array of Double) : Double
Function RSquared( const KnownY : array of Double; const KnownX : array of Double) : Double
Function Slope( const KnownY : array of Double; const KnownX : array of Double) : Double
Function StandardErrorY( const KnownY : array of Double; const KnownX : array of Double) : Double
Function BetaDist( X, Alpha, Beta, A, B : Single) : Single
Function BetaInv( Probability, Alpha, Beta, A, B : Single) : Single
Function BinomDist( NumberS, Trials : Integer; ProbabilityS : Single; Cumulative : Boolean) : Single
Function CritBinom( Trials : Integer; ProbabilityS, Alpha : Single) : Integer
Function ChiDist( X : Single; DegreesFreedom : Integer) : Single
Function ChiInv( Probability : Single; DegreesFreedom : Integer) : Single
Function ExponDist( X, Lambda : Single; Cumulative : Boolean) : Single
Function FDist( X : Single; DegreesFreedom1, DegreesFreedom2 : Integer) : Single
Function FInv( Probability : Single; DegreesFreedom1, DegreesFreedom2 : Integer) : Single
Function LogNormDist( X, Mean, StandardDev : Single) : Single
Function LogInv( Probability, Mean, StandardDev : Single) : Single
Function NormDist( X, Mean, StandardDev : Single; Cumulative : Boolean) : Single
Function NormInv( Probability, Mean, StandardDev : Single) : Single
Function NormSDist( Z : Single) : Single
Function NormSInv( Probability : Single) : Single
Function Poisson( X : Integer; Mean : Single; Cumulative : Boolean) : Single
Function TDist( X : Single; DegreesFreedom : Integer; TwoTails : Boolean) : Single
Function TInv( Probability : Single; DegreesFreedom : Integer) : Single
Function Erfc( X : Single) : Single
Function GammaLn( X : Single) : Single
Function LargestSort( const Data : array of Double; K : Integer) : Double
Function SmallestSort( const Data : array of double; K : Integer) : Double

procedure SIRegister_TStSorter(CL: TPSPascalCompiler);
 Function OptimumHeapToUse( RecLen : Cardinal; NumRecs : LongInt) : LongInt
 Function MinimumHeapToUse( RecLen : Cardinal) : LongInt
 Function MergeInfo( MaxHeap : LongInt; RecLen : Cardinal; NumRecs : LongInt) : TMergeInfo
 Function DefaultMergeName( MergeNum : Integer) : string
 Procedure ArraySort( var A, RecLen, NumRecs : Cardinal; Compare : TUntypedCompareFunc)

procedure SIRegister_StAstro(CL: TPSPascalCompiler);
Function AmountOfSunlight( LD : TStDate; Longitude, Latitude : Double) : TStTime
 Function FixedRiseSet( LD : TStDate; RA, DC, Longitude, Latitude : Double) : TStRiseSetRec
 Function SunPos( UT : TStDateTimeRec) : TStPosRec
 Function SunPosPrim( UT : TStDateTimeRec) : TStSunXYZRec
 Function SunRiseSet( LD : TStDate; Longitude, Latitude : Double) : TStRiseSetRec
 Function Twilight( LD : TStDate; Longitude, Latitude : Double; TwiType:TStTwilight):TStRiseSetRec
 Function LunarPhase( UT : TStDateTimeRec) : Double
 Function MoonPos( UT : TStDateTimeRec) : TStMoonPosRec
 Function MoonRiseSet( LD : TStDate; Longitude, Latitude : Double) : TStRiseSetRec
 Function FirstQuarter( D : TStDate) : TStLunarRecord
 Function FullMoon( D : TStDate) : TStLunarRecord
 Function LastQuarter( D : TStDate) : TStLunarRecord
 Function NewMoon( D : TStDate) : TStLunarRecord
 Function NextFirstQuarter( D : TStDate) : TStDateTimeRec
 Function NextFullMoon( D : TStDate) : TStDateTimeRec
 Function NextLastQuarter( D : TStDate) : TStDateTimeRec
 Function NextNewMoon( D : TStDate) : TStDateTimeRec
 Function PrevFirstQuarter( D : TStDate) : TStDateTimeRec
 Function PrevFullMoon( D : TStDate) : TStDateTimeRec
 Function PrevLastQuarter( D : TStDate) : TStDateTimeRec
 Function PrevNewMoon( D : TStDate) : TStDateTimeRec
 Function SiderealTime( UT : TStDateTimeRec) : Double
 Function Solstice( Y, Epoch : Integer; Summer : Boolean) : TStDateTimeRec
 Function Equinox( Y, Epoch : Integer; Vernal : Boolean) : TStDateTimeRec
 Function SEaster( Y, Epoch : Integer) : TStDate
 Function DateTimeToAJD( D : TDateTime) : Double
 Function HoursMin( RA : Double) : ShortString
 Function DegsMin( DC : Double) : ShortString
 Function AJDToDateTime( D : Double) : TDateTime

Procedure SIRegister_StDate(CL: TPSPascalCompiler);
 Function CurrentDate : TStDate
 Function StValidDate( Day, Month, Year, Epoch : Integer) : Boolean
 Function DMYtoStDate( Day, Month, Year, Epoch : Integer) : TStDate
 Procedure StDateToDMY( Julian : TStDate; var Day, Month, Year : Integer)
 Function StIncDate( Julian : TStDate; Days, Months, Years : Integer) : TStDate
 Function IncDateTrunc( Julian : TStDate; Months, Years : Integer) : TStDate
 Procedure StDateDiff( Date1, Date2 : TStDate; var Days, Months, Years : Integer)
 Function BondDateDiff( Date1, Date2 : TStDate; DayBasis : TStBondDateType) : TStDate
 Function WeekOfYear( Julian : TStDate) : Byte
 Function AstJulianDate( Julian : TStDate) : Double
 Function AstJulianDatetoStDate( AstJulian : Double; Truncate : Boolean) : TStDate
 Function AstJulianDatePrim( Year, Month, Date : Integer; UT : TStTime) : Double
 Function StDayOfWeek( Julian : TStDate) : TStDayType
 Function DayOfWeekDMY( Day, Month, Year, Epoch : Integer) : TStDayType
 Function StIsLeapYear( Year : Integer) : Boolean
 Function StDaysInMonth( Month : Integer; Year, Epoch : Integer) : Integer
 Function ResolveEpoch( Year, Epoch : Integer) : Integer
 Function ValidTime( Hours, Minutes, Seconds : Integer) : Boolean
 Procedure StTimeToHMS( T : TStTime; var Hours, Minutes, Seconds : Byte)
 Function HMStoStTime( Hours, Minutes, Seconds : Byte) : TStTime
 Function CurrentTime : TStTime
 Procedure TimeDiff( Time1, Time2 : TStTime; var Hours, Minutes, Seconds : Byte)
 Function StIncTime( T : TStTime; Hours, Minutes, Seconds : Byte) : TStTime
 Function DecTime( T : TStTime; Hours, Minutes, Seconds : Byte) : TStTime
 Function RoundToNearestHour( T : TStTime; Truncate : Boolean) : TStTime
 Function RoundToNearestMinute( const T : TStTime; Truncate : Boolean) : TStTime
 Procedure DateTimeDiff(const DT1:TStDateTimeRec;var DT2:TStDateTimeRec;var Days:LongInt;var Secs:LongInt
 Procedure IncDateTime(const DT1:TStDateTimeRec;var DT2:TStDateTimeRec;Days:Integer;Secs:LongInt)
 Function DateTimeToStDate( DT : TDateTime) : TStDate
 Function DateTimeToStTime( DT : TDateTime) : TStTime
 Function StDateToDateTime( D : TStDate) : TDateTime
 Function StTimeToDateTime( T : TStTime) : TDateTime
 Function Convert2ByteDate( TwoByteDate : Word) : TStDate
 Function Convert4ByteDate( FourByteDate : TStDate) : Word

Procedure SIRegister_StDateSt(CL: TPSPascalCompiler);
Function DateStringHMStoAstJD( const Picture, DS : string; H, M, S, Epoch : integer) : Double
 Function MonthToString( const Month : Integer) : string
 Function DateStringToStDate( const Picture, S : string; Epoch : Integer) : TStDate
 Function DateStringToDMY(const Picture,S:string; Epoch:Integer; var D, M, Y : Integer):Boolean
 Function StDateToDateString( const Picture : string; const Julian : TStDate; Pack : Boolean):string
 Function DayOfWeekToString( const WeekDay : TStDayType) : string
 Function DMYtoDateString(const Picture:string;Day,Month,Year,Epoch:Integer;Pack:Boolean): string);
 Function CurrentDateString( const Picture : string; Pack : Boolean) : string
 Function CurrentTimeString( const Picture : string; Pack : Boolean) : string
 Function TimeStringToHMS( const Picture, St : string; var H, M, S : Integer) : Boolean
 Function TimeStringToStTime( const Picture, S : string) : TStTime
 Function StTimeToAmPmString( const Picture : string; const T : TStTime; Pack : Boolean) : string
 Function StTimeToTimeString( const Picture : string; const T : TStTime; Pack : Boolean) : string
 Function DateStringIsBlank( const Picture, S : string) : Boolean
 Function InternationalDate( ForceCentury : Boolean) : string
 Function InternationalLongDate( ShortNames : Boolean; ExcludeDOW : Boolean) : string
 Function InternationalTime( ShowSeconds : Boolean) : string
 Procedure ResetInternationalInfo

procedure SIRegister_StBase(CL: TPSPascalCompiler);
 Function DestroyNode( Container : TStContainer; Node : TStNode; OtherData : Pointer) : Boolean
 Function AnsiUpperCaseShort32( const S : string) : string
 Function AnsiCompareTextShort32( const S1, S2 : string) : Integer
 Function AnsiCompareStrShort32( const S1, S2 : string) : Integer
 Function HugeCompressRLE( const InBuffer, InLen : Longint; var OutBuffer) : Longint
 Function HugeDecompressRLE( const InBuffer, InLen : Longint; var OutBuffer,OutLen:LongInt): Longint
 Procedure HugeFillChar( var Dest, Count : Longint; Value : Byte)
 Procedure HugeFillStruc( var Dest, Count : Longint; const Value, ValSize : Cardinal)
 Function Upcase( C : AnsiChar) : AnsiChar
 Function LoCase( C : AnsiChar) : AnsiChar
 Function CompareLetterSets( Set1, Set2 : LongInt) : Cardinal
 Function CompStruct( const S1, S2, Size : Cardinal) : Integer
 Function Search(const Buffer,BufLength:Cardinal;const Match,MatLength:Cardinal;var Pos:Cardinal):Bool;
 Function StSearch(const Buffer,BufLength:Cardinal;const Match,MatLength:Cardinal;var Pos:Cardi):Boolean
 Function SearchUC(const Buffer,BufLength:Cardinal;const Match,MatLength:Cardinal;var Pos:Cardi):Boolean
 Function IsOrInheritsFrom( Root, Candidate : TClass) : boolean
 Procedure RaiseContainerError( Code : longint)
 Procedure RaiseContainerErrorFmt( Code : Longint; Data : array of const)
 Function ProductOverflow( A, B : LongInt) : Boolean
 Function StNewStr( S : string) : PShortString
 Procedure StDisposeStr( PS : PShortString)
 Procedure ValLongInt( S : ShortString; var LI : Longint; var ErrorCode : integer)
 Procedure ValSmallint( const S : ShortString; var SI : smallint; var ErrorCode : integer)
 Procedure ValWord( const S : ShortString; var Wd : word; var ErrorCode : integer)
 Procedure RaiseStError( ExceptionClass : EStExceptionClass; Code : LongInt)
 Procedure RaiseStWin32Error( ExceptionClass : EStExceptionClass; Code : LongInt)
 Procedure RaiseStWin32ErrorEx( ExceptionClass : EStExceptionClass; Code : LongInt; Info : string)

procedure SIRegister_usvd(CL: TPSPascalCompiler);
begin
 Procedure SV_Decomp( A : TMatrix; Lb, Ub1, Ub2 : Integer; S : TVector; V : TMatrix)
 Procedure SV_SetZero( S : TVector; Lb, Ub : Integer; Tol : Float)
 Procedure SV_Solve(U:TMatrix; S:TVector;V:TMatrix;B:TVector;Lb,Ub1,Ub2:Integer;X:TVector);
 Procedure SV_Approx( U : TMatrix; S : TVector; V : TMatrix; Lb, Ub1, Ub2 : Integer; A : TMatrix)
 Procedure RKF45(F:TDiffEqs;Neqn:Int;Y,Yp:TVector;var T:Float;Tout,RelErr,AbsErr:Float;var Flag:Int;
end;

//**********unit unit ; StMath Package of SysTools******************************************
Function IntPowerS( Base : Extended; Exponent : Integer) : Extended
Function PowerS( Base, Exponent : Extended) : Extended
Function StInvCos( X : Double) : Double
Function StInvSin( Y : Double) : Double
Function StInvTan2( X, Y : Double) : Double
Function StTan( A : Double) : Double
Procedure DumpException;  //unit StExpEng;
Function HexifyBlock( var Buffer, BufferSize : Integer) : string

//**********unit unit ; StCRC Package of SysTools******************************************
Function Adler32Prim( var Data, DataSize : Cardinal; CurCrc : LongInt) : LongInt
Function Adler32OfStream( Stream : TStream; CurCrc : LongInt) : LongInt
Function Adler32OfFile( FileName : AnsiString) : LongInt
Function Crc16Prim( var Data, DataSize, CurCrc : Cardinal) : Cardinal
Function Crc16OfStream( Stream : TStream; CurCrc : Cardinal) : Cardinal
Function Crc16OfFile( FileName : AnsiString) : Cardinal
Function Crc32Prim( var Data, DataSize : Cardinal; CurCrc : LongInt) : LongInt
Function Crc32OfStream( Stream : TStream; CurCrc : LongInt) : LongInt
Function Crc32OfFile( FileName : AnsiString) : LongInt
Function InternetSumPrim( var Data, DataSize, CurCrc : Cardinal) : Cardinal
Function InternetSumOfStream( Stream : TStream; CurCrc : Cardinal) : Cardinal
Function InternetSumOfFile( FileName : AnsiString) : Cardinal
Function Kermit16Prim( var Data, DataSize, CurCrc : Cardinal) : Cardinal
Function Kermit16OfStream( Stream : TStream; CurCrc : Cardinal) : Cardinal
Function Kermit16OfFile( FileName : AnsiString) : Cardinal

//**********unit unit ; StBCD Package of SysTools*****************************************
Function AddBcd( const B1, B2 : TbcdS) : TbcdS
Function SubBcd( const B1, B2 : TbcdS) : TbcdS
Function MulBcd( const B1, B2 : TbcdS) : TbcdS
Function DivBcd( const B1, B2 : TbcdS) : TbcdS
Function ModBcd( const B1, B2 : TbcdS) : TbcdS
Function NegBcd( const B : TbcdS) : TbcdS
Function AbsBcd( const B : TbcdS) : TbcdS
Function FracBcd( const B : TbcdS) : TbcdS
Function IntBcd( const B : TbcdS) : TbcdS
Function RoundDigitsBcd( const B : TbcdS; Digits : Cardinal) : TbcdS
Function RoundPlacesBcd( const B : TbcdS; Places : Cardinal) : TbcdS
Function ValBcd( const S : string) : TbcdS
Function LongBcd( L : LongInt) : TbcdS
Function ExtBcd( E : Extended) : TbcdS
Function ExpBcd( const B : TbcdS) : TbcdS
Function LnBcd( const B : TbcdS) : TbcdS
Function IntPowBcd( const B : TbcdS; E : LongInt) : TbcdS
Function PowBcd( const B, E : TbcdS) : TbcdS
Function SqrtBcd( const B : TbcdS) : TbcdS
Function CmpBcd( const B1, B2 : TbcdS) : Integer
Function EqDigitsBcd( const B1, B2 : TbcdS; Digits : Cardinal) : Boolean
Function EqPlacesBcd( const B1, B2 : TbcdS; Digits : Cardinal) : Boolean
Function IsIntBcd( const B : TbcdS) : Boolean
Function TruncBcd( const B : TbcdS) : LongInt
Function BcdExt( const B : TbcdS) : Extended
Function RoundBcd( const B : TbcdS) : LongInt
Function StrBcd( const B : TbcdS; Width, Places : Cardinal) : string
Function StrExpBcd( const B : TbcdS; Width : Cardinal) : string
Function FormatBcd( const Format : string; const B : TbcdS) : string
Function StrGeneralBcd( const B : TbcdS) : string
Function FloatFormBcd(const Mask:string;B:TbcdS;const LtCurr,RtCurr:string;Sep,DecPt:AnsiChar):string
Procedure ConvertBcd( const SrcB, SrcSize : Byte; var DestB, DestSize : Byte)

////*******unit unit ; StTxtDat; TStTextDataRecordSet Package of SysTools**********************
Procedure StParseLine( const Data : AnsiString; Schema : TStTextDataSchema; Result : TStrings)
Function StFieldTypeToStr( FieldType : TStSchemaFieldType) : AnsiString
Function StStrToFieldType( const S : AnsiString) : TStSchemaFieldType
Function StDeEscape( const EscStr : AnsiString) : Char
Function StDoEscape( Delim : Char) : AnsiString
Function StTrimTrailingChars( const S : AnsiString; Trailer : Char) : AnsiString
Function AnsiHashText( const S : string; Size : Integer) : Integer
Function AnsiHashStr( const S : string; Size : Integer) : Integer
Function AnsiELFHashText( const S : string; Size : Integer) : Integer
Function AnsiELFHashStr( const S : string; Size : Integer) : Integer

//**********unit unit ; StNetCon Package of SysTools******************************************
  with AddClassN(FindClass('TStComponent'),'TStNetConnection') do begin
    Constructor Create( AOwner : TComponent)
    Function Connect : DWord
    Function Disconnect : DWord
    RegisterProperty('Password', 'String', iptrw);
    Property('UserName', 'String', iptrw);
    Property('ConnectOptions', 'TStNetConnectOptionsSet', iptrw);
    Property('DisconnectOptions', 'TStNetDisconnectOptionsSet', iptrw);
    Property('LocalDevice', 'String', iptrw);
    Property('ServerName', 'String', iptrw);
    Property('ShareName', 'String', iptrw);
    Property('OnConnect', 'TNotifyEvent', iptrw);
    Property('OnConnectFail', 'TOnConnectFailEvent', iptrw);
    Property('OnConnectCancel', 'TOnConnectCancelEvent', iptrw);
    Property('OnDisconnect', 'TNotifyEvent', iptrw);
    Property('OnDisconnectFail', 'TOnDisconnectFailEvent', iptrw);
    Property('OnDisconnectCancel', 'TOnDisconnectCancelEvent', iptrw);
  end;
//***********Thread Functions Context of Win API --- more objects in SyncObjs.pas
/ /153 unit uPSI_SyncObjs, unit uPSIParallelJobs;
Procedure InitializeCriticalSection( var lpCriticalSection : TRTLCriticalSection)
Procedure EnterCriticalSection( var lpCriticalSection : TRTLCriticalSection)
Procedure LeaveCriticalSection( var lpCriticalSection : TRTLCriticalSection)
Function InitializeCriticalSectionAndSpinCount(var
lpCriticalSection:TRTLCriticalSection;dwSpinCount:DWORD):BOOL;
Function SetCriticalSectionSpinCount(var lpCriticalSection:TRTLCriticalSection;dwSpinCount:DWORD):DWORD;
Function TryEnterCriticalSection( var lpCriticalSection : TRTLCriticalSection) : BOOL
```

```
Procedure DeleteCriticalSection( var lpCriticalSection : TRTLCriticalSection)
Function GetThreadContext( hThread : THandle; var lpContext : TContext) : BOOL
Function SetThreadContext( hThread : THandle; const lpContext : TContext) : BOOL
Function SuspendThread( hThread : THandle) : DWORD
Function ResumeThread( hThread : THandle) : DWORD
Function CreateThread2(ThreadFunc: TThreadFunction2; thrid: DWord) : THandle
Function GetCurrentThread : THandle
Procedure ExitThread( dwExitCode : DWORD)
Function TerminateThread( hThread : THandle; dwExitCode : DWORD) : BOOL
Function GetExitCodeThread( hThread : THandle; var lpExitCode : DWORD) : BOOL
Procedure EndThread(ExitCode: Integer);
Function WaitForSingleObject( hHandle : THandle; dwMilliseconds : DWORD) : DWORD
Function MakeProcInstance( Proc : FARPROC; Instance : THandle) : FARPROC
Procedure FreeProcInstance( Proc : FARPROC)
Procedure FreeLibraryAndExitThread( hLibModule : HMODULE; dwExitCode : DWORD)
Function DisableThreadLibraryCalls( hLibModule : HMODULE) : BOOL
Procedure ParallelJob( ASelf : TObject; ATarget : Pointer; AParam : Pointer; ASafeSection : boolean);
Procedure ParallelJob1( ATarget : Pointer; AParam : Pointer; ASafeSection : boolean);
Procedure ParallelJob2(AJobGroup:TJobsGroup;ASelf:TObject;ATarget:Ptr;AParam:Pointer;ASafeSection:bool;
Procedure ParallelJob3( AJobGroup:TJobsGroup;ATarget:Pointer;AParam:Pointer;ASafeSection: boolean);
Function CreateParallelJob(ASelf:TObject;ATarget:Pointer;AParam:Ptr;ASafeSection:bool:TParallelJob;
Function CreateParallelJob1(ATarget:Pointer; AParam:Pointer; ASafeSection : boolean) : TParallelJob;
Function CurrentParallelJobInfo : TParallelJobInfo
Function ObtainParallelJobInfo : TParallelJobInfo

****************************************************unit uPSI_JclMime;
 Function MimeEncodeString( const S : AnsiString) : AnsiString
 Function MimeDecodeString( const S : AnsiString) : AnsiString
 Procedure MimeEncodeStream( const InputStream : TStream; const OutputStream : TStream)
 Procedure MimeDecodeStream( const InputStream : TStream; const OutputStream : TStream)
 Function MimeEncodedSize( const I : Cardinal) : Cardinal
 Function MimeDecodedSize( const I : Cardinal) : Cardinal
 Procedure MimeEncode( var InputBuffer : string; const InputByteCount : Cardinal; var OutputBuffer)
 Function MimeDecode(var InputBuffer:string;const InputBytesCount:Cardinal;var OutputBuffer):Cardinal;
 Function MimeDecodePartial(var InputBuffer:string;const InputBytesCount:Cardinal;var
OutputBuffer:string;var ByteBuffer : Cardinal; var ByteBufferSpace : Cardinal) : Cardinal
 Function MimeDecodePartialEnd(var OutputBuf:string;const ByteBuf:Card;const ByteBufferSpace:Card):
Cardinal;

**************************************************unit uPSI_JclPrint;
 Procedure DirectPrint( const Printer, Data : string)
 Procedure SetPrinterPixelsPerInch
 Function GetPrinterResolution : TPoint
 Function CharFitsWithinDots( const Text : string; const Dots : Integer) : Integer
 Procedure PrintMemo( const Memo : TMemo; const Rect : TRect)


//***************************************unit uPSI_ShLwApi;***************************************
 Function StrChr( lpStart : PChar; wMatch : WORD) : PChar
 Function StrChrI( lpStart : PChar; wMatch : WORD) : PChar
 Function StrCmpN( lpStr1, lpStr2 : PChar; nChar : Integer) : Integer
 Function StrCmpNI( lpStr1, lpStr2 : PChar; nChar : Integer) : Integer
 Function StrCSpn( lpStr_, lpSet : PChar) : Integer
 Function StrCSpnI( lpStr1, lpSet : PChar) : Integer
 Function StrDup( lpSrch : PChar) : PChar
 Function StrFormatByteSize( dw : DWORD; szBuf : PChar; uiBufSize : UINT) : PChar
 Function StrFormatKBSize( qdw : Dword; szBuf : PChar; uiBufSize : UINT) : PChar
 Function StrFromTimeInterval(pszOut: PChar; cchMax:UINT;dwTimeMS:DWORD; digits: Integer): Integer
 Function StrIsIntlEqual( fCaseSens : BOOL; lpString1, lpString2 : PChar; nChar : Integer) : BOOL
/Function StrNCat( psz1 : PChar; psz2 : PChar; cchMax : Integer) : PChar
 Function StrPBrk( psz, pszSet : PChar) : PChar
 Function StrRChr( lpStart, lpEnd : PChar; wMatch : WORD) : PChar
 Function StrRChrI( lpStart, lpEnd : PChar; wMatch : WORD) : PChar
 Function StrRStrI( lpSource, lpLast, lpSrch : PChar) : PChar
 Function StrSpn( psz, pszSet : PChar) : Integer
 Function StrStr( lpFirst, lpSrch : PChar) : PChar
 Function StrStrI( lpFirst, lpSrch : PChar) : PChar
 Function StrToInt( lpSrch : PChar) : Integer
 Function StrToIntEx( pszString : PChar; dwFlags : DWORD; var piRet : Integer) : BOOL
 Function StrTrim( psz : PChar; pszTrimChars : PChar) : BOOL
 Function ChrCmpI( w1, w2 : WORD) : BOOL
 Function ChrCmpIA( w1, w2 : WORD) : BOOL
 Function ChrCmpIW( w1, w2 : WORD) : BOOL
 Function StrIntlEqN( s1, s2 : PChar; nChar : Integer) : BOOL
 Function StrIntlEqNI( s1, s2 : PChar; nChar : Integer) : BOOL
 Function StrCatBuff( pszDest, pszSrc : PChar; cchDestBuffSize : Integer) : PChar
 Function StrCpyNX( psz1 : PChar; psz2 : PChar; cchMax : Integer) : PChar
 Function IntlStrEqWorker( fCaseSens : BOOL; lpString1, lpString2 : PChar; nChar : Integer) : BOOL
 Function IntlStrEqN( s1, s2 : PChar; nChar : Integer) : BOOL
 SZ_CONTENTTYPE_HTMLA','String 'text/html
 SZ_CONTENTTYPE_HTMLW','String 'text/html
 SZ_CONTENTTYPE_HTML','string SZ_CONTENTTYPE_HTMLA);
 SZ_CONTENTTYPE_CDFA','String 'application/x-cdf
 SZ_CONTENTTYPE_CDFW','String 'application/x-cdf
 SZ_CONTENTTYPE_CDF','string SZ_CONTENTTYPE_CDFA);
 Function PathIsHTMLFile( pszPath : PChar) : BOOL
 STIF_DEFAULT','LongWord( $00000000);
 STIF_SUPPORT_HEX','LongWord( $00000001);
 Function StrNCmpI( lpStr1, lpStr2 : PChar; nChar : Integer) : Integer
 Function StrNCpy( psz1, psz2 : PChar; cchMax : Integer) : PChar
 Function StrCatN( psz1 : PChar; psz2 : PChar; cchMax : Integer) : PChar
 Function PathAddBackslash( pszPath : PChar) : PChar
 Function PathAddExtension( pszPath : PChar; pszExt : PChar) : BOOL
 Function PathAppend( pszPath : PChar; pMore : PChar) : BOOL
 Function PathBuildRoot( szRoot : PChar; iDrive : Integer) : PChar
 Function PathCanonicalize( pszBuf : PChar; pszPath : PChar) : BOOL
 Function PathCombine( szDest : PChar; lpszDir, lpszFile : PChar) : PChar
 Function PathCompactPath( hDC : HDC; pszPath : PChar; dx : UINT) : BOOL
 Function PathCompactPathEx( pszOut : PChar; pszSrc : PChar; cchMax : UINT; dwFlags:DWORD) : BOOL
 Function PathCommonPrefix( pszFile1, pszFile2 : PChar; achPath : PChar) : Integer
 Function PathFileExists( pszPath : PChar) : BOOL
 Function PathFindExtension( pszPath : PChar) : PChar
 Function PathFindFileName( pszPath : PChar) : PChar
 Function PathFindNextComponent( pszPath : PChar) : PChar
 Function PathFindOnPath( pszPath : PChar; var ppszOtherDirs : PChar) : BOOL
 Function PathGetArgs( pszPath : PChar) : PChar
 Function PathFindSuffixArray(pszPath: PChar; const apszSuffix: PChar; iArraySize: Integer): PChar
 Function PathIsLFNFileSpec( lpName : PChar) : BOOL
 Function PathGetCharType( ch : Char) : UINT
 GCT_INVALID','LongWord( $0000);
 GCT_LFNCHAR','LongWord( $0001);
 GCT_SHORTCHAR','LongWord( $0002);
 GCT_WILD','LongWord( $0004);
 GCT_SEPARATOR','LongWord( $0008);
 Function PathGetDriveNumber( pszPath : PChar) : Integer
 Function PathIsDirectory( pszPath : PChar) : BOOL
 Function PathIsDirectoryEmpty( pszPath : PChar) : BOOL
 Function PathIsFileSpec( pszPath : PChar) : BOOL
 Function PathIsPrefix( pszPrefix, pszPath : PChar) : BOOL
 Function PathIsRelative( pszPath : PChar) : BOOL
 Function PathIsRoot( pszPath : PChar) : BOOL
 Function PathIsSameRoot( pszPath1, pszPath2 : PChar) : BOOL
 Function PathIsUNC( pszPath : PChar) : BOOL
 Function PathIsNetworkPath( pszPath : PChar) : BOOL
 Function PathIsUNCServer( pszPath : PChar) : BOOL
 Function PathIsUNCServerShare( pszPath : PChar) : BOOL
 Function PathIsContentType( pszPath, pszContentType : PChar) : BOOL
 Function PathIsURL( pszPath : PChar) : BOOL
 Function PathMakePretty( pszPath : PChar) : BOOL
 Function PathMatchSpec( pszFile, pszSpec : PChar) : BOOL
 Function PathParseIconLocation( pszIconFile : PChar) : Integer
 Procedure PathQuoteSpaces( lpsz : PChar)
 Function PathRelativePathTo(pszPath:PChar;pszFrom:PChar;dwAttrFrom:DWORD;pszTo:PChar;dwAttrTo:DWORD):BOOL;
 Procedure PathRemoveArgs( pszPath : PChar)
 Function PathRemoveBackslash( pszPath : PChar) : PChar
 Procedure PathRemoveBlanks( pszPath : PChar)
 Procedure PathRemoveExtension( pszPath : PChar)
 Function PathRemoveFileSpec( pszPath : PChar) : BOOL
 Function PathRenameExtension( pszPath : PChar; pszExt : PChar) : BOOL
 Function PathSearchAndQualify( pszPath : PChar; pszBuf : PChar; cchBuf : UINT) : BOOL
 Procedure PathSetDlgItemPath( hDlg : HWND; id : Integer; pszPath : PChar)
 Function PathSkipRoot( pszPath : PChar) : PChar
 Procedure PathStripPath( pszPath : PChar)
 Function PathStripToRoot( pszPath : PChar) : BOOL
 Procedure PathUnquoteSpaces( lpsz : PChar)
 Function PathMakeSystemFolder( pszPath : PChar) : BOOL
 Function PathUnmakeSystemFolder( pszPath : PChar) : BOOL
 Function PathIsSystemFolder( pszPath : PChar; dwAttrb : DWORD) : BOOL
 Procedure PathUndecorate( pszPath : PChar)
 Function PathUnExpandEnvStrings( pszPath : PChar; pszBuf : PChar; cchBuf : UINT) : BOOL
 URL_SCHEME_INVALID','LongInt'( - 1);
 URL_SCHEME_UNKNOWN','LongInt'( 0);
 URL_SCHEME_FTP','LongInt'( 1);
 URL_SCHEME_HTTP','LongInt'( 2);
 URL_SCHEME_GOPHER','LongInt'( 3);
 URL_SCHEME_MAILTO','LongInt'( 4);
 URL_SCHEME_NEWS','LongInt'( 5);
 URL_SCHEME_NNTP','LongInt'( 6);
 URL_SCHEME_TELNET','LongInt'( 7);
 URL_SCHEME_WAIS','LongInt'( 8);
 URL_SCHEME_FILE','LongInt'( 9);
 URL_SCHEME_MK','LongInt'( 10);
 URL_SCHEME_HTTPS','LongInt'( 11);
 URL_SCHEME_SHELL','LongInt'( 12);
 URL_SCHEME_SNEWS','LongInt'( 13);
 URL_SCHEME_LOCAL','LongInt'( 14);
 URL_SCHEME_JAVASCRIPT','LongInt'( 15);
 URL_SCHEME_VBSCRIPT','LongInt'( 16);
 URL_SCHEME_ABOUT','LongInt'( 17);
 URL_SCHEME_RES','LongInt'( 18);
 URL_SCHEME_MAXVALUE','LongInt'( 19);
 URL_SCHEME', 'Integer
 URL_PART_NONE','LongInt'( 0);
 URL_PART_SCHEME','LongInt'( 1);
 URL_PART_HOSTNAME','LongInt'( 2);
 URL_PART_USERNAME','LongInt'( 3);
 URL_PART_PASSWORD','LongInt'( 4);
 URL_PART_PORT','LongInt'( 5);
```

```
URL_PART_QUERY','LongInt'( 6);
URL_PART', 'DWORD
URLIS_URL','LongInt'( 0);
URLIS_OPAQUE','LongInt'( 1);
URLIS_NOHISTORY','LongInt'( 2);
URLIS_FILEURL','LongInt'( 3);
URLIS_APPLIABLE','LongInt'( 4);
URLIS_DIRECTORY','LongInt'( 5);
URLIS_HASQUERY','LongInt'( 6);
TUrlIs', 'DWORD
URL_UNESCAPE','LongWord( $10000000);
URL_ESCAPE_UNSAFE','LongWord( $20000000);
URL_PLUGGABLE_PROTOCOL','LongWord( $40000000);
URL_WININET_COMPATIBILITY','LongWord( DWORD ( $80000000 ));
URL_DONT_ESCAPE_EXTRA_INFO','LongWord( $02000000);
URL_ESCAPE_SPACES_ONLY','LongWord( $04000000);
URL_DONT_SIMPLIFY','LongWord( $08000000);
URL_NO_META','longword( URL_DONT_SIMPLIFY);
URL_UNESCAPE_INPLACE','LongWord( $00100000);
URL_CONVERT_IF_DOSPATH','LongWord( $00200000);
URL_UNESCAPE_HIGH_ANSI_ONLY','LongWord( $00400000);
URL_INTERNAL_PATH','LongWord( $00800000);
URL_FILE_USE_PATHURL','LongWord( $00010000);
URL_ESCAPE_PERCENT','LongWord( $00001000);
URL_ESCAPE_SEGMENT_ONLY','LongWord( $00002000);
URL_PARTFLAG_KEEPSCHEME','LongWord( $00000001);
URL_APPLY_DEFAULT','LongWord( $00000001);
URL_APPLY_GUESSSCHEME','LongWord( $00000002);
URL_APPLY_GUESSFILE','LongWord( $00000004);
URL_APPLY_FORCEAPPLY','LongWord( $00000008);
Function UrlCompare( psz1, psz2 : PChar; fIgnoreSlash : BOOL) : Integer
Function UrlCombine(pszBase,pszRelative:PChar;pszCombin:PChar;out pcchCombin:DWORD;dwFlags:DWORD):HRESULT;
Function UrlCanonicalize(pszUrl:PChar;pszCanonicalized:PChar;pcchCanonic:DWORD;dwFlags:DWORD):HRESULT;
Function UrlIsOpaque( pszURL : PChar) : BOOL
Function UrlIsNoHistory( pszURL : PChar) : BOOL
Function UrlIsFileUrl( pszURL : PChar) : BOOL
Function UrlIs( pszUrl : PChar; UrlIs : TUrlIs) : BOOL
Function UrlGetLocation( psz1 : PChar) : PChar
Function UrlUnescape( pszUrl, pszUnescaped : PChar;pcchUnescaped:DWORD; dwFlags : DWORD) : HRESULT
Function UrlEscape(pszUrl: PChar; pszEscaped: PChar; pcchEscaped:DWORD; dwFlags : DWORD) : HRESULT
Function UrlCreateFromPath(pszPath:PChar; pszUrl: PChar;pcchUrl: DWORD; dwFlags : DWORD) : HRESULT
Function PathCreateFromUrl(pszUrl:PChar; pszPath:PChar; pcchPath:DWORD; dwFlags : DWORD) : HRESULT
Function UrlHash( pszUrl : PChar; pbHash : BYTE; cbHash : DWORD) : HRESULT
Function UrlGetPart(pszIn: PChar; pszOut: PChar; pcchOut: DWORD; dwPart,dwFlags: DWORD) : HRESULT
Function UrlApplyScheme( pszIn : PChar; pszOut : PChar; pcchOut : DWORD; dwFlags : DWORD) : HRESULT
Function HashData( pbData : BYTE; cbData : DWORD; pbHash : BYTE; cbHash : DWORD) : HRESULT
Function UrlEscapeSpaces( pszUrl : PChar; pszEscaped : PChar; pcchEscaped : DWORD) : HRESULT
Function UrlUnescapeInPlace( pszUrl : PChar; dwFlags : DWORD) : HRESULT
Function SHDeleteEmptyKey( hKey : HKEY; pszSubKey : PChar) : DWORD
Function SHDeleteKey( hKey : HKEY; pszSubKey : PChar) : DWORD
Function SHDeleteValue( hKey : HKEY; pszSubKey, pszValue : PChar) : DWORD
Function SHEnumKeyEx( hKey : HKEY; dwIndex : DWORD; pszName : PChar; var pcchName : DWORD) : Longint
Function SHEnumValue( hKey : HKEY; dwIndex : DWORD; pszValueName:PChar; var
pcchValueName:DWORD;pdwType:DWORD; pvData : ___Pointer; pcbData : DWORD) : Longint
Function SHQueryInfoKey(hKey:HKEY;pcSubKeys,pcchMaxSubKeyLen,pcVal,pcchMaxValueNameLen:DWORD):Longint;
Function SHCopyKey( hkeySrc : HKEY; szSrcSubKey : PChar; hkeyDest : HKEY; fReserved : DWORD) : DWORD
Function SHRegGetPath(hKey:HKEY; pcszSubKey,pcszValue: PChar; pszPath: PChar; dwFlags: DWORD): DWORD
Function SHRegSetPath( hKey:HKEY; pcszSubKey, pcszValue, pcszPath : PChar; dwFlags : DWORD): DWORD
SHREGDEL_DEFAULT','LongWord( $00000000);
SHREGDEL_HKCU','LongWord( $00000001);
SHREGDEL_HKLM','LongWord( $00000010);
SHREGDEL_BOTH','LongWord( $00000011);
SHREGENUM_DEFAULT','LongWord( $00000000);
SHREGENUM_HKCU','LongWord( $00000001);
SHREGENUM_HKLM','LongWord( $00000010);
SHREGENUM_BOTH','LongWord( $00000011);
SHREGSET_HKCU','LongWord( $00000001);
SHREGSET_FORCE_HKCU','LongWord( $00000002);
SHREGSET_HKLM','LongWord( $00000004);
SHREGSET_FORCE_HKLM','LongWord( $00000008);
TSHRegDelFlags', 'DWORD
TSHRegEnumFlags', 'DWORD
HUSKEY', 'THandle
ASSOCF_INIT_NOREMAPCLSID','LongWord( $00000001);
ASSOCF_INIT_BYEXENAME','LongWord( $00000002);
ASSOCF_OPEN_BYEXENAME','LongWord( $00000002);
ASSOCF_INIT_DEFAULTTOSTAR','LongWord( $00000004);
ASSOCF_INIT_DEFAULTTOFOLDER','LongWord( $00000008);
ASSOCF_NOUSERSETTINGS','LongWord( $00000010);
ASSOCF_NOTRUNCATE','LongWord( $00000020);
ASSOCF_VERIFY','LongWord( $00000040);
ASSOCF_REMAPRUNDLL','LongWord( $00000080);
ASSOCF_NOFIXUPS','LongWord( $00000100);
ASSOCF_IGNOREBASECLASS','LongWord( $00000200);
ASSOCF', 'DWORD
ASSOCSTR_COMMAND','LongInt'( 1);
ASSOCSTR_EXECUTABLE','LongInt'( 2);
ASSOCSTR_FRIENDLYDOCNAME','LongInt'( 3);
ASSOCSTR_FRIENDLYAPPNAME','LongInt'( 4);
```

```
 ASSOCSTR_NOOPEN','LongInt'( 5);
 ASSOCSTR_SHELLNEWVALUE','LongInt'( 6);
 ASSOCSTR_DDECOMMAND','LongInt'( 7);
 ASSOCSTR_DDEIFEXEC','LongInt'( 8);
 ASSOCSTR_DDEAPPLICATION','LongInt'( 9);
 ASSOCSTR_DDETOPIC','LongInt'( 10);
 ASSOCSTR_INFOTIP','LongInt'( 11);
 ASSOCSTR_MAX','LongInt'( 12);
 ASSOCSTR', 'DWORD
 ASSOCKEY_SHELLEXECCLASS','LongInt'( 1);
 ASSOCKEY_APP','LongInt'( 2);
 ASSOCKEY_CLASS','LongInt'( 3);
 ASSOCKEY_BASECLASS','LongInt'( 4);
 ASSOCKEY_MAX','LongInt'( 5);
 ASSOCKEY', 'DWORD
 ASSOCDATA_MSIDESCRIPTOR','LongInt'( 1);
 ASSOCDATA_NOACTIVATEHANDLER','LongInt'( 2);
 ASSOCDATA_QUERYCLASSSTORE','LongInt'( 3);
 ASSOCDATA_HASPERUSERASSOC','LongInt'( 4);
 ASSOCDATA_MAX','LongInt'( 5);
 ASSOCDATA', 'DWORD
 ASSOCENUM_NONE','LongInt'( 0);
 ASSOCENUM', 'DWORD
 SID_IQueryAssociations','String '{c46ca590-3c3f-11d2-bee6-0000f805ca57}
 SHACF_DEFAULT $00000000);
 SHACF_FILESYSTEM','LongWord( $00000001);
 SHACF_URLHISTORY','LongWord( $00000002);
 SHACF_URLMRU','LongWord( $00000004);
 SHACF_USETAB','LongWord( $00000008);
 SHACF_FILESYS_ONLY','LongWord( $00000010);
 SHACF_AUTOSUGGEST_FORCE_ON','LongWord( $10000000);
 SHACF_AUTOSUGGEST_FORCE_OFF','LongWord( $20000000);
 SHACF_AUTOAPPEND_FORCE_ON','LongWord( $40000000);
 SHACF_AUTOAPPEND_FORCE_OFF','LongWord( DWORD ( $80000000 ));
 Function SHAutoComplete( hwndEdit : HWND; dwFlags : DWORD) : HRESULT
 Procedure SHSetThreadRef( punk : IUnknown)
 Procedure SHGetThreadRef( out ppunk : IUnknown)
 CTF_INSIST','LongWord( $00000001);
 CTF_THREAD_REF','LongWord( $00000002);
 CTF_PROCESS_REF','LongWord( $00000004);
 CTF_COINIT','LongWord( $00000008);
 Function SHCreateShellPalette( hdc : HDC) : HPALETTE
 Procedure ColorRGBToHLS( clrRGB : TColorRef; out pwHue, pwLuminance, pwSaturation : WORD)
 Function ColorHLSToRGB( wHue, wLuminance, wSaturation : WORD) : TColorRef
 Function ColorAdjustLuma( clrRGB : TColorRef; n : Integer; fScale : Boolean) : TColorRef
 Function GetSysColorBrush( nIndex : Integer) : HBRUSH
 Function DrawFocusRect( hDC : HDC; const lprc : TRect) : BOOL
 Function FillRect( hDC : HDC; const lprc : TRect; hbr : HBRUSH) : Integer
 Function FrameRect( hDC : HDC; const lprc : TRect; hbr : HBRUSH) : Integer
 Function InvertRect( hDC : HDC; const lprc : TRect) : BOOL
 Function SetRect( var lprc : TRect; xLeft, yTop, xRight, yBottom : Integer) : BOOL
 Function SetRectEmpty( var lprc : TRect) : BOOL
 Function CopyRect( var lprcDst : TRect; const lprcSrc : TRect) : BOOL
 Function InflateRect( var lprc : TRect; dx, dy : Integer) : BOOL
 Function IntersectRect2( var lprcDst : TRect; const lprcSrc1, lprcSrc2 : TRect) : BOOL
 Function SubtractRect( var lprcDst : TRect; const lprcSrc1, lprcSrc2 : TRect) : BOOL

 Function InitializeFlatSB( hWnd : HWND) : Bool
 Procedure UninitializeFlatSB( hWnd : HWND)
 Function FlatSB_GetScrollProp( p1 : HWND; propIndex : Integer; p3 : PInteger) : Bool
 Function FlatSB_SetScrollProp( p1 : HWND; index : Integer; newValue : Integer; p4 : Bool) : Bool
 Function GET_APPCOMMAND_LPARAM( lParam : Integer) : Word  //of JvWin32
 Function GET_DEVICE_LPARAM( lParam : Integer) : Word
 Function GET_MOUSEORKEY_LPARAM( lParam : Integer) : Integer
 Function GET_FLAGS_LPARAM( lParam : Integer) : Word
 Function GET_KEYSTATE_LPARAM( lParam : Integer) : Word


// ********************************************* 204 unit uPSI_ShellAPI;
Function DragQueryFile( Drop : HDROP; FileIndex : UINT; FileName : PChar; cb : UINT) : UINT
Function DragQueryPoint( Drop : HDROP; var Point : TPoint) : BOOL
Procedure DragFinish( Drop : HDROP)
Procedure DragAcceptFiles( Wnd : HWND; Accept : BOOL)
Function ShellExecute(hWnd:HWND;Operation,FileName,Parameters,Directory:PChar;ShowCmd:Integer):HINST
Function FindExecutable( FileName, Directory : PChar; Result : PChar) : HINST
Function ShellAbout( Wnd : HWND; szApp, szOtherStuff : PChar; Icon : HICON) : Integer
Function DuplicateIcon( hInst : HINST; Icon : HICON) : HICON
Function ExtractAssociatedIcon( hInst : HINST; lpIconPath : PChar; var lpiIcon : Word) : HICON
Function ExtractIcon( hInst : HINST; lpszExeFileName : PChar; nIconIndex : UINT) : HICON
Function SHAppBarMessage( dwMessage : DWORD; var pData : TAppBarData) : UINT
Function DoEnvironmentSubst( szString : PChar; cbString : UINT) : DWORD
Function ExtractIconEx(lpszFile:PChar;nIconIndex:Int;var phiconLarge,phiconSmall:HICON;nIcons:UINT):UINT;
Procedure SHFreeNameMappings( hNameMappings : THandle)

 DLLVER_PLATFORM_WINDOWS','LongWord( $00000001);
 DLLVER_PLATFORM_NT','LongWord( $00000002);
 DLLVER_MAJOR_MASK','LongWord( Int64 ( $FFFF000000000000 ));
 DLLVER_MINOR_MASK','LongWord( Int64 ( $0000FFFF00000000 ));
 DLLVER_BUILD_MASK','LongWord( Int64 ( $00000000FFFF0000 ));
 DLLVER_QFE_MASK','LongWord( Int64 ( $000000000000FFFF ));
 Function MAKEDLLVERULL( Major, Minor, Build, Qfe : Word) : Int64
 Function SimpleXMLEncode( const S : string) : string
 Procedure SimpleXMLDecode( var S : string; TrimBlanks : Boolean)
 Function XMLEncode( const S : string) : string
 Function XMLDecode( const S : string) : string
 Function EntityEncode( const S : string) : string
 Function EntityDecode( const S : string) : string

procedure RIRegister_CPort_Routines(S: TPSExec);
 Procedure EnumComPorts( Ports : TStrings)
 Procedure ListComPorts( Ports : TStrings)
 Procedure ComPorts( Ports : TStrings)  //Alias to Arduino
 Function GetComPorts: TStringlist;
 Function StrToBaudRate( Str : string) : TBaudRate
 Function StrToStopBits( Str : string) : TStopBits
 Function StrToDataBits( Str : string) : TDataBits
 Function StrToParity( Str : string) : TParityBits
 Function StrToFlowControl( Str : string) : TFlowControl
 Function BaudRateToStr( BaudRate : TBaudRate) : string
 Function StopBitsToStr( StopBits : TStopBits) : string
 Function DataBitsToStr( DataBits : TDataBits) : string
 Function ParityToStr( Parity : TParityBits) : string
 Function FlowControlToStr( FlowControl : TFlowControl) : string
 Function ComErrorsToStr( Errors : TComErrors) : String

 Function GetMessage( var lpMsg : TMsg; hWnd : HWND; wMsgFilterMin, wMsgFilterMax : UINT) : BOOL
 Function DispatchMessage( const lpMsg : TMsg) : Longint
 Function TranslateMessage( const lpMsg : TMsg) : BOOL
 Function SetMessageQueue( cMessagesMax : Integer) : BOOL
 Function PeekMessage(var lpMsg:TMsg; hWnd:HWND;wMsgFilterMin,wMsgFilterMax,wRemoveMsg:UINT):BOOL
 Function GetMessagePos : DWORD
 Function GetMessageTime : Longint
 Function GetMessageExtraInfo : Longint
 Function GetSpecialFolderPath( const FolderName : string; CanCreate : Boolean) : string
 Procedure JAddToRecentDocs( const Filename : string)
 Procedure ClearRecentDocs
 Function ExtractIconFromFile( FileName : string; Index : Integer) : HICON
 Function CreateShellLink( const AppName, Desc : string; Dest : string) : string
 Procedure GetShellLinkInfo( const LinkFile : WideString; var SLI : TShellLinkInfo)
 Procedure SetShellLinkInfo( const LinkFile : WideString; const SLI : TShellLinkInfo)
 Function RecycleFile( FileToRecycle : string) : Boolean
 Function JCopyFile( FromFile, ToDir : string) : Boolean
 Function ShellObjectTypeEnumToConst( ShellObjectType : TShellObjectType) : UINT
 Function ShellObjectTypeConstToEnum( ShellObjectType : UINT) : TShellObjectType
 Function QueryServiceConfig2A( hService : SC_HANDLE; dwInfoLevel : DWORD; lpBuffer : LPBYTE; cbBufSize : DWORD; var pcbBytesNeeded : DWORD) : BOOL
 Function QueryServiceConfig2W( hService : SC_HANDLE; dwInfoLevel : DWORD; lpBuffer : LPBYTE; cbBufSize : DWORD; var pcbBytesNeeded : DWORD) : BOOL
 Function QueryServiceConfig2( hService : SC_HANDLE; dwInfoLevel : DWORD; lpBuffer : LPBYTE; cbBufSize : DWORD; var pcbBytesNeeded : DWORD) : BOOL
 Function EnumServicesStatusExA(hSCManager: SC_HANDLE; InfoLevel: SC_ENUM_TYPE; dwServiceType: DWORD; dwServiceState: DWORD;lpServices:LPBYTE;cbBufSize:DWORD;var pcbBytesNeeded,lpServicesReturned, lpResumeHandle:DWORD;pszGroupName: LP
 Function EnumServicesStatusExW( hSCManager : SC_HANDLE; InfoLevel : SC_ENUM_TYPE; dwServiceType : DWORD; dwServiceState : DWORD; lpServices:LPBYTE; cbBufSize:DWORD; var pcbBytesNeeded,lpServicesReturned, lpResumeHandle:DWORD; pszGroupNam
 Function EnumServicesStatusEx( hSCManager : SC_HANDLE; InfoLevel : SC_ENUM_TYPE; dwServiceType : DWORD; dwServiceState : DWORD;lpServices :LPBYTE;cbBufSize:DWORD; var pcbBytesNeeded,lpServicesReturned, lpResumeHandle:DWORD; pszGroupName
 Function ConvertSidToStringSid( sid : PSID; var stringSid : LPWSTR) : BOOL

******************************************* unit uPSI_JclPeImage;

 Function IsValidPeFile( const FileName : TFileName) : Boolean
// Function PeGetNtHeaders( const FileName : TFileName; var NtHeaders : TImageNtHeaders) : Boolean
 Function PeCreateNameHintTable( const FileName : TFileName) : Boolean
 Function PeRebaseImage(const ImageName: TFileName; NewBase : DWORD; TimeStamp : DWORD; MaxNewSize : DWORD) : TJclRebaseImageInfo
 Function PeVerifyCheckSum( const FileName : TFileName) : Boolean
 Function PeClearCheckSum( const FileName : TFileName) : Boolean
 Function PeUpdateCheckSum( const FileName : TFileName) : Boolean
 Function PeDoesExportFunction(const FileName:TFileName;const FuncName:string;Options:TJclSmartCompOptions):Bool;
 Function PeIsExportFunctionForwardedEx( const FileName : TFileName; const FunctionName : string; var ForwardedName : string; Options : TJclSmartCompOptions) : Boolean
 Function PeIsExportFunctionForwarded(const FileName:TFileName;const FunctionName:string;Options:TJclSmartCompOptions):Bool
 Function PeDoesImportFunction( const FileName : TFileName; const FunctionName : string; const LibraryName : string; Options : TJclSmartCompOptions) : Boolean
 Function PeDoesImportLibrary(const FileName:TFileName;const LibraryName:string;Recursive:Boolean):Boolean);
 Function PeImportedLibraries( const FileName : TFileName; const LibrariesList : TStrings; Recursive : Boolean; FullPathName : Boolean) : Boolean
 Function PeImportedFunctions(const FileName:TFileName;const FunctionsList:TStrings;const LibraryName:string; IncludeLibNames : Boolean): Boolean
 Function PeExportedFunctions( const FileName : TFileName; const FunctionsList : TStrings) : Boolean
 Function PeExportedNames( const FileName : TFileName; const FunctionsList : TStrings) : Boolean
 Function PeExportedVariables( const FileName : TFileName; const FunctionsList : TStrings) : Boolean
 Function PeResourceKindNames(const FileN:TFileName;ResourceType:TJclPeResourceKind;const NamesList:TStrings):Bool
 Function PeBorFormNames( const FileName : TFileName; const NamesList : TStrings) : Boolean
 Function PeBorDependedPackages(const FileName:TFileName;PackagesList:TStrings;FullPathName, Descript:Bool):Bool;
 Function PeFindMissingImports( const FileName : TFileName; MissingImportsList : TStrings) : Boolean;
 Function PeFindMissingImports1( RequiredImportsList, MissingImportsList : TStrings) : Boolean;
 Function PeCreateRequiredImportList(const FileName: TFileName; RequiredImportsList: TStrings): Boolean;
 //Function PeMapImgNtHeaders( const BaseAddress : Pointer) : PImageNtHeaders
 //Function PeMapImgLibraryName( const BaseAddress : Pointer) : string
 //Function PeMapImgSections( const NtHeaders : PImageNtHeaders) : PImageSectionHeader
 //Function PeMapImgFindSection( const NtHeaders : PImageNtHeaders; const SectionName : string) : PImageSectionHeader
 //Function PeMapImgExportedVariables(const Module: HMODULE; const VariablesList:TStrings):Boolean
 //Function PeMapImgResolvePackageThunk( Address : Pointer) : Pointer
 Function PeMapFindResource(const Module:HMODULE;const ResourceType:PChar;const ResourceName:string): ___Pointer;
  SIRegister_TJclPeSectionStream(CL);
  SIRegister_TJclPeMapImgHookItem(CL);
  SIRegister_TJclPeMapImgHooks(CL);
 //Function PeDbgImgNtHeaders(ProcessHandle:THandle;BaseAddress:Pointer;var NtHeaders:TImageNtHeaders):Boolean
 //Function PeDbgImgLibraryName(ProcessHandle:THandle; BaseAddress:Pointer; var Name:string):Boolean
  Type TJclBorUmSymbolKind','(skData,skFunction,skConstructor,skDestructor,skRTTI,skVTable)
  TJclBorUmSymbolModifier', '( smQualified, smLinkProc )
  TJclBorUmSymbolModifiers', 'set of TJclBorUmSymbolModifier
  TJclBorUmDescription', 'record Kind : TJclBorUmSymbolKind; Modifiers : TJclBorUmSymbolModifiers; end
  TJclBorUmResult', '( urOk, urNotMangled, urMicrosoft, urError )
  TJclPeUmResult', '( umNotMangled, umBorland, umMicrosoft )
 Function PeBorUnmangleName( const Name : string; var Unmangled : string; var Description : TJclBorUmDescription; var BasePos : Integer) : TJclBorUmResult;
 Function PeBorUnmangleName1(const Name:string;var Unmangled:string;var Descript:TJclBorUmDescription):TJclBorUmResult;
 Function PeBorUnmangleName2( const Name : string; var Unmangled : string) : TJclBorUmResult;
 Function PeBorUnmangleName3( const Name : string) : string;
 Function PeIsNameMangled( const Name : string) : TJclPeUmResult
 Function PeUnmangleName( const Name : string; var Unmangled : string) : TJclPeUmResult


//****************** SysTools uPSI_StSystem; ****************************************
 Function StCopyFile( const SrcPath, DestPath : AnsiString) : Cardinal
 Function CreateTempFile( const aFolder : AnsiString; const aPrefix : AnsiString) : AnsiString
 Function DeleteVolumeLabel( Drive : Char) : Cardinal
 //Procedure EnumerateDirectories(const StartDir:AnsiString;FL:TStrings;SubDirs:Bool;IncludeItem:TIncludeItemFunc);
 //Procedure EnumerateFiles(const StartDir:AnsiString;FL:TStrings;SubDirs:Bool IncludeItem:TIncludeItemFunc);
 Function FileHandlesLeft( MaxHandles : Cardinal) : Cardinal
 Function FileMatchesMask( const FileName, FileMask : AnsiString) : Boolean
 Function FileTimeToStDateTime( FileTime : LongInt) : TStDateTimeRec
 Function FindNthSlash( const Path : AnsiString; n : Integer) : Integer
 Function FlushOsBuffers( Handle : Integer) : Boolean
 Function GetCurrentUser : AnsiString
 Function GetDiskClass( Drive : Char) : DiskClass
 Function GetDiskInfo(Drive:Char;var ClustersAvail,TotalClusters,BytesPerSector, SectorsPerCluster:Cardinal):Bool;
 Function GetDiskSpace(Drive:Char;var UserSpaceAvail:Double;var TotalSpaceAvail:Double;var DiskSize:Double):Bool;
 Function GetDiskSpace(Drive:Char;var UserSpaceAvail:Comp;var TotalSpaceAvail:Comp;var DiskSize:Comp):Boolean;
  { index 0 - FreeBytesAvailable, 1 - TotalNumberOfBytes, 2 - TotalNumberOfFreeBytes  }
 Function getDiskSpace2(const path: String; index: integer): int64;
 Function GetFileCreateDate( const FileName : AnsiString) : TDateTime
 Function StGetFileLastAccess( const FileName : AnsiString) : TDateTime
 Function GetFileLastModify( const FileName : AnsiString) : TDateTime
 Function GetHomeFolder( aForceSlash : Boolean) : AnsiString
 Function GetLongPath( const APath : AnsiString) : AnsiString
 Function GetMachineName : AnsiString
 Function GetMediaID( Drive : Char; var MediaIDRec : MediaIDType) : Cardinal
 Function GetParentFolder( const APath : AnsiString; aForceSlash : Boolean) : AnsiString
 Function GetShortPath( const APath : AnsiString) : AnsiString
 Function GetSystemFolder( aForceSlash : Boolean) : AnsiString
 Function GetTempFolder( aForceSlash : boolean) : AnsiString
 Function StGetWindowsFolder( aForceSlash : boolean) : AnsiString
 Function GetWorkingFolder( aForceSlash : boolean) : AnsiString
 Function GlobalDateTimeToLocal( const UTC : TStDateTimeRec; MinOffset : Integer) : TStDateTimeRec
 Function StIsDirectory( const DirName : AnsiString) : Boolean
 Function IsDirectoryEmpty( const S : AnsiString) : Integer
 Function IsDriveReady( Drive : Char) : Boolean
 Function IsFile( const FileName : AnsiString) : Boolean
 Function IsFileArchive( const S : AnsiString) : Integer
 Function IsFileHidden( const S : AnsiString) : Integer
 Function IsFileReadOnly( const S : AnsiString) : Integer
 Function IsFileSystem( const S : AnsiString) : Integer
 Function LocalDateTimeToGlobal( const DT1 : TStDateTimeRec; MinOffset : Integer) : TStDateTimeRec
 Function ReadVolumeLabel( var VolName : AnsiString; Drive : Char) : Cardinal
 Function SameFile( const FilePath1, FilePath2 : AnsiString; var ErrorCode : Integer) : Boolean
 Function SetMediaID( Drive : Char; var MediaIDRec : MediaIDType) : Cardinal
 Procedure SplitPath( const APath : AnsiString; Parts : TStrings)
 Function StDateTimeToFileTime( const FileTime : TStDateTimeRec) : LongInt
 Function StDateTimeToUnixTime( const DT1 : TStDateTimeRec) : Longint
 Function UnixTimeToStDateTime( UnixTime : Longint) : TStDateTimeRec
 Function ValidDrive( Drive : Char) : Boolean
 Function WriteVolumeLabel( const VolName : AnsiString; Drive : Char) : Cardinal

//****************************unit uPSI_JclLANMan;******************************************
Function CreateAccount( const Server, Username, Fullname, Password, Description, Homedir, Script : string; const PasswordNeverExpires : Boolean) : Boolean
 Function CreateLocalAccount( const Username, Fullname, Password, Description, Homedir, Script : string; const PasswordNeverExpires : Boolean) : Boolean
 Function DeleteAccount( const Servername, Username : string) : Boolean
 Function DeleteLocalAccount( Username : string) : Boolean
 Function CreateLocalGroup( const Server, Groupname, Description : string) : Boolean
 Function CreateGlobalGroup( const Server, Groupname, Description : string) : Boolean
 Function DeleteLocalGroup( const Server, Groupname : string) : Boolean
 Function GetLocalGroups( const Server : string; const Groups : TStrings) : Boolean
 Function GetGlobalGroups( const Server : string; const Groups : TStrings) : Boolean
 Function LocalGroupExists( const Group : string) : Boolean
 Function GlobalGroupExists( const Server, Group : string) : Boolean
 Function AddAccountToLocalGroup( const Accountname, Groupname : string) : Boolean
 Function LookupGroupName( const Server : string; const RID : TNetWellKnownRID) : string
 Procedure ParseAccountName( const QualifiedName : string; var Domain, UserName : string)
 Function IsLocalAccount( const AccountName : string) : Boolean
 Function TimeStampInterval( StartStamp, EndStamp : TDateTime) : integer
 Function GetRandomString( NumChar : cardinal) : string

//****************************unit uPSI_cUtils;******************************************
TypeS('TUnitType', '( utSrc, utHead, utRes, utPrj, utOther )
Function cIsWinNT : boolean
 Procedure cFilesFromWildcard(Directory,Mask: string;var Files:TStringList;Subdirs,ShowDirs, Multitasking:Boolean;
 Function cExecuteFile( const FileName, Params, DefaultDir : string; ShowCmd : Integer) : THandle
 Function cRunAndGetOutput(Cmd,WorkDir:string; ErrFunc:TErrFunc; LineOutputFunc:TLineOutputFunc; CheckAbortFunc : TCheckAbortFunc; ShowReturnValue : Boolean) : string
 Function cGetShortName( FileName : string) : string
 Procedure cShowError( Msg : String)
 Function cCommaStrToStr( s : string; formatstr : string) : string
 Function cIncludeQuoteIfSpaces( s : string) : string
 Function cIncludeQuoteIfNeeded( s : string) : string
 Procedure cLoadFilefromResource( const FileName : string; ms : TMemoryStream)
 Function cValidateFile(const FileName:string; const WorkPath:string;const CheckDirs:boolean):string;
 Function cBuildFilter( var value : string; const FLTStyle : TFILTERSET) : boolean;
 Function cBuildFilter1( var value : string; const _filters : array of string) : boolean;
 Function cCodeInstoStr( s : string) : string
 Function cStrtoCodeIns( s : string) : string
 Procedure cStrtoAttr( var Attr : TSynHighlighterAttributes; Value : string)
 Function cAttrtoStr( const Attr : TSynHighlighterAttributes) : string
 Procedure cStrtoPoint( var pt : TPoint; value : string)
 Function cPointtoStr( const pt : TPoint) : string
 Function cListtoStr( const List : TStrings) : string
 Function ListtoStr( const List : TStrings) : string
 Procedure StrtoList( s : string; const List : TStrings; const delimiter : char)
 Procedure cStrtoList( s : string; const List : TStrings; const delimiter : char)
 Function cGetFileTyp( const FileName : string) : TUnitType
 Function cGetExTyp( const FileName : string) : TExUnitType
 Procedure cSetPath( Add : string; const UseOriginal : boolean)
 Function cExpandFileto( const FileName : string; const BasePath : string) : string
 Function cFileSamePath( const FileName : string; const TestPath : string) : boolean
 Procedure cCloneMenu( const FromMenu : TMenuItem; ToMenu : TMenuItem)
 Function cGetLastPos( const SubStr : string; const S : string) : integer
 Function cGenMakePath( FileName : String) : String;
 Function cGenMakePath2( FileName : String) : String
 Function cGenMakePath1( FileName : String; EscapeSpaces, EncloseInQuotes : Boolean) : String;
 Function cGetRealPath( BrokenFileName : String; Directory : String) : String
 Function cCalcMod( Count : Integer) : Integer
 Function cGetVersionString( FileName : string) : string
 Function cCheckChangeDir( var Dir : string) : boolean
 Function cGetAssociatedProgram(const Extension:string; var Filename,Description: string):boolean
 Function cIsNumeric( s : string) : boolean
 Procedure StrtoAttr( var Attr : TSynHighlighterAttributes; Value : string)
 Function AttrtoStr( const Attr : TSynHighlighterAttributes) : string
 Function GetFileTyp( const FileName : string) : TUnitType
 Function Atoi(const aStr: string): integer
 Function Itoa(const aint: integer): string


procedure SIRegister_cHTTPUtils(CL: TPSPascalCompiler);
begin
  FindClass('TOBJECT'),'EHTTP
  FindClass('TOBJECT'),'EHTTPParser
  //AnsiCharSet', 'set of AnsiChar
  AnsiStringArray', 'array of AnsiString
  THTTPProtocolEnum', '( hpNone, hpCustom, hpHTTP, hpHTTPS )
  THTTPVersionEnum', '( hvNone, hvCustom, hvHTTP10, hvHTTP11 )
  THTTPVersion', 'record Version : THTTPVersionEnum; Protocol : TH'
   +'TTPProtocolEnum; CustomProtocol : AnsiString; CustomMajVersion : Integer; '
   +'CustomMinVersion : Integer; end
  THTTPHeaderNameEnum', '( hntCustom, hntHost, hntContentType, hnt'
   +'ContentLength, hntContentTransferEncoding, hntContentLocation, hntContentL'
   +'anguage, hntContentEncoding, hntTransferEncoding, hntDate, hntServer, hntU'
   +'serAgent, hntLocation, hntConnection, hntExpires, hntCacheControl, hntSetC'
   +'ookie, hntCookie, hntAuthorization, hntVia, hntWarning, hntContentRange, h'
   +'ntXForwardedFor, hntPragma, hntXPoweredBy, hntWWWAuthenticate, hntLastModi'
   +'fied, hntETag, hntProxyAuthorization, hntReferer, hntAge, hntAcceptRanges,'
   +' hntAcceptEncoding, hntAcceptLanguage, hntAcceptCharset, hntIfModifiedSinc'
   +'e, hntIfUnmodifiedSince, hntRetryAfter, hntUpgrade, hntStatus, hntProxyCon'
   +'nection, hntOrigin, hntKeepAlive )
  THTTPHeaderName', 'record Value : THTTPHeaderNameEnum; Custom: AnsiString; end
  THTTPCustomHeader', 'record FieldName : AnsiString; FieldValue :'
   +' AnsiString; end
  //PHTTPCustomHeader', '^THTTPCustomHeader // will not work
  THTTPContentLengthEnum', '( hcltNone, hcltByteCount )
  THTTPContentLength', 'record Value : THTTPContentLengthEnum; ByteCount:Int64; end
  //PHTTPContentLength', '^THTTPContentLength // will not work
  THTTPContentTypeMajor', '( hctmCustom, hctmText, hctmImage )
  THTTPContentTypeEnum', '( hctNone, hctCustomParts, hctCustomStri'
   +'ng, hctTextHtml, hctTextAscii, hctTextCss, hctTextPlain, hctTextXml, hctTe'
   +'xtCustom, hctImageJpeg, hctImagePng, hctImageGif, hctImageCustom, hctAppli'
   +'cationJSON, hctApplicationOctetStream, hctApplicationJavaScript, hctApplic'
   +'ationCustom, hctAudioCustom, hctVideoCustom )
  THTTPContentType', 'record Value : THTTPContentTypeEnum; CustomM'
   +'ajor : AnsiString; CustomMinor : AnsiString; Parameters : AnsiStringArray;'
   +' CustomStr : AnsiString; end
  THTTPDateFieldEnum', '( hdNone, hdCustom, hdParts, hdDateTime )
  THTTPDateField', 'record Value : THTTPDateFieldEnum; DayOfWeek :'
   +' Integer; Day : integer; Month : integer; Year : Integer; Hour : integer; '
   +'Min : integer; Sec : Integer; TimeZoneGMT : Boolean; CustomTimeZone : Ansi'
   +'String; DateTime : TDateTime; Custom : AnsiString; end
  THTTPTransferEncodingEnum', '( hteNone, hteCustom, hteChunked )
  THTTPTransferEncoding', 'record Value : THTTPTransferEncodingEnu'
   +'m; Custom : AnsiString; end
  THTTPConnectionFieldEnum', '( hcfNone, hcfCustom, hcfClose, hcfKeepAlive )
  THTTPConnectionField', 'record Value : THTTPConnectionFieldEnum;'
   +' Custom : AnsiString; end
  THTTPAgeFieldEnum', '( hafNone, hafCustom, hafAge )
  THTTPAgeField', 'record Value: THTTPAgeFieldEnum; Age : Int64;Custom:AnsiString; end
  THTTPCacheControlFieldEnum', '( hccfNone, hccfDecoded, hccfCustom )
  THTTPCacheControlRequestSubField', '( hccsfNoCache, hccsfNoStore'
   +', hccsfMaxAge, hccsfMaxStale, hccsfMinFresh, hccsfNoTransform, hccsfOnlyIfCached )
  THTTPCacheControlResponseSubField', '( hccrfPublic, hccrfPrivate'
   +', hccrfNoCache, hccrfNoStore, hccrfNoTransform, hccrfMustRevalidate, hccrf'
   +'ProxyRevalidate, hccrfMaxAge, hccrfSMaxAge )
  THTTPCacheControlField', 'record Value : THTTPCacheControlFieldEnum; end
  THTTPContentEncodingEnum', '( hceNone, hceCustom, hceIdentity, h'
   +'ceCompress, hceDeflate, hceExi, hceGzip, hcePack200Gzip )
  THTTPContentEncoding', 'record Value:THTTPContentEncodingEnum;Custom:AnsiString; end;
  THTTPContentEncodingFieldEnum', '( hcefNone, hcefList )
  THTTPContentEncodingField', 'record Value : THTTPContentEncoding'
   +'FieldEnum; List : array of THTTPContentEncoding; end
  THTTPRetryAfterFieldEnum', '( hrafNone, hrafCustom, harfDate, harfSeconds )
  THTTPRetryAfterField', 'record Value : THTTPRetryAfterFieldEnum;'
   +' Custom : AnsiString; Date : TDateTime; Seconds : Int64; end
  THTTPContentRangeFieldEnum', '( hcrfNone, hcrfCustom, hcrfByteRange )
  THTTPContentRangeField', 'record Value : THTTPContentRangeFieldE'
   +'num; ByteFirst : Int64; ByteLast : Int64; ByteSize : Int64; Custom : AnsiString; end
  THTTPSetCookieFieldEnum', '( hscoNone, hscoDecoded, hscoCustom )
  THTTPSetCookieCustomField', 'record Name : AnsiString; Value : AnsiString; end
  THTTPSetCookieCustomFieldArray', 'array of THTTPSetCookieCustomField
  THTTPSetCookieField', 'record Value : THTTPSetCookieFieldEnum; D'
   +'omain : AnsiString; Path : AnsiString; Expires : THTTPDateField; MaxAge : '
   +'Int64; HttpOnly : Boolean; Secure : Boolean; CustomFields : THTTPSetCookie'
   +'CustomFieldArray; Custom : AnsiString; end
  //PHTTPSetCookieField', '^THTTPSetCookieField // will not work
  THTTPSetCookieFieldArray', 'array of THTTPSetCookieField
  THTTPCookieFieldEnum', '( hcoNone, hcoDecoded, hcoCustom )
  THTTPCookieFieldEntry', 'record Name : AnsiString; Value : AnsiString; end
  //PHTTPCookieFieldEntry', '^THTTPCookieFieldEntry // will not work
  THTTPCookieFieldEntryArray', 'array of THTTPCookieFieldEntry
  THTTPCookieField', 'record Value : THTTPCookieFieldEnum; Entries'
   +' : THTTPCookieFieldEntryArray; Custom : AnsiString; end
  THTTPCommonHeaders', 'record TransferEncoding : THTTPTransferEnc'
   +'oding; ContentType : THTTPContentType; ContentLength : THTTPContentLength;'
   +' Connection : THTTPConnectionField; ProxyConnection : THTTPConnectionField'
   +'; Date : THTTPDateField; ContentEncoding : THTTPContentEncodingField; end
  THTTPCustomHeaders', 'array of THTTPCustomHeader
   //THTTPFixedHeaders','array[THTTPHeaderNameEnum] of AnsiString
  THTTPFixedHeaders','array[0..42] of AnsiString
   THTTPMethodEnum', '( hmNone, hmCustom, hmGET, hmPUT, hmPOST, hmC'
   +'ONNECT, hmHEAD, hmDELETE, hmOPTIONS, hmTRACE )
  THTTPMethod', 'record Value : THTTPMethodEnum; Custom : AnsiString; end
  THTTPRequestStartLine','record Method: THTTPMethod;URI: AnsiString;Version:THTTPVersion; end
  THTTPRequestHeader', 'record CommonHeaders : THTTPCommonHeaders;'
   +' FixedHeaders : THTTPFixedHeaders; CustomHeaders : THTTPCustomHeaders; Coo'
   +'kie : THTTPCookieField; IfModifiedSince:THTTPDateField;IfUnmodifiedSince:THTTPDateField;end
  //PHTTPRequestHeader', '^THTTPRequestHeader // will not work
  THTTPRequest', 'record StartLine : THTTPRequestStartLine; Header'
   +' : THTTPRequestHeader; HeaderComplete : Boolean; HasContent : Boolean; end
  THTTPResponseStartLineMessage', '( hslmNone, hslmCustom, hslmOK)
  THTTPResponseStartLine', 'record Version : THTTPVersion; Code : '
   +'Integer; Msg : THTTPResponseStartLineMessage; CustomMsg : AnsiString; end
  THTTPResponseHeader', 'record CommonHeaders : THTTPCommonHeaders'
   +'; FixedHeaders : THTTPFixedHeaders; CustomHeaders : THTTPCustomHeaders; Co'
   +'okies : THTTPSetCookieFieldArray; Expires : THTTPDateField; LastModified :'
   +' THTTPDateField; Age : THTTPAgeField; end
  //PHTTPResponseHeader', '^THTTPResponseHeader // will not work
  THTTPResponse', 'record StartLine : THTTPResponseStartLine; Head'
   +'er : THTTPResponseHeader; HeaderComplete : Boolean; HasContent : Boolean; end
 Function HTTPMessageHasContent( const H : THTTPCommonHeaders) : Boolean
 Procedure InitHTTPRequest( var A : THTTPRequest)
 Procedure InitHTTPResponse( var A : THTTPResponse)
 Procedure ClearHTTPVersion( var A : THTTPVersion)
 Procedure ClearHTTPContentLength( var A : THTTPContentLength)
 Procedure ClearHTTPContentType( var A : THTTPContentType)
 Procedure ClearHTTPDateField( var A : THTTPDateField)
 Procedure ClearHTTPTransferEncoding( var A : THTTPTransferEncoding)
 Procedure ClearHTTPConnectionField( var A : THTTPConnectionField)
 Procedure ClearHTTPAgeField( var A : THTTPAgeField)
 Procedure ClearHTTPContentEncoding( var A : THTTPContentEncoding)
 Procedure ClearHTTPContentEncodingField( var A : THTTPContentEncodingField)
 Procedure ClearHTTPContentRangeField( var A : THTTPContentRangeField)
 Procedure ClearHTTPSetCookieField( var A : THTTPSetCookieField)
 Procedure ClearHTTPCommonHeaders( var A : THTTPCommonHeaders)
 //Procedure ClearHTTPFixedHeaders( var A : THTTPFixedHeaders)
 Procedure ClearHTTPCustomHeaders( var A : THTTPCustomHeaders)
 Procedure ClearHTTPCookieField( var A : THTTPCookieField)
 Procedure ClearHTTPMethod( var A : THTTPMethod)
 Procedure ClearHTTPRequestStartLine( var A : THTTPRequestStartLine)
 Procedure ClearHTTPRequestHeader( var A : THTTPRequestHeader)
 Procedure ClearHTTPRequest( var A : THTTPRequest)
 Procedure ClearHTTPResponseStartLine( var A : THTTPResponseStartLine)
 Procedure ClearHTTPResponseHeader( var A : THTTPResponseHeader)
 Procedure ClearHTTPResponse( var A : THTTPResponse)
  THTTPStringOption', '( hsoNone )
  THTTPStringOptions', 'set of THTTPStringOption
  FindClass('TOBJECT'),'TAnsiStringBuilder

 Procedure BuildStrHTTPVersion(const A:THTTPVersion; const B:TAnsiStringBuilder;const P:THTTPStringOptions);
 Procedure BuildStrHTTPContentLengthValue(const A: THTTPContentLength; const B : TAnsiStringBuilder; const P : THTTPStringOptions)
 Procedure BuildStrHTTPContentLength( const A : THTTPContentLength; const B : TAnsiStringBuilder; const P : THTTPStringOptions)
 Procedure BuildStrHTTPContentTypeValue( const A : THTTPContentType; const B : TAnsiStringBuilder; const P : THTTPStringOptions)
 Procedure BuildStrHTTPContentType(const A:THTTPContentType;const B:TAnsiStringBuilder; const P:THTTPStringOptions)
 Procedure BuildStrRFCDateTime( const DOW, Da, Mo, Ye, Ho, Mi, Se : Integer; const TZ : AnsiString; const B : TAnsiStringBuilder; const P : THTTPStringOptions)
 Procedure BuildStrHTTPDateFieldValue( const A : THTTPDateField; const B : TAnsiStringBuilder; const P : THTTPStringOptions)
 Procedure BuildStrHTTPDateField(const A:THTTPDateField;const B:TAnsiStringBuilder;const P:THTTPStringOptions);
 Procedure BuildStrHTTPTransferEncodingValue( const A : THTTPTransferEncoding; const B : TAnsiStringBuilder; const P : THTTPStringOptions)
 Procedure BuildStrHTTPTransferEncoding( const A : THTTPTransferEncoding; const B : TAnsiStringBuilder; const P : THTTPStringOptions)
 Procedure BuildStrHTTPContentRangeField( const A : THTTPContentRangeField; const B : TAnsiStringBuilder; const P : THTTPStringOptions)
 Procedure BuildStrHTTPConnectionFieldValue( const A : THTTPConnectionField; const B : TAnsiStringBuilder; const P : THTTPStringOptions)
 Procedure BuildStrHTTPConnectionField( const A : THTTPConnectionField; const B : TAnsiStringBuilder; const P : THTTPStringOptions)
 Procedure BuildStrHTTPAgeField(const A:THTTPAgeField;const B:TAnsiStringBuilder;const P:THTTPStringOptions);
 Procedure BuildStrHTTPContentEncoding( const A : THTTPContentEncoding; const B : TAnsiStringBuilder; const P : THTTPStringOptions)
 Procedure BuildStrHTTPContentEncodingField(const A:THTTPContentEncodingField;const B:TAnsiStringBuilder;const P:THTTPStringOptions)
 Procedure BuildStrHTTPProxyConnectionField(const A : THTTPConnectionField; const B : TAnsiStringBuilder; const P : THTTPStringOptions)
 Procedure BuildStrHTTPCommonHeaders( const A : THTTPCommonHeaders; const B : TAnsiStringBuilder; const P : THTTPStringOptions)
 Procedure BuildStrHTTPFixedHeaders(const A:THTTPFixedHeaders;const B:TAnsiStrBuilder;const P:THTTPStringOptions)
 Procedure BuildStrHTTPCustomHeaders( const A : THTTPCustomHeaders; const B : TAnsiStringBuilder; const P : THTTPStringOptions)
 Procedure BuildStrHTTPSetCookieFieldValue( const A : THTTPSetCookieField; const B : TAnsiStringBuilder; const P : THTTPStringOptions)
 Procedure BuildStrHTTPCookieFieldValue( const A : THTTPCookieField; const B : TAnsiStringBuilder; const P : THTTPStringOptions)
 Procedure BuildStrHTTPCookieField(const A:THTTPCookieField;const B:TAnsiStringBuilder;const P:THTTPStringOptions);
 Procedure BuildStrHTTPMethod( const A : THTTPMethod; const B : TAnsiStringBuilder; const P : THTTPStringOptions)
 Procedure BuildStrHTTPRequestStartLine( const A : THTTPRequestStartLine; const B : TAnsiStringBuilder; const P : THTTPStringOptions)
 Procedure BuildStrHTTPRequestHeader(const A:THTTPRequestHeader;const B:TAnsiStringBuilder;const
P:THTTPStringOptions);
 Procedure BuildStrHTTPRequest( const A : THTTPRequest; const B : TAnsiStringBuilder; const P :
THTTPStringOptions)
 Procedure BuildStrHTTPResponseCookieFieldArray( const A : THTTPSetCookieFieldArray; const B :
TAnsiStringBuilder; const P : THTTPStringOptions)
 Procedure BuildStrHTTPResponseStartLine(const A:THTTPResponseStartLine;const B:TAnsiStrBldr;const P
THTTPStrOptions);
 Procedure BuildStrHTTPResponseHeader(const A:THTTPRespHeader;const B:TAnsiStrBuilder;const
P:THTTPStringOptions);
 Procedure BuildStrHTTPResponse(const A:THTTPResponse; const B:TAnsiStringBuilder; const
P:THTTPStringOptions);
 Function HTTPContentTypeValueToStr( const A : THTTPContentType) : AnsiString
 Function HTTPSetCookieFieldValueToStr( const A : THTTPSetCookieField) : AnsiString
 Function HTTPCookieFieldValueToStr( const A : THTTPCookieField) : AnsiString
 Function HTTPMethodToStr( const A : THTTPMethod) : AnsiString
 Function HTTPRequestToStr( const A : THTTPRequest) : AnsiString
 Function HTTPResponseToStr( const A : THTTPResponse) : AnsiString
 Procedure PrepareCookie(var A:THTTPCookieField;const B:THTTPSetCookieFieldArray;const
Domain:AnsiString;const Secure:Bool; THTTPParserHeaderParseFunc', 'Function ( const HeaderName : THTT'
   +PHeaderNameEnum; const HeaderPtr : ___Pointer) : Boolean
  SIRegister_THTTPParser(CL);
  FindClass('TOBJECT'),'THTTPContentDecoder
  THTTPContentDecoderProc', 'Procedure ( const Sender : THTTPContentDecoder)
  THTTPContentDecoderContentType', '( crctFixedSize, crctChunked, crctUnsized )
  THTTPContentDecoderChunkState', '( crcsChunkHeader, crcsContent,'
   +' crcsContentCRLF, crcsTrailer, crcsFinished )
  THTTPContentDecoderLogEvent', 'Procedure ( const Sender : THTTPContentDecoder; const LogMsg : String)
  SIRegister_THTTPContentDecoder(CL);
  THTTPContentReaderMechanism', '( hcrmEvent, hcrmString, hcrmStream, hcrmFile )
  FindClass('TOBJECT'),'THTTPContentReader
  THTTPContentReaderProc', 'Procedure ( const Sender : THTTPContentReader)
  THTTPContentReaderLogEvent','Procedure(const Sender:THTTPContentReader;const LogMsg:String;const
LogLevel:Int;
  SIRegister_THTTPContentReader(CL);
  THTTPContentWriterMechanism','(hctmEvent, hctmString, hctmStream, hctmFile )
  FindClass('TOBJECT'),'THTTPContentWriter
  THTTPContentWriterLogEvent', 'Procedure (const Sender : THTTPContentWriter;const LogMsg:AnsiString);
  SIRegister_THTTPContentWriter(CL);
 Procedure SelfTestcHTTPUtils
end;

(*----------------------------------------------------------------------------*)
procedure SIRegister_cTLSUtils(CL: TPSPascalCompiler);
begin
 'TLSLibraryVersion','String '1.00
 'TLSError_None','LongInt'( 0);
 'TLSError_InvalidBuffer','LongInt'( 1);
 'TLSError_InvalidParameter','LongInt'( 2);
 'TLSError_InvalidCertificate','LongInt'( 3);
 'TLSError_InvalidState','LongInt'( 4);
 'TLSError_DecodeError','LongInt'( 5);
 'TLSError_BadProtocol','LongInt'( 6);
 Function TLSErrorMessage( const TLSError : Integer) : String
  SIRegister_ETLSError(CL);
  TTLSProtocolVersion', 'record major : Byte; minor : Byte; end
  PTLSProtocolVersion', '^TTLSProtocolVersion // will not work
 Procedure InitSSLProtocolVersion30( var A : TTLSProtocolVersion)
 Procedure InitTLSProtocolVersion10( var A : TTLSProtocolVersion)
 Procedure InitTLSProtocolVersion11( var A : TTLSProtocolVersion)
 Procedure InitTLSProtocolVersion12( var A : TTLSProtocolVersion)
 Function IsTLSProtocolVersion( const A, B : TTLSProtocolVersion) : Boolean
 Function IsSSL2( const A : TTLSProtocolVersion) : Boolean
 Function IsSSL3( const A : TTLSProtocolVersion) : Boolean
 Function IsTLS10( const A : TTLSProtocolVersion) : Boolean
 Function IsTLS11( const A : TTLSProtocolVersion) : Boolean
 Function IsTLS12( const A : TTLSProtocolVersion) : Boolean
 Function IsTLS10OrLater( const A : TTLSProtocolVersion) : Boolean
 Function IsTLS11OrLater( const A : TTLSProtocolVersion) : Boolean
 Function IsTLS12OrLater( const A : TTLSProtocolVersion) : Boolean
 Function IsFutureTLSVersion( const A : TTLSProtocolVersion) : Boolean
 Function IsKnownTLSVersion( const A : TTLSProtocolVersion) : Boolean
 Function TLSProtocolVersionToStr( const A : TTLSProtocolVersion) : String
 Function TLSProtocolVersionName( const A : TTLSProtocolVersion) : String
  PTLSRandom', '^TTLSRandom // will not work
 Procedure InitTLSRandom( var Random : TTLSRandom)
 Function TLSRandomToStr( const Random : TTLSRandom) : AnsiString
 'TLSSessionIDMaxLen','LongInt'( 32);
 Procedure InitTLSSessionID( var SessionID : TTLSSessionID; const A : AnsiString)
 Function EncodeTLSSessionID(var Buffer:string;const Size:Int;const SessionID:TTLSSessionID):Int;
 Function DecodeTLSSessionID(const Buffer:string;const Size:Int;var SessionID:TTLSSessionID):Int;
  TTLSSignatureAndHashAlgorithm', 'record Hash : TTLSHashAlgorithm'
   +'; Signature : TTLSSignatureAlgorithm; end
 // PTLSSignatureAndHashAlgorithm', '^TTLSSignatureAndHashAlgorithm +'// will not work
  TTLSSignatureAndHashAlgorithmArray', 'array of TTLSSignatureAndHashAlgorithm
  TTLSKeyExchangeAlgorithm', '( tlskeaNone, tlskeaNULL, tlskeaDHE_'
   +'DSS, tlskeaDHE_RSA, tlskeaDH_Anon, tlskeaRSA, tlskeaDH_DSS, tlskeaDH_RSA )
  TTLSMACAlgorithm', '( tlsmaNone, tlsmaNULL, tlsmaHMAC_MD5, tlsma'
   +'HMAC_SHA1, tlsmaHMAC_SHA256, tlsmaHMAC_SHA384, tlsmaHMAC_SHA512 )
  TTLSMacAlgorithmInfo', 'record Name : AnsiString; DigestSize : I'
   +'nteger; Supported : Boolean; end
  PTLSMacAlgorithmInfo', '^TTLSMacAlgorithmInfo // will not work
 'TLS_MAC_MAXDIGESTSIZE','LongInt'( 64);
  TTLSPRFAlgorithm', '( tlspaSHA256 )
 Function tlsP_MD5( const Secret, Seed : AnsiString; const Size : Integer) : AnsiString
 Function tlsP_SHA1( const Secret, Seed : AnsiString; const Size : Integer) : AnsiString
 Function tlsP_SHA256( const Secret, Seed : AnsiString; const Size : Integer) : AnsiString
 Function tlsP_SHA512( const Secret, Seed : AnsiString; const Size : Integer) : AnsiString
 Function tls10PRF( const Secret, ALabel, Seed : AnsiString; const Size : Integer) : AnsiString
 Function tls12PRF_SHA256( const Secret, ALabel, Seed : AnsiString; const Size : Integer) : AnsiString
 Function tls12PRF_SHA512( const Secret, ALabel, Seed : AnsiString; const Size : Integer) : AnsiString
 Function TLSPRF(const ProtoVersion:TTLSProtocolVersion;const Secret,ALabel,Seed:AString;const
Size:Int):AString;
 Function tls10KeyBlock(const MasterSecret, ServerRandom,ClientRandom:AnsiString; const
Size:Integer):AnsiString
 Function tls12SHA256KeyBlock(const MasterSecret,ServerRandom,ClientRandom: AnsiString;const
Size:Int):AnsiString;
 Function tls12SHA512KeyBlock(const MasterSecret,ServerRandom,ClientRandom: AnsiString;const
Size:Int):AnsiString;
 Function TLSKeyBlock(const ProtocolVersion:TTLSProtocolVersion; const MasterSecret, ServerRandom,
ClientRandom : AnsiString; const Size : Integer) : AnsiString
 Function tls10MasterSecret(const PreMasterSecret,ClientRandom, ServerRandom:AnsiString) :AnsiString;
 Function tls12SHA256MasterSecret(const PreMasterSecret,ClientRandom,ServerRandom:AnsiString):AnsiString;
 Function tls12SHA512MasterSecret(const PreMasterSecret,ClientRandom,ServerRandom:AnsiString): AnsiString;
 Function TLSMasterSecret( const ProtocolVersion: TTLSProtocolVersion;const PreMasterSecret,ClientRandom,
ServerRandom:AnsiString) : AnsiString
  TTLSKeys', 'record KeyBlock : AnsiString; ClientMACKey : AnsiStr'
   +'ing; ServerMACKey : AnsiString; ClientEncKey : AnsiString; ServerEncKey : '
   +'AnsiString; ClientIV : AnsiString; ServerIV : AnsiString; end
 Procedure GenerateTLSKeys( const ProtocolVersion : TTLSProtocolVersion; const MACKeyBits, CipherKeyBits,
IVBits : Integer; const MasterSecret, ServerRandom, ClientRandom : AnsiString; var TLSKeys : TTLSKeys)
 Procedure GenerateFinalTLSKeys(const ProtocolVersion:TTLSProtocolVersion;const IsExportable:Boolean;const
ExpandedKeyBits:Integer;const ServerRandom,ClientRandom:AnsiString;var TLSKeys:TTLSKeys)
 'TLS_PLAINTEXT_FRAGMENT_MAXSIZE','LongInt'( 16384 - 1);
 'TLS_COMPRESSED_FRAGMENT_MAXSIZE','LongInt'( 16384 + 1024);
 Procedure SelfTestcTLSUtils
end;

(*----------------------------------------------------------------------------*)
procedure SIRegister_Reversi(CL: TPSPascalCompiler);
begin
  sPosData','record corner : boolean; square2x2 : boolean; edge:boolean; stable : integer; internal :
integer; disks : integer; mx : integer; my : integer; end
 // pBoard', '^tBoard // will not work
 Function rCalculateData( cc : byte; cx, cy : integer) : sPosData
 Function rCheckMove( color : byte; cx, cy : integer) : integer
 //Function rDoStep( data : pBoard) : word
 Function winExecAndWait( const sAppPath : string; wVisible : word) : boolean
end;

procedure SIRegister_IWDBCommon(CL: TPSPascalCompiler);
begin
Function InEditMode( ADataset : TDataset) : Boolean
Function CheckDataSource( ADataSource : TDataSource) : Boolean;
Function CheckDataSource1(ADataSource:TDataSource;const AFieldName:string;var VField:TField):boolean;
Function GetFieldText( AField : TField) : String
end;

procedure SIRegister_SortGrid(CL: TPSPascalCompiler);
begin
  TPrintMode', '( pmPrint, pmPreview, pmPageCount )
  TMyPrintRange', '( prAll, prSelected )
  TSortStyle', '( ssAutomatic, ssNormal, ssNumeric, ssNumericExten'
   +'ded, ssDateTime, ssTime, ssCustom )
  TSortDirection', '( sdAscending, sdDescending )
  TSetChecked', 'Procedure ( Sender : TObject; ACol, ARow : integer; State : Boolean)
  TGetCombobox', 'Procedure ( Sender : TObject; ACol, ARow : integ'
   +'er; var Strs : TStringList; var Width, Height : integer; var Sorted : Boolean)
  TSetCombobox', 'Procedure ( Sender : TObject; ACol, ARow : integer; Str : String)
  TSetEllipsis', 'Procedure ( Sender : TObject; ACol, ARow : integer)
  SIRegister_TSortOptions(CL);
  SIRegister_TPrintOptions(CL);
  TSortedListEntry', 'record Str : String; RowNum : integer; SortOption : TSortOptions; end
  SIRegister_TSortedList(CL);
  TCellBevelStyle', '( cbNone, cbRaised, cbLowered )
  TCellBevel', 'record Style: TCellBevelStyle; UpperLeftColor: TColor; LowerRightColor : TColor; end
  TVertAlignment', '( taTopJustify, taBottomJustify, taMiddle )
  TFormatOptions', 'record Brush : TBrush; Font : TFont; Alignment'
   +'Horz : TAlignment; AlignmentVert : TVertAlignment; Bevel : TCellBevel; HideText : Boolean; end
  SIRegister_TFontSetting(CL);
  SIRegister_TFontList(CL);
  AddTypeS(TFormatDrawCellEvent', 'Procedure ( Sender : TObject; Col, Row :'
   + integer;State:TGridDrawState;var FormatOptions:TFormatOptions;var CheckBox,Combobox,Ellipsis:Bool);
  TSetFilterEvent', 'Procedure ( ARows : TStrings; var Accept : Boolean)
  TSearchEvent', 'Procedure ( ARows : TStrings; var Accept : Boolean)
  TUpdateGridEvent', 'Procedure ( Sender : TObject; ARow : integer)
  TSizeChangedEvent', 'Procedure ( Sender : TObject; OldColCount, OldRowCount : integer)
  TBeginSortEvent', 'Procedure ( Sender : TObject; var Col : integer)
  TEndSortEvent', 'Procedure ( Sender : TObject; Col : integer)
  TGetSortStyleEvent', 'Procedure ( Sender : TObject; Col : intege'
   +'r; var SortStyle : TSortStyle)
  TCellValidateEvent', 'Procedure ( Sender : TObject; ACol, ARow :'
   +' integer; const OldValue : string; var NewValue : String; var Valid : Boolean)
  SIRegister_TSortGrid(CL);
 Function ExtendedCompare( const Str1, Str2 : String) : Integer
 Function NormalCompare( const Str1, Str2 : String) : Integer
 Function DateTimeCompare( const Str1, Str2 : String) : Integer
 Function NumericCompare( const Str1, Str2 : String) : Integer
 Function TimeCompare( const Str1, Str2 : String) : Integer
 //Function Compare( Item1, Item2 : Pointer) : Integer
end;

************************************ procedure Register_IB(CL: TPSPascalCompiler);
 Procedure IBAlloc( var P, OldSize, NewSize : Integer)
 Procedure IBError( ErrMess : TIBClientError; const Args : array of const)
 Procedure IBDataBaseError
 Function StatusVector : PISC_STATUS
 Function StatusVectorArray : PStatusVector
 Function CheckStatusVector( ErrorCodes : array of ISC_STATUS) : Boolean
 Function StatusVectorAsText : string
 Procedure SetIBDataBaseErrorMessages( Value : TIBDataBaseErrorMessages)
 Function GetIBDataBaseErrorMessages : TIBDataBaseErrorMessages


//****************************unit uPSI_BoldUtils;******************************************
 Function CharCount( c : char; const s : string) : integer
 Function BoldNamesEqual( const name1, name2 : string) : Boolean
 Procedure BoldAppendToStrings(strings: TStrings; const aString: string; const ForceNewLine:Boolean)
 Function BoldSeparateStringList(strings:TStringList;const Separator,PreString,PostString:String):String
 Function BoldSeparatedAppend( const S1, S2 : string; const Separator : string) : string
 Function BoldTrim( const S : string) : string
 Function BoldIsPrefix( const S, Prefix : string) : Boolean
 Function BoldStrEqual( P1, P2 : PChar; Len : integer) : Boolean
 Function BoldStrAnsiEqual( P1, P2 : PChar; Len : integer) : Boolean
 Function BoldAnsiEqual( const S1, S2 : string) : Boolean
 Function BoldStrStringEqual( const S1 : string; P2 : PChar; Len : integer) : Boolean
 Function BoldCaseIndependentPos( const Substr, S : string) : Integer
 //Procedure EnumToStrings( aTypeInfo : pTypeInfo; Strings : TStrings)
 Function CapitalisedToSpaced( Capitalised : String) : String
 Function SpacedToCapitalised( Spaced : String) : String
 Function BooleanToString( BoolValue : Boolean) : String
 Function StringToBoolean( StrValue : String) : Boolean
 Function GetUpperLimitForMultiplicity( const Multiplicity : String) : Integer
 Function GetLowerLimitForMultiplicity( const Multiplicity : String) : Integer
 Function StringListToVarArray( List : TStringList) : variant
 Function IsLocalMachine( const Machinename : WideString) : Boolean
 Function GetComputerNameStr : string
 Function TimeStampComp( const Time1, Time2 : TTimeStamp) : Integer
 Function BoldStrToDateFmt(const S:string;const DateFormat:string;const DateSeparatorChar:char):TDateTime
 Function BoldDateToStrFmt(const aDate:TDateTime;DateFormat:string;const DateSeparatorChar:char):String;
 Function BoldParseFormattedDateList(const value:String;const formats:TStrings;var Date:TDateTime):Boolean;
 Function BoldParseFormattedDate(const value:String;const formats:array of string; var
Date:TDateTime):Boolean;
 Procedure EnsureTrailing( var Str : String; ch : char)
 Function BoldDirectoryExists( const Name : string) : Boolean
 Function BoldForceDirectories( Dir : string) : Boolean
 Function BoldRootRegistryKey : string
 Function GetModuleFileNameAsString( IncludePath : Boolean) : string
 Function BoldVariantToStrings( V : OleVariant; Strings : TStrings) : Integer
 Function LogicalAnd( A, B : Integer) : Boolean
 record TByHandleFileInformation dwFileAttributes : DWORD; '
   +'ftCreationTime : TFileTime; ftLastAccessTime : TFileTime; ftLastWriteTime '
   +': TFileTime; dwVolumeSerialNumber : DWORD; nFileSizeHigh : DWORD; nFileSiz'
   +'eLow : DWORD; nNumberOfLinks : DWORD; nFileIndexHigh : DWORD; nFileIndexLow : DWORD; end
 Function GetFileInformationByHandle(hFile:THandle;var lpFileInformation:TByHandleFileInformation):BOOL;
 Function IsFirstInstance : Boolean
 Procedure ActivateFirst( AString : PChar)
 Procedure ActivateFirstCommandLine
 function MakeAckPkt(const BlockNumber: Word): string;
 procedure SendError(UDPBase:TIdUDPBase;APeerIP:string;const APort:Int;const ErrNumber:Word;ErrStr:string;
 procedure SendError(UDPClient: TIdUDPClient; const ErrNumber: Word; ErrorString: string); overload;
 procedure SendError(UDPBase: TIdUDPBase; APeerIP: string; const APort: Integer;  E: Exception); overload;
 procedure SendError(UDPClient: TIdUDPClient; E: Exception); overload;
 function IdStrToWord(const Value: String): Word;
 function IdWordToStr(const Value: Word): WordStr;
 Function HasInstructionSet( const InstructionSet : TCPUInstructionSet) : Boolean
 Function CPUFeatures : TCPUFeatures

procedure SIRegister_xrtl_util_CPUUtils(CL: TPSPascalCompiler);
begin
  AddTypeS('TXRTLBitIndex', 'Integer
 Function XRTLSwapBits( Data : Cardinal; Bit1Index, Bit2Index : TXRTLBitIndex) : Cardinal
 Function XRTLBitTest( Data : Cardinal; BitIndex : TXRTLBitIndex) : Boolean
 Function XRTLBitSet( Data : Cardinal; BitIndex : TXRTLBitIndex) : Cardinal
 Function XRTLBitReset( Data : Cardinal; BitIndex : TXRTLBitIndex) : Cardinal
 Function XRTLBitComplement( Data : Cardinal; BitIndex : TXRTLBitIndex) : Cardinal
 Function XRTLSwapHiLo16( X : Word) : Word
 Function XRTLSwapHiLo32( X : Cardinal) : Cardinal
 Function XRTLSwapHiLo64( X : Int64) : Int64
 Function XRTLROL32( A, S : Cardinal) : Cardinal
 Function XRTLROR32( A, S : Cardinal) : Cardinal
 Function XRTLROL16( A : Word; S : Cardinal) : Word
 Function XRTLROR16( A : Word; S : Cardinal) : Word
 Function XRTLROL8( A : Byte; S : Cardinal) : Byte
 Function XRTLROR8( A : Byte; S : Cardinal) : Byte
 //Procedure XRTLXorBlock( I1, I2, O1 : PByteArray; Len : integer)
 //Procedure XRTLIncBlock( P : PByteArray; Len : integer)
 Procedure XRTLUMul64( const A, B : Integer; var MulL, MulH : Integer)
 Function XRTLPopulation( A : Cardinal) : Cardinal
end;

Function XRTLURLDecode( const ASrc : WideString) : WideString
Function XRTLURLEncode( const ASrc : WideString) : WideString
Function XRTLURINormalize( const AURI : WideString) : WideString
Procedure XRTLURIParse(const AURI:WideString;var VProtocol,VHost,VPath,VDocument,VPort,VBookmark,VUserName,
VPassword : WideString)
Function XRTLExtractLongPathName(APath: string): string;

procedure SIRegister_cFundamentUtils(CL: TPSPascalCompiler);
begin
  AddTypeS('Int8', 'ShortInt
  AddTypeS('Int16', 'SmallInt
  AddTypeS('Int32', 'LongInt
  AddTypeS('UInt8', 'Byte
  AddTypeS('UInt16', 'Word
  AddTypeS('UInt32', 'LongWord
  AddTypeS('UInt64', 'Int64
  AddTypeS('Word8', 'UInt8
  AddTypeS('Word16', 'UInt16
  AddTypeS('Word32', 'UInt32
  AddTypeS('Word64', 'UInt64
  AddTypeS('LargeInt', 'Int64
  AddTypeS('NativeInt', 'Integer
  AddTypeS('NativeUInt', 'Cardinal
  Const('BitsPerByte','LongInt'( 8);
  Const('BitsPerWord','LongInt'( 16);
  Const('BitsPerLongWord','LongInt'( 32);
 //Const('BitsPerCardinal','LongInt'( BytesPerCardinal * 8);
 //Const('BitsPerNativeWord','LongInt'( BytesPerNativeWord * 8);
 Function MinI( const A, B : Integer) : Integer
 Function MaxI( const A, B : Integer) : Integer
 Function MinC( const A, B : Cardinal) : Cardinal
 Function MaxC( const A, B : Cardinal) : Cardinal
 Function SumClipI( const A, I : Integer) : Integer
 Function SumClipC( const A : Cardinal; const I : Integer) : Cardinal
 Function InByteRange( const A : Int64) : Boolean
 Function InWordRange( const A : Int64) : Boolean
 Function InLongWordRange( const A : Int64) : Boolean
 Function InShortIntRange( const A : Int64) : Boolean
 Function InSmallIntRange( const A : Int64) : Boolean
 Function InLongIntRange( const A : Int64) : Boolean
  AddTypeS('Bool8', 'ByteBool
  AddTypeS('Bool16', 'WordBool
  AddTypeS('Bool32', 'LongBool
  AddTypeS('TCompareResult', '( crLess, crEqual, crGreater, crUndefined )
  AddTypeS('TCompareResultSet', 'set of TCompareResult
 Function ReverseCompareResult( const C : TCompareResult) : TCompareResult
 Const('MinSingle','Single').setExtended( 1.5E-45);
 Const('MaxSingle','Single').setExtended( 3.4E+38);
 Const('MinDouble','Double').setExtended( 5.0E-324);
 Const('MaxDouble','Double').setExtended( 1.7E+308);
 Const('MinExtended','Extended').setExtended(3.4E-4932);
 Const('MaxExtended','Extended').setExtended(1.1E+4932);
 Const('MinCurrency','Currency').SetExtended( - 922337203685477.5807);
 Const('MaxCurrency','Currency').SetExtended( 922337203685477.5807);
 Function MinF( const A, B : Float) : Float
 Function MaxF( const A, B : Float) : Float
 Function ClipF( const Value : Float; const Low, High : Float) : Float
 Function InSingleRange( const A : Float) : Boolean
 Function InDoubleRange( const A : Float) : Boolean
 Function InCurrencyRange( const A : Float) : Boolean;
 Function InCurrencyRange1( const A : Int64) : Boolean;
 Function FloatExponentBase2( const A : Extended; var Exponent : Integer) : Boolean
 Function FloatExponentBase10( const A : Extended; var Exponent : Integer) : Boolean
 Function FloatIsInfinity( const A : Extended) : Boolean
 Function FloatIsNaN( const A : Extended) : Boolean
 Const('SingleCompareDelta','Extended').setExtended( 1.0E-34);
 Const('DoubleCompareDelta','Extended').setExtended( 1.0E-280);
 Const('ExtendedCompareDelta','Extended').setExtended( 1.0E-4400);
 Const('DefaultCompareDelta','Extended').SetExtended( 1.0E-34);
 Function FloatZero( const A : Float; const CompareDelta : Float) : Boolean
 Function FloatOne( const A : Float; const CompareDelta : Float) : Boolean
 Function FloatsEqual( const A, B : Float; const CompareDelta : Float) : Boolean
 Function FloatsCompare( const A, B : Float; const CompareDelta : Float) : TCompareResult
 Const('SingleCompareEpsilon','Extended').setExtended( 1.0E-5);
 Const('DoubleCompareEpsilon','Extended').setExtended( 1.0E-13);
 Const('ExtendedCompareEpsilon','Extended').setExtended( 1.0E-17);
 Const('DefaultCompareEpsilon','Extended').setExtended( 1.0E-10);
 Function ApproxEqual( const A, B : Extended; const CompareEpsilon : Double) : Boolean
 Function ApproxCompare( const A, B : Extended; const CompareEpsilon : Double): TCompareResult
 Function cClearBit( const Value, BitIndex : LongWord) : LongWord
 Function cSetBit( const Value, BitIndex : LongWord) : LongWord
 Function cIsBitSet( const Value, BitIndex : LongWord) : Boolean
 Function cToggleBit( const Value, BitIndex : LongWord) : LongWord
 Function cIsHighBitSet( const Value : LongWord) : Boolean
 Function SetBitScanForward( const Value : LongWord) : Integer;
 Function SetBitScanForward1( const Value, BitIndex : LongWord) : Integer;
 Function SetBitScanReverse( const Value : LongWord) : Integer;
 Function SetBitScanReverse1( const Value, BitIndex : LongWord) : Integer;
 Function ClearBitScanForward( const Value : LongWord) : Integer;
 Function ClearBitScanForward1( const Value, BitIndex : LongWord) : Integer;
 Function ClearBitScanReverse( const Value : LongWord) : Integer;
 Function ClearBitScanReverse1( const Value, BitIndex : LongWord) : Integer;
 Function cReverseBits( const Value : LongWord) : LongWord;
 Function cReverseBits1( const Value : LongWord; const BitCount : Integer) : LongWord;
 Function cSwapEndian( const Value : LongWord) : LongWord
 Function cTwosComplement( const Value : LongWord) : LongWord
 Function RotateLeftBits16( const Value : Word; const Bits : Byte) : Word
 Function RotateLeftBits32( const Value : LongWord; const Bits : Byte) : LongWord
 Function RotateRightBits16( const Value : Word; const Bits : Byte) : Word
 Function RotateRightBits32( const Value : LongWord; const Bits : Byte) : LongWord
 Function cBitCount( const Value : LongWord) : LongWord
 Function cIsPowerOfTwo( const Value : LongWord) : Boolean
 Function LowBitMask( const HighBitIndex : LongWord) : LongWord
 Function HighBitMask( const LowBitIndex : LongWord) : LongWord
 Function RangeBitMask( const LowBitIndex, HighBitIndex : LongWord) : LongWord
 Function SetBitRange( const Value: LongWord; const LowBitIndex, HighBitIndex: LongWord) : LongWord
 Function ClearBitRange(const Value: LongWord; const LowBitIndex,HighBitIndex: LongWord) : LongWord
 Function ToggleBitRange(const Value:LongWord; const LowBitIndex,HighBitIndex: LongWord) : LongWord
 Function IsBitRangeSet(const Value: LongWord; const LowBitIndex,HighBitIndex : LongWord) : Boolean
 Function IsBitRangeClear(const Value: LongWord; const LowBitIndex,HighBitIndex: LongWord): Boolean
//  AddTypeS('CharSet', 'set of AnsiChar
  AddTypeS('CharSet', 'set of Char       //!!!
  AddTypeS('AnsiCharSet', 'TCharSet
  AddTypeS('ByteSet', 'set of Byte
  AddTypeS('AnsiChar', 'Char
   // Function AsCharSet( const C : array of AnsiChar) : CharSet
 Function AsByteSet( const C : array of Byte) : ByteSet
 Procedure ComplementChar( var C : CharSet; const Ch : Char)
 Procedure ClearCharSet( var C : CharSet)
 Procedure FillCharSet( var C : CharSet)
 Procedure ComplementCharSet( var C : CharSet)
 Procedure AssignCharSet( var DestSet : CharSet; const SourceSet : CharSet)
 Procedure Union( var DestSet : CharSet; const SourceSet : CharSet)
 Procedure Difference( var DestSet : CharSet; const SourceSet : CharSet)
 Procedure Intersection( var DestSet : CharSet; const SourceSet : CharSet)
 Procedure XORCharSet( var DestSet : CharSet; const SourceSet : CharSet)
 Function IsSubSet( const A, B : CharSet) : Boolean
 Function IsEqual( const A, B : CharSet) : Boolean
 Function IsEmpty( const C : CharSet) : Boolean
 Function IsComplete( const C : CharSet) : Boolean
 Function cCharCount( const C : CharSet) : Integer
 Procedure ConvertCaseInsensitive( var C : CharSet)
 Function CaseInsensitiveCharSet( const C : CharSet) : CharSet
 Function IntRangeLength( const Low, High : Integer) : Int64
 Function IntRangeAdjacent( const Low1, High1, Low2, High2 : Integer) : Boolean
 Function IntRangeOverlap( const Low1, High1, Low2, High2 : Integer) : Boolean
 Function IntRangeHasElement( const Low, High, Element : Integer) : Boolean
 Function IntRangeIncludeElement( var Low, High : Integer; const Element : Integer) : Boolean
 Function IntRangeIncludeElementRange(var Low,High:Integer;const LowElement,HighElement:Integer):Boolean
 Function CardRangeLength( const Low, High : Cardinal) : Int64
 Function CardRangeAdjacent( const Low1, High1, Low2, High2 : Cardinal) : Boolean
 Function CardRangeOverlap( const Low1, High1, Low2, High2 : Cardinal) : Boolean
 Function CardRangeHasElement( const Low, High, Element : Cardinal) : Boolean
 Function CardRangeIncludeElement( var Low, High : Cardinal; const Element : Cardinal) : Boolean
 Function CardRangeIncludeElementRange(var Low,High:Card;const LowElement,HighElement:Card):Boolean;
  AddTypeS('UnicodeChar', 'WideChar
 Function Compare( const I1, I2 : Boolean) : TCompareResult;
 Function Compare1( const I1, I2 : Integer) : TCompareResult;
 Function Compare2( const I1, I2 : Int64) : TCompareResult;
 Function Compare3( const I1, I2 : Extended) : TCompareResult;
 Function CompareA( const I1, I2 : AnsiString) : TCompareResult
 Function CompareW( const I1, I2 : WideString) : TCompareResult
 Function cSgn( const A : LongInt) : Integer;
 Function cSgn1( const A : Int64) : Integer;
 Function cSgn2( const A : Extended) : Integer;
 AddTypeS('TConvertResult', '( convertOK, convertFormatError, convertOverflow )
 Function AnsiCharToInt( const A : AnsiChar) : Integer
 Function WideCharToInt( const A : WideChar) : Integer
 Function CharToInt( const A : Char) : Integer
 Function IntToAnsiChar( const A : Integer) : AnsiChar
 Function IntToWideChar( const A : Integer) : WideChar
 Function IntToChar( const A : Integer) : Char
 Function IsHexAnsiChar( const Ch : AnsiChar) : Boolean
 Function IsHexWideChar( const Ch : WideChar) : Boolean
 Function IsHexChar( const Ch : Char) : Boolean
 Function HexAnsiCharToInt( const A : AnsiChar) : Integer
 Function HexWideCharToInt( const A : WideChar) : Integer
 Function HexCharToInt( const A : Char) : Integer
 Function IntToUpperHexAnsiChar( const A : Integer) : AnsiChar
 Function IntToUpperHexWideChar( const A : Integer) : WideChar
 Function IntToUpperHexChar( const A : Integer) : Char
 Function IntToLowerHexAnsiChar( const A : Integer) : AnsiChar
 Function IntToLowerHexWideChar( const A : Integer) : WideChar
 Function IntToLowerHexChar( const A : Integer) : Char
 Function IntToStringA( const A : Int64) : AnsiString
 Function IntToStringW( const A : Int64) : WideString
 Function IntToString( const A : Int64) : String
 Function UIntToStringA( const A : NativeUInt) : AnsiString
 Function UIntToStringW( const A : NativeUInt) : WideString
 Function UIntToString( const A : NativeUInt) : String
 Function LongWordToStrA( const A : LongWord; const Digits : Integer) : AnsiString
 Function LongWordToStrW( const A : LongWord; const Digits : Integer) : WideString
 Function LongWordToStrU( const A : LongWord; const Digits : Integer) : UnicodeString
 Function LongWordToStr( const A : LongWord; const Digits : Integer) : String
 Function LongWordToHexA(const A:LongWord;const Digits:Integer;const UpperCase:Boolean):AnsiString;
 Function LongWordToHexW(const A:LongWord;const Digits:Integer;const UpperCase:Boolean):WideString;
 Function LongWordToHex( const A : LongWord; const Digits : Integer;const UpperCase:Boolean):String
 Function LongWordToOctA( const A : LongWord; const Digits : Integer) : AnsiString
 Function LongWordToOctW( const A : LongWord; const Digits : Integer) : WideString
 Function LongWordToOct( const A : LongWord; const Digits : Integer) : String
 Function LongWordToBinA( const A : LongWord; const Digits : Integer) : AnsiString
 Function LongWordToBinW( const A : LongWord; const Digits : Integer) : WideString
 Function LongWordToBin( const A : LongWord; const Digits : Integer) : String
 Function TryStringToInt64A( const S : AnsiString; out A : Int64) : Boolean
 Function TryStringToInt64W( const S : WideString; out A : Int64) : Boolean
 Function TryStringToInt64( const S : String; out A : Int64) : Boolean
 Function StringToInt64DefA( const S : AnsiString; const Default : Int64) : Int64
 Function StringToInt64DefW( const S : WideString; const Default : Int64) : Int64
 Function StringToInt64Def( const S : String; const Default : Int64) : Int64
 Function StringToInt64A( const S : AnsiString) : Int64
 Function StringToInt64W( const S : WideString) : Int64
 Function StringToInt64( const S : String) : Int64
 Function TryStringToIntA( const S : AnsiString; out A : Integer) : Boolean
 Function TryStringToIntW( const S : WideString; out A : Integer) : Boolean
 Function TryStringToInt( const S : String; out A : Integer) : Boolean
 Function StringToIntDefA( const S : AnsiString; const Default : Integer) : Integer
 Function StringToIntDefW( const S : WideString; const Default : Integer) : Integer
 Function StringToIntDef( const S : String; const Default : Integer) : Integer
 Function StringToIntA( const S : AnsiString) : Integer
 Function StringToIntW( const S : WideString) : Integer
 Function StringToInt( const S : String) : Integer
 Function TryStringToLongWordA( const S : AnsiString; out A : LongWord) : Boolean
 Function TryStringToLongWordW( const S : WideString; out A : LongWord) : Boolean
 Function TryStringToLongWord( const S : String; out A : LongWord) : Boolean
 Function StringToLongWordA( const S : AnsiString) : LongWord
 Function StringToLongWordW( const S : WideString) : LongWord
 Function StringToLongWord( const S : String) : LongWord
 Function HexToUIntA( const S : AnsiString) : NativeUInt
 Function HexToUIntW( const S : WideString) : NativeUInt
 Function HexToUInt( const S : String) : NativeUInt
 Function TryHexToLongWordA( const S : AnsiString; out A : LongWord) : Boolean
 Function TryHexToLongWordW( const S : WideString; out A : LongWord) : Boolean
 Function TryHexToLongWord( const S : String; out A : LongWord) : Boolean
 Function HexToLongWordA( const S : AnsiString) : LongWord
 Function HexToLongWordW( const S : WideString) : LongWord
 Function HexToLongWord( const S : String) : LongWord
 Function TryOctToLongWordA( const S : AnsiString; out A : LongWord) : Boolean
 Function TryOctToLongWordW( const S : WideString; out A : LongWord) : Boolean
 Function TryOctToLongWord( const S : String; out A : LongWord) : Boolean
 Function OctToLongWordA( const S : AnsiString) : LongWord
 Function OctToLongWordW( const S : WideString) : LongWord
 Function OctToLongWord( const S : String) : LongWord
 Function TryBinToLongWordA( const S : AnsiString; out A : LongWord) : Boolean
 Function TryBinToLongWordW( const S : WideString; out A : LongWord) : Boolean
 Function TryBinToLongWord( const S : String; out A : LongWord) : Boolean
 Function BinToLongWordA( const S : AnsiString) : LongWord
 Function BinToLongWordW( const S : WideString) : LongWord
 Function BinToLongWord( const S : String) : LongWord
 Function FloatToStringA( const A : Extended) : AnsiString
 Function FloatToStringW( const A : Extended) : WideString
 Function FloatToString( const A : Extended) : String
 Function TryStringToFloatA( const A : AnsiString; out B : Extended) : Boolean
 Function TryStringToFloatW( const A : WideString; out B : Extended) : Boolean
 Function TryStringToFloat( const A : String; out B : Extended) : Boolean
 Function StringToFloatA( const A : AnsiString) : Extended
 Function StringToFloatW( const A : WideString) : Extended
 Function StringToFloat( const A : String) : Extended
 Function StringToFloatDefA( const A : AnsiString; const Default : Extended) : Extended
 Function StringToFloatDefW( const A : WideString; const Default : Extended) : Extended
 Function StringToFloatDef( const A : String; const Default : Extended) : Extended
 Function EncodeBase64(const S,Alphabet:AnsiString; const Pad:Boolean; const PadMultiple:Integer; const
PadChar: AnsiChar) : AnsiString
 Function DecodeBase64( const S, Alphabet : AnsiString; const PadSet : CharSet) : AnsiString
```

```
 unit uPSI_cFundamentUtils;
 Const('b64_MIMEBase64','Str').String('ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/
 Const('b64_UUEncode','String').String('!"#$%&''()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_';
 Const('b64_XXEncode','String').String('+-0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz';
 Const('CCHARSET','Stringb64_XXEncode);
 Const('CHEXSET','String'0123456789ABCDEF
 Const('HEXDIGITS','String'0123456789ABCDEF
 StHexDigits  : array[0..$F] of Char = '0123456789ABCDEF';
 Const('DIGISET','String'0123456789
 Const('LETTERSET','String'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
 Const('DIGISET2','TCharset').SetSet('0123456789'
 Const('LETTERSET2','TCharset').SetSet('ABCDEFGHIJKLMNOPQRSTUVWXYZ'
 Const('HEXSET2','TCharset').SetSET('0123456789ABCDEF');
 Const('NUMBERSET','TCharset').SetSet('0123456789');
 Const('NUMBERS','String'0123456789');
 Const('LETTERS','String'ABCDEFGHIJKLMNOPQRSTUVWXYZ');
 Function CharSetToStr( const C : CharSet) : AnsiString
 Function StrToCharSet( const S : AnsiString) : CharSet
 Function MIMEBase64Decode( const S : AnsiString) : AnsiString
 Function MIMEBase64Encode( const S : AnsiString) : AnsiString
 Function UUDecode( const S : AnsiString) : AnsiString
 Function XXDecode( const S : AnsiString) : AnsiString
 Function BytesToHex( const P : array of Byte; const UpperCase : Boolean) : AnsiString
 Function InterfaceToStrA( const I : IInterface) : AnsiString
 Function InterfaceToStrW( const I : IInterface) : WideString
 Function InterfaceToStr( const I : IInterface) : String
 Function ObjectClassName( const O : TObject) : String
 Function ClassClassName( const C : TClass) : String
 Function ObjectToStr( const O : TObject) : String
 Function ObjectToString( const O : TObject) : String
 Function CharSetToStr( const C : CharSet) : AnsiString
 Function StrToCharSet( const S : AnsiString) : CharSet
 Function HashStrA(const S : AnsiString; const Index : Integer; const Count: Integer;const
AsciiCaseSensitive : Boolean; const Slots : LongWord) : LongWord
 Function HashStrW(const S:WideString;const Index:Integer;const Count:Integer;const
AsciiCaseSensitive:Boolean; const Slots:LongWord) : LongWord
 Function HashStr(const S : String; const Index : Integer; const Count : Integer; const AsciiCaseSensitive
: Boolean; const Slots : LongWord) : LongWord
 Function HashInteger( const I : Integer; const Slots : LongWord) : LongWord
 Function HashLongWord( const I : LongWord; const Slots : LongWord) : LongWord
 Const('Bytes1KB','LongInt'( 1024);
  SIRegister_IInterface(CL);
 Procedure SelfTestCFundamentUtils

Function CreateSchedule : IJclSchedule
Function NullStamp : TTimeStamp
Function CompareTimeStamps( const Stamp1, Stamp2 : TTimeStamp) : Int64
Function EqualTimeStamps( const Stamp1, Stamp2 : TTimeStamp) : Boolean
Function IsNullTimeStamp( const Stamp : TTimeStamp) : Boolean

procedure SIRegister_uwinplot(CL: TPSPascalCompiler);
begin
 AddTypeS('TFunc', 'function(X : Float) : Float;
Function InitGraphics( Width, Height : Integer) : Boolean
Procedure SetWindow( Canvas : TCanvas; X1, X2, Y1, Y2 : Integer; GraphBorder : Boolean)
Procedure SetOxScale( Scale : TScale; OxMin, OxMax, OxStep : Float)
Procedure SetOyScale( Scale : TScale; OyMin, OyMax, OyStep : Float)
Procedure GetOxScale( var Scale : TScale; var OxMin, OxMax, OxStep : Float)
Procedure GetOyScale( var Scale : TScale; var OyMin, OyMax, OyStep : Float)
Procedure SetGraphTitle( Title : String)
Procedure SetOxTitle( Title : String)
Procedure SetOyTitle( Title : String)
Function GetGraphTitle : String
Function GetOxTitle : String
Function GetOyTitle : String
Procedure PlotOxAxis( Canvas : TCanvas)
Procedure PlotOyAxis( Canvas : TCanvas)
Procedure PlotGrid( Canvas : TCanvas; Grid : TGrid)
Procedure WriteGraphTitle( Canvas : TCanvas)
Function SetMaxCurv( NCurv : Byte) : Boolean
Procedure SetPointParam( CurvIndex, Symbol, Size : Integer; Color : TColor)
Procedure SetLineParam( CurvIndex : Integer; Style : TPenStyle; Width : Integer; Color : TColor)
Procedure SetCurvLegend( CurvIndex : Integer; Legend : String)
Procedure SetCurvStep( CurvIndex, Step : Integer)
Function GetMaxCurv : Byte
Procedure GetPointParam( CurvIndex : Integer; var Symbol, Size : Integer; var Color : TColor)
Procedure GetLineParam(CurvIndex:Integer;var Style:TPenStyle;var Width:Integer; var Color:TColor);
Function GetCurvLegend( CurvIndex : Integer) : String
Function GetCurvStep( CurvIndex : Integer) : Integer
Procedure PlotPoint( Canvas : TCanvas; X, Y : Float; CurvIndex : Integer)
Procedure PlotCurve( Canvas : TCanvas; X, Y : TVector; Lb, Ub, CurvIndex : Integer)
Procedure PlotCurveWithErrorBars(Canvas : TCanvas; X,Y,S: TVector; Ns,Lb,Ub,CurvIndex:Integer)
Procedure PlotFunc(Canvas: TCanvas; Func: TFunc; Xmin,Xmax: Float; Npt,CurvIndex: Integer)
Procedure WriteLegend( Canvas : TCanvas; NCurv : Integer; ShowPoints, ShowLines : Boolean)
Procedure ConRec( Canvas : TCanvas; Nx, Ny, Nc : Integer; X, Y, Z : TVector; F : TMatrix)
Function Xpixel( X : Float) : Integer
Function Ypixel( Y : Float) : Integer
Function Xuser( X : Integer) : Float
Function Yuser( Y : Integer) : Float
```

```
end;

Procedure FFT( NumSamples : Integer; InArray, OutArray : TCompVector)
Procedure IFFT( NumSamples : Integer; InArray, OutArray : TCompVector)
Procedure FFT_Integer( NumSamples : Integer; RealIn, ImagIn : TIntVector; OutArray : TCompVector)
Procedure FFT_Integer_Cleanup
Procedure CalcFrequency(NumSamples,FrequencyIndex: Integer;InArray: TCompVector;var FT : Complex)
//unit uPSI_JclStreams;
Function StreamSeek( Stream : TStream; const Offset : Int64; const Origin : TSeekOrigin) : Int64
Function StreamCopy( Source : TStream; Dest : TStream; BufferSize : Integer) : Int64
Function CompareStreams( A, B : TStream; BufferSize : Integer) : Boolean
Function JCompareFiles( const FileA, FileB : TFileName; BufferSize : Integer) : Boolean

procedure SIRegister_FmxUtils(CL: TPSPascalCompiler);
begin
 FindClass('TOBJECT'),'EInvalidDest
 FindClass('TOBJECT'),'EFCantMove
 Procedure fmxCopyFile( const FileName, DestName : string)
 Procedure fmxMoveFile( const FileName, DestName : string)
 Function fmxGetFileSize( const FileName : string) : LongInt
 Function fmxFileDateTime( const FileName : string) : TDateTime
 Function fmxHasAttr( const FileName : string; Attr : Word) : Boolean
 Function fmxExecuteFile( const FileName, Params, DefaultDir : string; ShowCmd : Integer):THandle;
end;

procedure SIRegister_FindFileIter(CL: TPSPascalCompiler);
begin
 SIRegister_IFindFileIterator(CL);
 Function CreateFindFile(const Path:string; IncludeAttr:Integer;out iffi:IFindFileIterator):Bool;
end;

procedure SIRegister_PCharUtils(CL: TPSPascalCompiler);
begin
 Function SkipWhite( cp : PChar) : PChar
 Function ReadStringDoubleQuotedMaybe( cp : PChar; var AStr : string) : PChar
 Function ReadStringSingleQuotedMaybe( cp : PChar; var AStr : string) : PChar
 Function ReadIdent( cp : PChar; var ident : string) : PChar
end;

procedure SIRegister_JclStrHashMap(CL: TPSPascalCompiler);
begin
  SIRegister_TStringHashMapTraits(CL);
 Function CaseSensitiveTraits : TStringHashMapTraits
 Function CaseInsensitiveTraits : TStringHashMapTraits
  THashNode', 'record Str : string; Ptr : Pointer; Left : PHashNod'
   +'e; Right : PHashNode; end
  //PHashArray', '^THashArray // will not work
  SIRegister_TStringHashMap(CL);
 THashValue', 'Cardinal
 Function StrHash( const s : string) : THashValue
 Function TextHash( const s : string) : THashValue
 Function DataHash( var AValue, ASize : Cardinal) : THashValue
 Function Iterate_FreeObjects( AUserData : Pointer; const AStr : string; var AData : Pointer) : Boolean
 Function Iterate_Dispose( AUserData : Pointer; const AStr : string; var AData : Pointer) : Boolean
 Function Iterate_FreeMem( AUserData : Pointer; const AStr : string; var AData : Pointer) : Boolean
  SIRegister_TCaseSensitiveTraits(CL);
  SIRegister_TCaseInsensitiveTraits(CL);


//**********************************************************************unit uPSI_umath;
 Function uExpo( X : Float) : Float
 Function uExp2( X : Float) : Float
 Function uExp10( X : Float) : Float
 Function uLog( X : Float) : Float
 Function uLog2( X : Float) : Float
 Function uLog10( X : Float) : Float
 Function uLogA( X, A : Float) : Float
 Function uIntPower( X : Float; N : Integer): Float
 Function uPower( X, Y : Float) : Float
 Function SgnGamma( X : Float) : Integer
 Function Stirling( X : Float) : Float
 Function StirLog( X : Float) : Float
 Function Gamma( X : Float) : Float
 Function LnGamma( X : Float) : Float
 Function DiGamma( X : Float) : Float
 Function TriGamma( X : Float) : Float
 Function IGamma( X : Float) : Float
 Function JGamma( X : Float) : Float
 Function InvGamma( X : Float) : Float
 Function Erf( X : Float) : Float
 Function Erfc( X : Float) : Float
Function Correl(X, Y : TVector; Lb, Ub : Integer) : Float;
{ Correlation coefficient between samples X and Y }
function DBeta(A, B, X : Float) : Float;
{ Density of Beta distribution with parameters A and B }
Function LambertW( X : Float; UBranch, Offset : Boolean) : Float
Function Beta(X, Y : Float) : Float
Function Binomial( N, K : Integer) : Float
Function PBinom( N : Integer; P : Float; K : Integer) : Float
```

```
Procedure Cholesky( A, L : TMatrix; Lb, Ub : Integer)
Procedure LU_Decomp( A : TMatrix; Lb, Ub : Integer)
Procedure LU_Solve( A : TMatrix; B : TVector; Lb, Ub : Integer; X : TVector)
Function DNorm( X : Float) : Float

function DGamma(A, B, X : Float) : Float;
{ Density of Gamma distribution with parameters A and B }
function DKhi2(Nu : Integer; X : Float) : Float;
{ Density of Khi-2 distribution with Nu d.o.f. }
function DStudent(Nu : Integer; X : Float) : Float;
{ Density of Student distribution with Nu d.o.f. }
function DSnedecor(Nu1, Nu2 : Integer; X : Float) : Float;
{ Density of Fisher-Snedecor distribution with Nu1 and Nu2 d.o.f. }
function IBeta(A, B, X : Float) : Float;
{ Incomplete Beta function}
function Correl(X, Y : TVector; Lb, Ub : Integer) : Float;

procedure SIRegister_unlfit(CL: TPSPascalCompiler);
begin
 Procedure SetOptAlgo( Algo : TOptAlgo)
procedure SetOptAlgo(Algo : TOptAlgo);
{ ------------------------------------------------------------------
  Sets the optimization algorithm according to Algo, which must be
  NL_MARQ, NL_SIMP, NL_BFGS, NL_SA, NL_GA. Default is NL_MARQ   }

 Function GetOptAlgo : TOptAlgo
 Procedure SetMaxParam( N : Byte)
 Function GetMaxParam : Byte
 Procedure SetParamBounds( I : Byte; ParamMin, ParamMax : Float)
 Procedure GetParamBounds( I : Byte; var ParamMin, ParamMax : Float)
 Function NullParam( B : TVector; Lb, Ub : Integer) : Boolean
 Procedure NLFit( RegFunc : TRegFunc; DerivProc : TDerivProc; X, Y : TVector; Lb, Ub : Integer; MaxIter :
Integer; Tol : Float; B : TVector; FirstPar, LastPar : Integer; V : TMatrix)
 Procedure WNLFit( RegFunc : TRegFunc; DerivProc : TDerivProc; X, Y, S : TVector; Lb, Ub:Integer;
MaxIter:Integer;Tol : Float; B : TVector; FirstPar, LastPar : Integer; V : TMatrix)
 Procedure SetMCFile( FileName : String)
 Procedure SimFit(RegFunc:TRegFunc;X,Y:TVector;Lb,Ub:Integer;B:TVector;FirstPar,LastPar:Integer;V:TMatrix;
 Procedure WSimFit(RegFunc:TRegFunc; X,Y,S:TVector;Lb,Ub:Integer;B:TVector;FirstPar,
LastPar:Integer;V:TMatrix);
end;

(*----------------------------------------------------------------------------*)
procedure SIRegister_usimplex(CL: TPSPascalCompiler);
begin
 Procedure SaveSimplex( FileName : string)
 Procedure Simplex(Func:TFuncNVar; X:TVector;Lb,Ub:Integer; MaxIter:Integer;Tol:Float; var F_min:Float);
end;
(*----------------------------------------------------------------------------*)
procedure SIRegister_uregtest(CL: TPSPascalCompiler);
begin
 Procedure RegTest(Y,Ycalc: TVector;LbY,UbY:Integer;V:TMatrix;LbV,UbV:Integer;var Test:TRegTest)
 Procedure WRegTest(Y,Ycalc,S:TVector;LbY,UbY:Integer;V:TMatrix;LbV,UbV:Integer;var Test:TRegTest);
end;

procedure SIRegister_ustrings(CL: TPSPascalCompiler);
begin
 Function LTrim( S : String) : String
 Function RTrim( S : String) : String
 Function uTrim( S : String) : String
 Function StrChar( N : Byte; C : Char) : String
 Function RFill( S : String; L : Byte) : String
 Function LFill( S : String; L : Byte) : String
 Function CFill( S : String; L : Byte) : String
 Function Replace( S : String; C1, C2 : Char) : String
 Function Extract( S : String; var Index : Byte; Delim : Char) : String
 Procedure Parse( S : String; Delim : Char; Field : TStrVector; var N : Byte)
 Procedure SetFormat( NumLength, MaxDec : Integer; FloatPoint, NSZero : Boolean)
 Function FloatStr( X : Float) : String
 Function IntStr( N : LongInt) : String
 Function uCompStr( Z : Complex) : String
end;

procedure SIRegister_uhyper(CL: TPSPascalCompiler);
begin
 Function uSinh( X : Float) : Float
 Function uCosh( X : Float) : Float
 Function uTanh( X : Float) : Float
 Function uArcSinh( X : Float) : Float
 Function uArcCosh( X : Float) : Float
 Function ArcTanh( X : Float) : Float
 Procedure SinhCosh( X : Float; var SinhX, CoshX : Float)
end;

procedure SIRegister_urandom(CL: TPSPascalCompiler);
begin
type RNG_Type =
  (RNG_MWC,      { Multiply-With-Carry }
   RNG_MT,       { Mersenne Twister }
   RNG_UVAG);    { Universal Virtual Array Generator }
 Procedure SetRNG( RNG : RNG_Type)
 Procedure InitGen( Seed : RNG_IntType)
 Procedure SRand( Seed : RNG_IntType)
 Function IRanGen : RNG_IntType
 Function IRanGen31 : RNG_IntType
 Function RanGen1 : Float
 Function RanGen2 : Float
 Function RanGen3 : Float
 Function RanGen53 : Float
end;

//  Optimization by Simulated Annealing
procedure SIRegister_usimann(CL: TPSPascalCompiler);
begin
 Procedure InitSAParams( NT, NS, NCycles : Integer; RT : Float)
 Procedure SA_CreateLogFile( FileName : String)
 Procedure SimAnn(Func: TFuncNVar; X, Xmin, Xmax : TVector; Lb, Ub : Integer; var F_min : Float);
end;

procedure SIRegister_uranuvag(CL: TPSPascalCompiler);
begin
 Procedure InitUVAGbyString( KeyPhrase : string)
 Procedure InitUVAG( Seed : RNG_IntType)
 Function IRanUVAG : RNG_IntType
end;

procedure SIRegister_ugenalg(CL: TPSPascalCompiler);
begin
 Procedure InitGAParams( NP, NG : Integer; SR, MR, HR : Float)
 Procedure GA_CreateLogFile( LogFileName : String)
 Procedure GenAlg(Func: TFuncNVar; X, Xmin, Xmax : TVector; Lb, Ub : Integer; var F_min : Float);
end;

 TVector', 'array of Float
procedure SIRegister_uqsort(CL: TPSPascalCompiler);
begin
 Procedure QSort( X : TVector; Lb, Ub : Integer)
 Procedure DQSort( X : TVector; Lb, Ub : Integer)
end;

procedure SIRegister_uinterv(CL: TPSPascalCompiler);
begin
 Procedure Interval( X1, X2 : Float; MinDiv, MaxDiv : Integer; var Min, Max, Step : Float)
 Procedure AutoScale(X: TVector; Lb, Ub : Integer; Scale : TScale; var XMin, XMax, XStep:Float)
end;

procedure SIRegister_D2XXUnit(CL: TPSPascalCompiler);
begin
  FT_Result', 'Integer
  //TDWordptr', '^DWord // will not work
  TFT_Program_Data', 'record Signature1 : DWord; Signature2 : DWor'
   d; Version : DWord; VendorID : Word; ProductID : Word; Manufacturer : PCha'
   r; ManufacturerID : PChar; Description : PChar; SerialNumber : PChar; MaxP'
   ower : Word; PnP : Word; SelfPowered : Word; RemoteWakeup : Word; Rev4 : B'
   yte; IsoIn : Byte; IsoOut : Byte; PullDownEnable : Byte; SerNumEnable : By'
   te; USBVersionEnable : Byte; USBVersion : Word; Rev5 : Byte; IsoInA : Byte'
   ; IsoInB : Byte; IsoOutA : Byte; IsoOutB : Byte; PullDownEnable5 : Byte; S'
   erNumEnable5 : Byte; USBVersionEnable5 : Byte; USBVersion5 : Word; AIsHigh'
   Current : Byte; BIsHighCurrent : Byte; IFAIsFifo : Byte; IFAIsFifoTar : By'
   te; IFAIsFastSer : Byte; AIsVCP : Byte; IFBIsFifo : Byte; IFBIsFifoTar : B'
   yte; IFBIsFastSer : Byte; BIsVCP : Byte; UseExtOsc : Byte; HighDriveIOs : '
   Byte; EndpointSize : Byte; PullDownEnableR : Byte; SerNumEnableR : Byte; I'
   nvertTXD : Byte; InvertRXD : Byte; InvertRTS : Byte; InvertCTS : Byte; Inv'
   ertDTR : Byte; InvertDSR : Byte; InvertDCD : Byte; InvertRI : Byte; Cbus0 '
   : Byte; Cbus1 : Byte; Cbus2 : Byte; Cbus3 : Byte; Cbus4 : Byte; RIsVCP : B'
   yte; end
end;


//**************************************** PaintFX****************************
procedure SIRegister_TJvPaintFX(CL: TPSPascalCompiler);
begin
  //with RegClassS(CL,'TComponent', 'TJvPaintFX') do
  with AddClassN(FindClass('TComponent'),'TJvPaintFX') do begin
    Procedure Solarize( const Src : TBitmap; var Dst : TBitmap; Amount : Integer)
    Procedure Posterize( const Src : TBitmap; var Dst : TBitmap; Amount : Integer)
    Procedure Blend( const Src1, Src2 : TBitmap; var Dst : TBitmap; Amount : Single)
    Procedure Blend2( const Src1, Src2 : TBitmap; var Dst : TBitmap; Amount : Single)
    Procedure ExtractColor( const Dst : TBitmap; AColor : TColor)
    Procedure ExcludeColor( const Dst : TBitmap; AColor : TColor)
    Procedure Turn( Src, Dst : TBitmap)
    Procedure TurnRight( Src, Dst : TBitmap)
    Procedure HeightMap( const Dst : TBitmap; Amount : Integer)
    Procedure TexturizeTile( const Dst : TBitmap; Amount : Integer)
    Procedure TexturizeOverlap( const Dst : TBitmap; Amount : Integer)
    Procedure RippleRandom( const Dst : TBitmap; Amount : Integer)
    Procedure RippleTooth( const Dst : TBitmap; Amount : Integer)
    Procedure RippleTriangle( const Dst : TBitmap; Amount : Integer)
    Procedure Triangles( const Dst : TBitmap; Amount : Integer)
    Procedure DrawMandelJulia(const Dst: TBitmap; x0,y0,x1,y1:Single; Niter:Integer; Mandel:Boolean)
    Procedure FilterXBlue( const Dst : TBitmap; Min, Max : Integer)
    Procedure FilterXGreen( const Dst : TBitmap; Min, Max : Integer)
    Procedure FilterXRed( const Dst : TBitmap; Min, Max : Integer)
    Procedure FilterBlue( const Dst : TBitmap; Min, Max : Integer)
    Procedure FilterGreen( const Dst : TBitmap; Min, Max : Integer)
    Procedure FilterRed( const Dst : TBitmap; Min, Max : Integer)
    Procedure Emboss( var Bmp : TBitmap)
    Procedure Plasma( Src1, Src2, Dst : TBitmap; Scale, Turbulence : Single)
    Procedure Shake( Src, Dst : TBitmap; Factor : Single)
    Procedure ShakeDown( Src, Dst : TBitmap; Factor : Single)
    Procedure KeepBlue( const Dst : TBitmap; Factor : Single)
    Procedure KeepGreen( const Dst : TBitmap; Factor : Single)
    Procedure KeepRed( const Dst : TBitmap; Factor : Single)
    Procedure Mandelbrot( const Dst : TBitmap; Factor : Integer)
    Procedure MaskMandelbrot( const Dst : TBitmap; Factor : Integer)
    Procedure FoldRight( Src1, Src2, Dst : TBitmap; Amount : Single)
    Procedure QuartoOpaque( Src, Dst : TBitmap)
    Procedure SemiOpaque( Src, Dst : TBitmap)
    Procedure ShadowDownLeft( const Dst : TBitmap)
    Procedure ShadowDownRight( const Dst : TBitmap)
    Procedure ShadowUpLeft( const Dst : TBitmap)
    Procedure ShadowUpRight( const Dst : TBitmap)
    Procedure Darkness( const Dst : TBitmap; Amount : Integer)
    Procedure Trace( const Dst : TBitmap; Intensity : Integer)
    Procedure FlipRight( const Dst : TBitmap)
    Procedure FlipDown( const Dst : TBitmap)
    Procedure SpotLight( const Dst : TBitmap; Amount : Integer; Spot : TRect)
    Procedure SplitLight( const Dst : TBitmap; Amount : Integer)
    Procedure MakeSeamlessClip( var Dst : TBitmap; Seam : Integer)
    Procedure Wave( const Dst : TBitmap; Amount, Inference, Style : Integer)
    Procedure Mosaic( const Bm : TBitmap; Size : Integer)
    Procedure SmoothRotate( var Src, Dst : TBitmap; CX, CY : Integer; Angle : Single)
    Procedure SmoothResize( var Src, Dst : TBitmap)
    Procedure Twist( var Bmp, Dst : TBitmap; Amount : Integer)
    Procedure SplitBlur( const Dst : TBitmap; Amount : Integer)
    Procedure GaussianBlur( const Dst : TBitmap; Amount : Integer)
    Procedure Smooth( const Dst : TBitmap; Weight : Integer)
    Procedure GrayScale( const Dst : TBitmap)
    Procedure AddColorNoise( const Dst : TBitmap; Amount : Integer)
    Procedure AddMonoNoise( const Dst : TBitmap; Amount : Integer)
    Procedure Contrast( const Dst : TBitmap; Amount : Integer)
    Procedure Lightness( const Dst : TBitmap; Amount : Integer)
    Procedure Saturation( const Dst : TBitmap; Amount : Integer)
    Procedure Spray( const Dst : TBitmap; Amount : Integer)
    Procedure AntiAlias( const Dst : TBitmap)
    Procedure AntiAliasRect( const Dst : TBitmap; XOrigin, YOrigin, XFinal, YFinal : Integer)
    Procedure SmoothPoint( const Dst : TBitmap; XK, YK : Integer)
    Procedure FishEye( var Bmp, Dst : TBitmap; Amount : Single)
    Procedure Marble( const Src : TBitmap; var Dst : TBitmap; Scale : Single; Turbulence : Integer)
    Procedure Marble2( const Src : TBitmap; var Dst : TBitmap; Scale : Single; Turbulence: Integer)
    Procedure Marble3( const Src : TBitmap; var Dst : TBitmap; Scale : Single; Turbulence: Integer)
    Procedure Marble4( const Src : TBitmap; var Dst : TBitmap; Scale : Single; Turbulence: Integer)
    Procedure Marble5( const Src : TBitmap; var Dst : TBitmap; Scale : Single; Turbulence: Integer)
    Procedure Marble6( const Src : TBitmap; var Dst : TBitmap; Scale : Single; Turbulence: Integer)
    Procedure Marble7( const Src : TBitmap; var Dst : TBitmap; Scale : Single; Turbulence: Integer)
    Procedure Marble8( const Src : TBitmap; var Dst : TBitmap; Scale : Single; Turbulence: Integer)
    Procedure SqueezeHor( Src, Dst : TBitmap; Amount : Integer; Style : TLightBrush)
    Procedure SplitRound( Src, Dst : TBitmap; Amount : Integer; Style : TLightBrush)
    Procedure Tile( Src, Dst : TBitmap; Amount : Integer)
    Procedure Stretch( Src, Dst : TBitmap; Filter : TFilterProc; AWidth : Single)
    Procedure Grow( Src1, Src2, Dst : TBitmap; Amount : Single; X, Y : Integer)
    Procedure Invert( Src : TBitmap)
    Procedure MirrorRight( Src : TBitmap)
    Procedure MirrorDown( Src : TBitmap)
  end;
end;

(*----------------------------------------------------------------------------*)
procedure SIRegister_JvPaintFX(CL: TPSPascalCompiler);
begin
  AddTypeS('TLightBrush', '( lbBrightness, lbContrast, lbSaturation, lbFishe'
   +'ye, lbrotate, lbtwist, lbrimple, mbHor, mbTop, mbBottom, mbDiamond, mbWast'
   +'e, mbRound, mbRound2, mbSplitRound, mbSplitWaste )
  SIRegister_TJvPaintFX(CL);
 Function SplineFilter( Value : Single) : Single
 Function BellFilter( Value : Single) : Single
 Function TriangleFilter( Value : Single) : Single
 Function BoxFilter( Value : Single) : Single
 Function HermiteFilter( Value : Single) : Single
 Function Lanczos3Filter( Value : Single) : Single
 Function MitchellFilter( Value : Single) : Single
end;


(*----------------------------------------------------------------------------*)
procedure SIRegister_Chart(CL: TPSPascalCompiler);
begin
 'TeeMsg_DefaultFunctionName','String 'TeeFunction
```

```
TeeMsg_DefaultSeriesName','String 'Series
TeeMsg_DefaultToolName','String 'ChartTool
ChartComponentPalette','String 'TeeChart
TeeMaxLegendColumns',LongInt'( 2);
TeeDefaultLegendSymbolWidth','LongInt'( 20);
TeeTitleFootDistance,LongInt( 5);
 SIRegister_TCustomChartWall(CL);
 SIRegister_TChartWall(CL);
 SIRegister_TChartLegendGradient(CL);
 TLegendStyle', '( lsAuto, lsSeries, lsValues, lsLastValues, lsSeriesGroups )
 TLegendAlignment', '( laLeft, laRight, laTop, laBottom )
 FindClass('TOBJECT'),'LegendException
 TOnGetLegendText', 'Procedure ( Sender : TCustomAxisPanel; Legen'
  +'dStyle : TLegendStyle; Index : Integer; var LegendText : String)
 FindClass('TOBJECT'),'TCustomChartLegend
 TLegendSymbolSize', '( lcsPercent, lcsPixels )
 TLegendSymbolPosition', '( spLeft, spRight )
 TSymbolDrawEvent','Procedure(Sender:TObject;Series:TChartSeries;ValueIndex:Integer;R:TRect);
 TSymbolCalcHeight', 'Function  : Integer
 SIRegister_TLegendSymbol(CL);
 SIRegister_TTeeCustomShapePosition(CL);
 TCheckBoxesStyle', '( cbsCheck, cbsRadio )
 SIRegister_TLegendTitle(CL);
 SIRegister_TLegendItem(CL);
 SIRegister_TLegendItems(CL);
 TLegendCalcSize', 'Procedure ( Sender : TCustomChartLegend; var ASize : Integer)
 FindClass('TOBJECT'),'TCustomChart
 SIRegister_TCustomChartLegend(CL);
 SIRegister_TChartLegend(CL);
 SIRegister_TChartTitle(CL);
 SIRegister_TChartFootTitle(CL);
 TChartClick', 'Procedure ( Sender : TCustomChart; Button : TMous'
  +'eButton; Shift : TShiftState; X, Y : Integer)
 TChartClickAxis', 'Procedure ( Sender : TCustomChart; Axis : TCh'
  +'artAxis; Button : TMouseButton; Shift : TShiftState; X, Y : Integer)
 TChartClickSeries', 'Procedure ( Sender : TCustomChart; Series :'
  +TChartSeries; ValueIndex : Integer; Button: TMouseButton; Shift:TShiftState;X,Y:Integer)
 TChartClickTitle', 'Procedure ( Sender : TCustomChart; ATitle : '
  +'TChartTitle; Button : TMouseButton; Shift : TShiftState; X, Y : Integer)
 TOnGetLegendPos', 'Procedure (Sender: TCustomChart; Index: Integer; var X,Y,XColor:Integer)
 TOnGetLegendRect', 'Procedure ( Sender : TCustomChart; var Rect : TRect)
 TAxisSavedScales', 'record Auto : Boolean; AutoMin : Boolean; Au'
  +'toMax : Boolean; Min : Double; Max : Double; end
 TAllAxisSavedScales', 'array of TAxisSavedScales
 SIRegister_TChartBackWall(CL);
 SIRegister_TChartRightWall(CL);
 SIRegister_TChartBottomWall(CL);
 SIRegister_TChartLeftWall(CL);
 SIRegister_TChartWalls(CL);
 TChartAllowScrollEvent','Procedure(Sender:TChartAxis;var AMin,AMax:Double;var AllowScroll:Boolean);
 SIRegister_TCustomChart(CL);
 SIRegister_TChart(CL);
 SIRegister_TTeeSeriesTypes(CL);
 SIRegister_TTeeToolTypes(CL);
 SIRegister_TTeeDragObject(CL);
 SIRegister_TColorPalettes(CL);
Procedure RegisterTeeSeries(ASeriesClass:TChartSeriesClass;ADescription,
AGalleryPage:PString;ANumGallerySeries:Integer;
Procedure RegisterTeeSeries1( ASeriesClass : TChartSeriesClass; ADescription : PString);
Procedure RegisterTeeFunction(AFunctClass:TTeeFunctionClass;ADescription,
AGalleryPage:PString;ANumGallerySeries: Int;
Procedure RegisterTeeBasicFunction( AFunctionClass : TTeeFunctionClass; ADescription : PString)
Procedure RegisterTeeSeriesFunction(ASeriesClass: TChartSeriesClass;AFunctionClass:TTeeFunctionClass;
ADescription, AGalleryPage : PString; ANumGallerySeries : Integer; ASubIndex : Integer)
Procedure UnRegisterTeeSeries( const ASeriesList : array of TChartSeriesClass)
Procedure UnRegisterTeeFunctions( const AFunctionList : array of TTeeFunctionClass)
Procedure AssignSeries( var OldSeries, NewSeries : TChartSeries)
Function CreateNewTeeFunction( ASeries : TChartSeries; AClass : TTeeFunctionClass) : TTeeFunction
Function CreateNewSeries( AOwner : TComponent; AChart : TCustomAxisPanel; AClass : TChartSeriesClass;
AFunctionClass : TTeeFunctionClass) : TChartSeries
Function CloneChartSeries( ASeries : TChartSeries) : TChartSeries;
Function CloneChartSeries1( ASeries : TChartSeries; AChart : TCustomAxisPanel) : TChartSeries;
Function CloneChartSeries2(ASeries:TChartSeries;AOwner:TComponent;AChart:TCustomAxisPanel):TChartSeries;;
Function CloneChartTool( ATool : TTeeCustomTool; AOwner : TComponent) : TTeeCustomTool
Function ChangeSeriesType( var ASeries : TChartSeries; NewType : TChartSeriesClass) : TChartSeries
Procedure ChangeAllSeriesType( AChart : TCustomChart; AClass : TChartSeriesClass)
Function GetNewSeriesName( AOwner : TComponent) : TComponentName
Procedure RegisterTeeTools( const ATools : array of TTeeCustomToolClass)
Procedure UnRegisterTeeTools( const ATools : array of TTeeCustomToolClass)
Function GetGallerySeriesName( ASeries : TChartSeries) : String
Procedure PaintSeriesLegend(ASeries:TChartSeries; ACanvas:TCanvas; const R:TRect;ReferenceChart:
TCustomChart);
 SIRegister_TChartTheme(CL);
 //TChartThemeClass', 'class of TChartTheme
 //TCanvasClass', 'class of TCanvas3D
Function SeriesNameOrIndex( ASeries : TCustomChartSeries) : String
Function SeriesTitleOrName( ASeries : TCustomChartSeries) : String
Procedure FillSeriesItems( AItems : TStrings; AList : TCustomSeriesList; UseTitles : Boolean)
Procedure ShowMessageUser( const S : String)
```

```
 Function HasNoMandatoryValues( ASeries : TChartSeries) : Boolean
 Function HasLabels( ASeries : TChartSeries) : Boolean
 Function HasColors( ASeries : TChartSeries) : Boolean
 Function SeriesGuessContents( ASeries : TChartSeries) : TeeFormatFlag
 Procedure TeeDrawBitmapEditor( Canvas : TCanvas; Element : TCustomChartElement; Left, Top : Integer)
end;


procedure SIRegister_TeeProcs(CL: TPSPascalCompiler);
begin
 //'TeeFormBorderStyle',' bsNone);
  SIRegister_TMetafile(CL);
 'TeeDefVerticalMargin','LongInt'( 4);
 'TeeDefHorizMargin','LongInt'( 3);
 'crTeeHand','LongInt'( TCursor ( 2020 ));
 'TeeMsg_TeeHand','String 'crTeeHand
 'TeeNormalPrintDetail','LongInt'( 0);
 'TeeHighPrintDetail','LongInt'( - 100);
 'TeeDefault_PrintMargin','LongInt'( 15);
 'MaxDefaultColors','LongInt'( 19);
 'TeeTabDelimiter','Char #9);
  TDateTimeStep', '( dtOneMicroSecond, dtOneMillisecond, dtOneSeco'
   +'nd, dtFiveSeconds, dtTenSeconds, dtFifteenSeconds, dtThirtySeconds, dtOneM'
   +'inute, dtFiveMinutes, dtTenMinutes, dtFifteenMinutes, dtThirtyMinutes, dtO'
   +'neHour, dtTwoHours, dtSixHours, dtTwelveHours, dtOneDay, dtTwoDays, dtThre'
   +'eDays, dtOneWeek, dtHalfMonth, dtOneMonth, dtTwoMonths, dtThreeMonths, dtF'
   +'ourMonths, dtSixMonths, dtOneYear, dtNone )
  SIRegister_TCustomPanelNoCaption(CL);
  FindClass('TOBJECT'),'TCustomTeePanel
  SIRegister_TZoomPanning(CL);
  SIRegister_TTeeEvent(CL);
  //SIRegister_TTeeEventListeners(CL);
  TTeeMouseEventKind', '( meDown, meUp, meMove )
  SIRegister_TTeeMouseEvent(CL);
  SIRegister_TCustomTeePanel(CL);
  //TChartGradient', 'TTeeGradient
  //TChartGradientClass', 'class of TChartGradient
  TPanningMode', '( pmNone, pmHorizontal, pmVertical, pmBoth )
  SIRegister_TTeeZoomPen(CL);
  SIRegister_TTeeZoomBrush(CL);
  TTeeZoomDirection', '( tzdHorizontal, tzdVertical, tzdBoth )
  SIRegister_TTeeZoom(CL);
  FindClass('TOBJECT'),'TCustomTeePanelExtended
  TTeeBackImageMode', '( pbmStretch, pbmTile, pbmCenter, pbmCustom )
  SIRegister_TBackImage(CL);
  SIRegister_TCustomTeePanelExtended(CL);
  //TChartBrushClass', 'class of TChartBrush
  SIRegister_TTeeCustomShapeBrushPen(CL);
  TChartObjectShapeStyle', '( fosRectangle, fosRoundRectangle, fosEllipse )
  TTextFormat', '( ttfNormal, ttfHtml )
  SIRegister_TTeeCustomShape(CL);
  SIRegister_TTeeShape(CL);
  SIRegister_TTeeExportData(CL);
 Function TeeStr( const Num : Integer) : String
 Function DateTimeDefaultFormat( const AStep : Double) : String
 Function TEEDaysInMonth( Year, Month : Word) : Word
 Function FindDateTimeStep( const StepValue : Double) : TDateTimeStep
 Function NextDateTimeStep( const AStep : Double) : Double
 Function PointInLine( const P : TPoint; const px, py, qx, qy : Integer) : Boolean;
 Function PointInLine1( const P, FromPoint, ToPoint : TPoint) : Boolean;
 Function PointInLine2(const P,FromPoint,ToPoint:TPoint;const TolerancePixels:Integer):Boolean;
 Function PointInLine3( const P : TPoint; const px, py, qx, qy, TolerancePixels:Integer):Boolean;
 Function PointInLineTolerance(const P:TPoint;const px,py,qx,qy,TolerancePixels:Integer):Boolean;
 Function PointInPolygon( const P : TPoint; const Poly : array of TPoint) : Boolean
 Function PointInTriangle( const P, P0, P1, P2 : TPoint) : Boolean;
 Function PointInTriangle1( const P : TPoint; X0, X1, Y0, Y1 : Integer) : Boolean;
 Function PointInHorizTriangle( const P : TPoint; Y0, Y1, X0, X1 : Integer) : Boolean
 Function PointInEllipse( const P : TPoint; const Rect : TRect) : Boolean;
 Function PointInEllipse1( const P: TPoint;Left,Top,Right, Bottom : Integer) : Boolean;
 Function DelphiToLocalFormat( const Format : String) : String
 Function LocalToDelphiFormat( const Format : String) : String
 Procedure TEEEnableControls(Enable: Boolean; const ControlArray : array of TControl)
 Function TeeRoundDate(const ADate : TDateTime; AStep : TDateTimeStep) : TDateTime
 Procedure TeeDateTimeIncrement(IsDateTime:Boolean;Increment:Boolean;var Value:Double;const
AnIncrement:Double; tmpWhichDateTime:TDateTimeStep)
  TTeeSortCompare', 'Function ( a, b : Integer) : Integer
  TTeeSortSwap', 'Procedure ( a, b : Integer)
 Procedure TeeSort(StartIndex,EndIndex:Integer;CompareFunc:TTeeSortCompare;SwapFunc:TTeeSortSwap);
 Function TeeGetUniqueName( AOwner : TComponent; const AStartName : String) : string
 Function TeeExtractField( St : String; Index : Integer) : String;
 Function TeeExtractField1( St : String; Index : Integer; const Separator : String) : String;
 Function TeeNumFields( St : String) : Integer;
 Function TeeNumFields1( const St, Separator : String) : Integer;
 Procedure TeeGetBitmapEditor( AObject : TObject; var Bitmap : TBitmap)
 Procedure TeeLoadBitmap( Bitmap : TBitmap; const Name1, Name2 : String)
 // TColorArray', 'array of TColor
 Function GetDefaultColor( const Index : Integer) : TColor
 Procedure SetDefaultColorPalette;
 Procedure SetDefaultColorPalette1( const Palette : array of TColor);
 'TeeCheckBoxSize','LongInt'( 11);
 Procedure TeeDrawCheckBox(x,y:Integer;Canvas:TCanvas;Checked:Boolean;ABackColor:TColor;CheckBox:Boolean);
 Function TEEStrToFloatDef( const S : string; const Default : Extended) : Extended
 Function TryStrToFloat( const S : String; var Value : Double) : Boolean
 Function CrossingLines( const X1, Y1, X2, Y2, X3, Y3, X4, Y4 : Double; out x, y : Double) : Boolean
 Procedure TeeTranslateControl( AControl : TControl);
 Procedure TeeTranslateControl1( AControl : TControl; const ExcludeChilds : array of TControl);
 Function ReplaceChar( const AString : String; const Search : Char; const Replace : Char) : String
 //Procedure RectToFourPoints( const ARect : TRect; const Angle : Double; var P : TFourPoints)
 Function TeeAntiAlias( Panel : TCustomTeePanel; ChartRect : Boolean) : TBitmap
 //Procedure DrawBevel(Canvas:TTeeCanvas;Bevel:TPanelBevel;var R:TRect;Width:Integer;Round:Integer);
 //Function ScreenRatio( ACanvas : TCanvas3D) : Double
 Function TeeReadBoolOption( const AKey : String; DefaultValue : Boolean) : Boolean
 Procedure TeeSaveBoolOption( const AKey : String; Value : Boolean)
 Function TeeReadIntegerOption( const AKey : String; DefaultValue : Integer) : Integer
 Procedure TeeSaveIntegerOption( const AKey : String; Value : Integer)
 Function TeeReadStringOption( const AKey, DefaultValue : String) : String
 Procedure TeeSaveStringOption( const AKey, Value : String)
 Function TeeDefaultXMLEncoding : String
 Procedure ConvertTextToXML( Stream : TStream; XMLHeader : Boolean)
  TeeWindowHandle', 'Integer
 Procedure TeeGotoURL( Handle : TeeWindowHandle; const URL : String)
 Procedure HtmlTextOut( ACanvas : TCanvas; x, y : Integer; Text : String)
 Function HtmlTextExtent( ACanvas : TCanvas; const Text : String) : TSize
end;


using mXBDEUtils
*******************************************************************************
 Procedure SetAlias( aAlias, aDirectory : String)
 Procedure CheckRegistryEntry(Reg:TRegistry;const Path,Value:String;const Default,
Desired:Variant;Size:Byte);
 Function GetFileVersionNumber( const FileName : String) : TVersionNo
 Procedure SetBDE( aPath, aNode, aValue : String)
 function RestartDialog(Wnd: HWnd; Reason: PChar; Flags: Integer): Integer; stdcall;
 Function GetSystemDirectory( lpBuffer : string; uSize : UINT) : UINT
 Function GetSystemDirectoryW( lpBuffer : pchar; uSize : UINT) : UINT
 Function GetTempPath( nBufferLength : DWORD; lpBuffer : string) : DWORD
 Function GetWindowsDirectoryW( nBufferLength : DWORD; lpBuffer : string) : DWORD
 Function GetTempFileName(lpPathName,lpPrefixString:string;uUnique:UINT;lpTempFileName:string):UINT;


procedure SIRegister_cDateTime(CL: TPSPascalCompiler);
begin
AddClassN(FindClass('TOBJECT'),'EDateTime
Function DatePart( const D : TDateTime) : Integer
Function TimePart( const D : TDateTime) : Double
Function Century( const D : TDateTime) : Word
Function Year( const D : TDateTime) : Word
Function Month( const D : TDateTime) : Word
Function Day( const D : TDateTime) : Word
Function Hour( const D : TDateTime) : Word
Function Minute( const D : TDateTime) : Word
Function Second( const D : TDateTime) : Word
Function Millisecond( const D : TDateTime) : Word
('OneDay','Extended').setExtended( 1.0);
('OneHour','Extended').SetExtended( OneDay / 24);
('OneMinute','Extended').SetExtended( OneHour / 60);
('OneSecond','Extended').SetExtended( OneMinute / 60);
('OneMillisecond','Extended').SetExtended( OneSecond / 1000);
('OneWeek','Extended').SetExtended( OneDay * 7);
('HoursPerDay','Extended').SetExtended( 24);
('MinutesPerHour','Extended').SetExtended( 60);
('SecondsPerMinute','Extended').SetExtended( 60);
Procedure SetYear( var D : TDateTime; const Year : Word)
Procedure SetMonth( var D : TDateTime; const Month : Word)
Procedure SetDay( var D : TDateTime; const Day : Word)
Procedure SetHour( var D : TDateTime; const Hour : Word)
Procedure SetMinute( var D : TDateTime; const Minute : Word)
Procedure SetSecond( var D : TDateTime; const Second : Word)
Procedure SetMillisecond( var D : TDateTime; const Milliseconds : Word)
Function IsEqual( const D1, D2 : TDateTime) : Boolean;
Function IsEqual1( const D1 : TDateTime; const Ye, Mo, Da : Word):Boolean;
Function IsEqual2( const D1 : TDateTime; const Ho, Mi, Se, ms : Word):Boolean;
Function IsAM( const D : TDateTime) : Boolean
Function IsPM( const D : TDateTime) : Boolean
Function IsMidnight( const D : TDateTime) : Boolean
Function IsNoon( const D : TDateTime) : Boolean
Function IsSunday( const D : TDateTime) : Boolean
Function IsMonday( const D : TDateTime) : Boolean
Function IsTuesday( const D : TDateTime) : Boolean
Function IsWedneday( const D : TDateTime) : Boolean
Function IsThursday( const D : TDateTime) : Boolean
Function IsFriday( const D : TDateTime) : Boolean
Function IsSaturday( const D : TDateTime) : Boolean
Function IsWeekend( const D : TDateTime) : Boolean
Function Noon( const D : TDateTime) : TDateTime
Function Midnight( const D : TDateTime) : TDateTime
Function FirstDayOfMonth( const D : TDateTime) : TDateTime
Function LastDayOfMonth( const D : TDateTime) : TDateTime
Function NextWorkday( const D : TDateTime) : TDateTime
Function PreviousWorkday( const D : TDateTime) : TDateTime
Function FirstDayOfYear( const D : TDateTime) : TDateTime
Function LastDayOfYear( const D : TDateTime) : TDateTime
Function EasterSunday( const Year : Word) : TDateTime
Function GoodFriday( const Year : Word) : TDateTime
Function AddMilliseconds( const D : TDateTime; const N : Int64) : TDateTime
Function AddSeconds( const D : TDateTime; const N : Int64) : TDateTime
Function AddMinutes( const D : TDateTime; const N : Integer) : TDateTime
Function AddHours( const D : TDateTime; const N : Integer) : TDateTime
Function AddDays( const D : TDateTime; const N : Integer) : TDateTime
Function AddWeeks( const D : TDateTime; const N : Integer) : TDateTime
Function AddMonths( const D : TDateTime; const N : Integer) : TDateTime
Function AddYears( const D : TDateTime; const N : Integer) : TDateTime
Function DayOfYear( const Ye, Mo, Da : Word) : Integer
Function DayOfYear( const D : TDateTime) : Integer
Function DaysInMonth( const Ye, Mo : Word) : Integer
Function DaysInMonth( const D : TDateTime) : Integer
Function DaysInYear( const Ye : Word) : Integer
Function DaysInYearDate( const D : TDateTime) : Integer
Function WeekNumber( const D : TDateTime) : Integer
Function ISOFirstWeekOfYear( const Ye : Word) : TDateTime
Procedure ISOWeekNumber( const D : TDateTime; var WeekNumber, WeekYear : Word)
Function DiffMilliseconds( const D1, D2 : TDateTime) : Int64
Function DiffSeconds( const D1, D2 : TDateTime) : Integer
Function DiffMinutes( const D1, D2 : TDateTime) : Integer
Function DiffHours( const D1, D2 : TDateTime) : Integer
Function DiffDays( const D1, D2 : TDateTime) : Integer
Function DiffWeeks( const D1, D2 : TDateTime) : Integer
Function DiffMonths( const D1, D2 : TDateTime) : Integer
Function DiffYears( const D1, D2 : TDateTime) : Integer
Function GMTBias : Integer
Function GMTTimeToLocalTime( const D : TDateTime) : TDateTime
Function LocalTimeToGMTTime( const D : TDateTime) : TDateTime
Function NowAsGMTTime : TDateTime
Function DateTimeToISO8601String( const D : TDateTime) : AnsiString
Function ISO8601StringToTime( const D : AnsiString) : TDateTime
Function ISO8601StringAsDateTime( const D : AnsiString) : TDateTime
Function DateTimeToANSI( const D : TDateTime) : Integer
Function ANSIToDateTime( const Julian : Integer) : TDateTime
Function DateTimeToISOInteger( const D : TDateTime) : Integer
Function DateTimeToISOString( const D : TDateTime) : AnsiString
Function ISOIntegerToDateTime( const ISOInteger : Integer) : TDateTime
Function TwoDigitRadix2000YearToYear( const Y : Integer) : Integer
Function DateTimeAsElapsedTime(const D:TDateTime; const IncludeMilliseconds:Boolean):AnsiString
Function UnixTimeToDateTime( const UnixTime : LongWord) : TDateTime
Function DateTimeToUnixTime( const D : TDateTime) : LongWord
Function EnglishShortDayOfWeekStrA( const DayOfWeek : Integer) : AnsiString
Function EnglishShortDayOfWeekStrU( const DayOfWeek : Integer) : UnicodeString
Function EnglishLongDayOfWeekStrA( const DayOfWeek : Integer) : AnsiString
Function EnglishLongDayOfWeekStrU( const DayOfWeek : Integer) : UnicodeString
Function EnglishShortMonthStrA( const Month : Integer) : AnsiString
Function EnglishShortMonthStrU( const Month : Integer) : UnicodeString
Function EnglishLongMonthStrA( const Month : Integer) : AnsiString
Function EnglishLongMonthStrU( const Month : Integer) : UnicodeString
Function EnglishShortDayOfWeekA( const S : AnsiString) : Integer
Function EnglishShortDayOfWeekU( const S : UnicodeString) : Integer
Function EnglishLongDayOfWeekA( const S : AnsiString) : Integer
Function EnglishLongDayOfWeekU( const S : UnicodeString) : Integer
Function EnglishShortMonthA( const S : AnsiString) : Integer
Function EnglishShortMonthU( const S : UnicodeString) : Integer
Function EnglishLongMonthA( const S : AnsiString) : Integer
Function EnglishLongMonthU( const S : UnicodeString) : Integer
Function RFC850DayOfWeekA( const S : AnsiString) : Integer
Function RFC850DayOfWeekU( const S : UnicodeString) : Integer
Function RFC1123DayOfWeekA( const S : AnsiString) : Integer
Function RFC1123DayOfWeekU( const S : UnicodeString) : Integer
Function RFCMonthA( const S : AnsiString) : Word
Function RFCMonthU( const S : UnicodeString) : Word
Function GMTTimeToRFC1123TimeA( const D : TDateTime; const IncludeSeconds:Boolean) : AnsiString
Function GMTTimeToRFC1123TimeU( const D : TDateTime; const IncludeSeconds:Boolean) : UnicodeString
Function GMTDateTimeToRFC1123DateTimeA(const D: TDateTime; const IncludeDayOfWeek:Bool):AnsiString;
Function GMTDateTimeToRFC1123DateTimeU(const D:TDateTime;const IncludeDayOfWeek:Bool):UnicodeString;
Function DateTimeToRFCDateTimeA( const D : TDateTime) : AnsiString
Function DateTimeToRFCDateTimeU( const D : TDateTime) : UnicodeString
Function NowAsRFCDateTimeA : AnsiString
Function NowAsRFCDateTimeU : UnicodeString
Function RFCDateTimeToGMTDateTime( const S : AnsiString) : TDateTime
Function RFCDateTimeToDateTime( const S : AnsiString) : TDateTime
Function RFCTimeZoneToGMTBias( const Zone : AnsiString) : Integer
Function TimePeriodStr( const D : TDateTime) : AnsiString
Procedure SelfTest
end;
//*******************************************CFileUtils
Function PathHasDriveLetterA( const Path : AnsiString) : Boolean
Function PathHasDriveLetter( const Path : String) : Boolean
Function PathIsDriveLetterA( const Path : AnsiString) : Boolean
Function PathIsDriveLetter( const Path : String) : Boolean
Function PathIsDriveRootA( const Path : AnsiString) : Boolean
Function PathIsDriveRoot( const Path : String) : Boolean
Function PathIsRootA( const Path : AnsiString) : Boolean
Function PathIsRoot( const Path : String) : Boolean
Function PathIsUNCPathA( const Path : AnsiString) : Boolean
Function PathIsUNCPath( const Path : String) : Boolean
Function PathIsAbsoluteA( const Path : AnsiString) : Boolean
Function PathIsAbsolute( const Path : String) : Boolean
Function PathIsDirectoryA( const Path : AnsiString) : Boolean
Function PathIsDirectory( const Path : String) : Boolean
Function PathInclSuffixA( const Path : AnsiString; const PathSep : Char) : AnsiString
Function PathInclSuffix( const Path : String; const PathSep : Char) : String
Function PathExclSuffixA( const Path : AnsiString; const PathSep : Char) : AnsiString
Function PathExclSuffix( const Path : String; const PathSep : Char) : String
Procedure PathEnsureSuffixA( var Path : AnsiString; const PathSep : Char)
Procedure PathEnsureSuffix( var Path : String; const PathSep : Char)
Procedure PathEnsureNoSuffixA( var Path : AnsiString; const PathSep : Char)
Procedure PathEnsureNoSuffix( var Path : String; const PathSep : Char)
//Function PathCanonicalA( const Path : AnsiString; const PathSep : Char) : AnsiString
Function PathCanonical( const Path : String; const PathSep : Char) : String
Function PathExpandA(const Path:AnsiString;const BasePath:AnsiString;const PathSep:Char):AnsiString
Function PathExpand( const Path : String; const BasePath : String; const PathSep : Char) : String
Function PathLeftElementA( const Path : AnsiString; const PathSep : Char) : AnsiString
Function PathLeftElement( const Path : String; const PathSep : Char) : String
Procedure PathSplitLeftElementA(const Path:AString;var LeftElement,RightPath:AString;const PathSep:Char);
Procedure PathSplitLeftElement(const Path:String; var LeftElement,RightPath: String;const PathSep:Char);
Procedure DecodeFilePathA(const FilePath:AnsiString; var Path,FileName:AnsiString;const PathSep:Char;
Procedure DecodeFilePath( const FilePath : String; var Path, FileName : String; const PathSep : Char)
Function FileNameValidA( const FileName : AnsiString) : AnsiString
Function FileNameValid( const FileName : String) : String
Function FilePathA(const FileName,Path:AnsiString;const BasePath:AnsiStr;const PathSep:Char):AnsiString;
Function FilePath(const FileName, Path: String;const BasePath: String;const PathSep : Char) : String
Function DirectoryExpandA(const Path:AnsiString;const BasePath:AnsiString;const PathSep:Char):AnsiString
Function DirectoryExpand(const Path: String; const BasePath: String; const PathSep : Char) : String
Function UnixPathToWinPath( const Path : AnsiString) : AnsiString
Function WinPathToUnixPath( const Path : AnsiString) : AnsiString
Procedure CCopyFile( const FileName, DestName : String)
Procedure CMoveFile( const FileName, DestName : String)
Function CDeleteFiles( const FileMask : String) : Boolean
Function FileSeekEx(const FHandle:TFileHandle;const FileOffset:Int64; const FilePos:TFileSeekPos):Int64;
Procedure FileCloseEx( const FileHandle : TFileHandle)
Function FileExistsA( const FileName : AnsiString) : Boolean
 Function CFileExists( const FileName : String) : Boolean
 Function CFileGetSize( const FileName : String) : Int64
Function FileGetDateTime( const FileName : String) : TDateTime
Function FileGetDateTime2( const FileName : String) : TDateTime
Function FileIsReadOnly( const FileName : String) : Boolean
Procedure FileDeleteEx( const FileName : String)
Procedure FileRenameEx( const OldFileName, NewFileName : String)
Function ReadFileStrA( const FileName:AnsiString; const FileSharing : TFileSharing; const FileCreationMode
: TFileCreationMode; const FileOpenWait : PFileOpenWait) : AnsiString
Function DirectoryEntryExists( const Name : String) : Boolean
Function DirectoryEntrySize( const Name : String) : Int64
 Function CDirectoryExists( const DirectoryName : String) : Boolean
Function DirectoryGetDateTime( const DirectoryName : String) : TDateTime
Procedure CDirectoryCreate( const DirectoryName : String)
Function GetFirstFileNameMatching( const FileMask : String) : String
Function DirEntryGetAttr( const FileName : AnsiString) : Integer
Function DirEntryIsDirectory( const FileName : AnsiString) : Boolean
Function FileHasAttr( const FileName : String; const Attr : Word) : Boolean
  AddTypeS('TLogicalDriveType', '( DriveRemovable, DriveFixed, DriveRemote, '
   +'DriveCDRom, DriveRamDisk, DriveTypeUnknown )
Function DriveIsValid( const Drive : Char) : Boolean
Function DriveGetType( const Path : AnsiString) : TLogicalDriveType
Function DriveFreeSpace( const Path : AnsiString) : Int64

procedure SIRegister_cTimers(CL: TPSPascalCompiler);
begin
 AddClassN(FindClass('TOBJECT'),'ETimers
 Const('TickFrequency','LongInt'( 1000);Function GetTick : LongWord
Function TickDelta( const D1, D2 : LongWord) : Integer
Function TickDeltaW( const D1, D2 : LongWord) : LongWord
  AddTypeS('THPTimer', 'Int64
Procedure StartTimer( var Timer : THPTimer)
Procedure StopTimer( var Timer : THPTimer)
Procedure ResumeTimer( var StoppedTimer : THPTimer)
Procedure InitStoppedTimer( var Timer : THPTimer)
Procedure InitElapsedTimer( var Timer : THPTimer; const Milliseconds : Integer)
Function MillisecondsElapsed( const Timer: THPTimer; const TimerRunning : Boolean) : Integer
Function MicrosecondsElapsed( const Timer: THPTimer; const TimerRunning : Boolean) : Int64
Procedure WaitMicroseconds( const MicroSeconds : Integer)
Function GetHighPrecisionFrequency : Int64
Function GetHighPrecisionTimerOverhead : Int64
Procedure AdjustTimerForOverhead( var StoppedTimer : THPTimer; const Overhead : Int64)
Procedure SelfTestCTimer
end;

procedure SIRegister_cRandom(CL: TPSPascalCompiler);
begin
```

```
 Function RandomSeed : LongWord
 Procedure AddEntropy( const Value : LongWord)
 Function RandomUniform : LongWord;
 Function RandomUniform1( const N : Integer) : Integer;
 Function RandomBoolean : Boolean
 Function RandomByte : Byte
 Function RandomByteNonZero : Byte
 Function RandomWord : Word
 Function RandomInt64 : Int64;
 Function RandomInt641( const N : Int64) : Int64;
 Function RandomHex( const Digits : Integer) : String
 Function RandomFloat : Extended
 Function RandomAlphaStr( const Length : Integer) : AnsiString
 Function RandomPseudoWord( const Length : Integer) : AnsiString
 Function RandomPassword(const MinL,MaxLength:Int;const CaseSens,UseSymbols,UseNumbers:Bool):AnsiString;
 Function mwcRandomLongWord : LongWord
 Function urnRandomLongWord : LongWord
 Function moaRandomFloat : Extended
 Function mwcRandomFloat : Extended
 Function RandomNormalF : Extended
 Procedure SelfTestCRandom
end;

procedure SIRegister_SynEditMiscProcs(CL: TPSPascalCompiler);
begin
 // PIntArray', '^TIntArray // will not work
 Addtypes('TConvertTabsProc','function(const Line:AnsiString; TabWidth: integer):AnsiString
 Addtypes('TConvertTabsProcEx','function(const Line:AnsiString; TabWidth: integer;var HasTabs: boolean):
AnsiString
 Function synMax( x, y : integer) : integer
 Function synMin( x, y : integer) : integer
 Function synMinMax( x, mi, ma : integer) : integer
 Procedure synSwapInt( var l, r : integer)
 Function synMaxPoint( const P1, P2 : TPoint) : TPoint
 Function synMinPoint( const P1, P2 : TPoint) : TPoint
 //Function synGetIntArray( Count : Cardinal; InitialValue : integer) : PIntArray
 Procedure synInternalFillRect( dc : HDC; const rcPaint : TRect)
 Function synGetBestConvertTabsProc( TabWidth : integer) : TConvertTabsProc
 Function synConvertTabs( const Line : AnsiString; TabWidth : integer) : AnsiString
 Function synGetBestConvertTabsProcEx( TabWidth : integer) : TConvertTabsProcEx
 Function synConvertTabsEx(const Line:AnsiString;TabWidth:integer; var HasTabs:boolean):AnsiString;
 Function synGetExpandedLength( const aStr : string; aTabWidth : integer) : integer
 Function synCharIndex2CaretPos( Index, TabWidth : integer; const Line : string) : integer
 Function synCaretPos2CharIndex(Position,TabWidth:int;const Line:string;var InsideTabChar:boolean):int;
 Function synStrScanForCharInSet(const Line:string;Start:integer;AChars:TSynIdentChars):integer;
 Function synStrRScanForCharInSet(const Line:string;Start:integer;AChars:TSynIdentChars):integer;
  TStringType', '( stNone, stHalfNumAlpha, stHalfSymbol, stHalfKat'
   +'akana,stWideNumAlpha,stWideSymbol,stWideKatakana,stHiragana,stIdeograph,stControl,stKashida )
 ('C3_NONSPACING','LongInt'( 1);
 'C3_DIACRITIC','LongInt'( 2);
 'C3_VOWELMARK','LongInt'( 4);
 ('C3_SYMBOL','LongInt'( 8);
 ('C3_KATAKANA','LongWord( $0010);
 ('C3_HIRAGANA','LongWord( $0020);
 ('C3_HALFWIDTH','LongWord( $0040);
 ('C3_FULLWIDTH','LongWord( $0080);
 ('C3_IDEOGRAPH','LongWord( $0100);
 ('C3_KASHIDA','LongWord( $0200);
 ('C3_LEXICAL','LongWord( $0400);
 ('C3_ALPHA','LongWord( $8000);
 ('C3_NOTAPPLICABLE','LongInt'( 0);
 Function synStrScanForMultiByteChar( const Line : string; Start : Integer) : Integer
 Function synStrRScanForMultiByteChar( const Line : string; Start : Integer) : Integer
 Function synIsStringType( Value : Word) : TStringType
 Function synGetEOL( Line : PChar) : PChar
 Function synEncodeString( s : string) : string
 Function synDecodeString( s : string) : string
 Procedure synFreeAndNil( var Obj: TObject)
 Procedure synAssert( Expr : Boolean)
 Function synLastDelimiter( const Delimiters, S : string) : Integer
  TReplaceFlag', '( rfReplaceAll, rfIgnoreCase )
  TReplaceFlags', 'set of TReplaceFlag )
 Function synStringReplace(const S, OldPattern, NewPattern : string; Flags: TReplaceFlags) : string
 Function synGetRValue( RGBValue : TColor) : byte
 Function synGetGValue( RGBValue : TColor) : byte
 Function synGetBValue( RGBValue : TColor) : byte
 Function synRGB( r, g, b : Byte) : Cardinal
//  THighlighterAttriProc', 'Function ( Highlighter : TSynCustomHigh'
  // +'lighter; Attri:TSynHighlighterAttributes;UniqueAttriName:string;Params array of Pointer):Boolean;
 //Function synEnumHighlighterAttris( Highlighter : TSynCustomHighlighter; SkipDuplicates : Boolean;
HighlighterAttriProc : THighlighterAttriProc; Params : array of Pointer) : Boolean
 Function synCalcFCS( const ABuf, ABufSize : Cardinal) : Word
 Procedure synSynDrawGradient(const ACanvas:TCanvas;const AStartColor,
AEndColor:TColor;ASteps:integer;const ARect : TRect; const AHorizontal : boolean)
end;

 Function GET_APPCOMMAND_LPARAM( lParam : LPARAM) : WORD
 Function GET_DEVICE_LPARAM( lParam : LPARAM) : WORD
 Function GET_KEYSTATE_LPARAM( lParam : LPARAM) : WORD
```

```

procedure SIRegister_synautil(CL: TPSPascalCompiler);
begin
 Function STimeZoneBias : integer
 Function TimeZone : string
 Function Rfc822DateTime( t : TDateTime) : string
 Function CDateTime( t : TDateTime) : string
 Function SimpleDateTime( t : TDateTime) : string
 Function AnsiCDateTime( t : TDateTime) : string
 Function GetMonthNumber( Value : String) : integer
 Function GetTimeFromStr( Value : string) : TDateTime
 Function GetDateMDYFromStr( Value : string) : TDateTime
 Function DecodeRfcDateTime( Value : string) : TDateTime
 Function GetUTTime : TDateTime
 Function SetUTTime( Newdt : TDateTime) : Boolean
 Function SGetTick : LongWord
 Function STickDelta( TickOld, TickNew : LongWord) : LongWord
 Function CodeInt( Value : Word) : Ansistring
 Function DecodeInt( const Value : Ansistring; Index : Integer) : Word
 Function CodeLongInt( Value : LongInt) : Ansistring
 Function DecodeLongInt( const Value : Ansistring; Index : Integer) : LongInt
 Function DumpStr( const Buffer : Ansistring) : string
 Function DumpExStr( const Buffer : Ansistring) : string
 Procedure Dump( const Buffer : AnsiString; DumpFile : string)
 Procedure DumpEx( const Buffer : AnsiString; DumpFile : string)
 Function TrimSPLeft( const S : string) : string
 Function TrimSPRight( const S : string) : string
 Function TrimSP( const S : string) : string
 Function SeparateLeft( const Value, Delimiter : string) : string
 Function SeparateRight( const Value, Delimiter : string) : string
 Function SGetParameter( const Value, Parameter : string) : string
 Procedure ParseParametersEx( Value, Delimiter : string; const Parameters : TStrings)
 Procedure ParseParameters( Value : string; const Parameters : TStrings)
 Function IndexByBegin( Value : string; const List : TStrings) : integer
 Function GetEmailAddr( const Value : string) : string
 Function GetEmailDesc( Value : string) : string
 Function CStrToHex( const Value : Ansistring) : string
 Function CIntToBin( Value : Integer; Digits : Byte) : string
 Function CBinToInt( const Value : string) : Integer
 Function ParseURL( URL : string; var Prot, User, Pass, Host, Port, Path, Para:string):string
 Function CReplaceString( Value, Search, Replace : AnsiString) : AnsiString
 Function CRPosEx( const Sub, Value : string; From : integer) : Integer
 Function CRPos( const Sub, Value : String) : Integer
 Function FetchBin( var Value : string; const Delimiter : string) : string
 Function CFetch( var Value : string; const Delimiter : string) : string
 Function FetchEx( var Value : string; const Delimiter, Quotation : string) : string
 Function IsBinaryString( const Value : AnsiString) : Boolean
 Function PosCRLF( const Value : AnsiString; var Terminator : AnsiString) : integer
 Procedure StringsTrim( const value : TStrings)
 Function PosFrom( const SubStr, Value : String; From : integer) : integer
 Function IncPoint( const p : ___pointer; Value : integer) : ___pointer
 Function GetBetween( const PairBegin, PairEnd, Value : string) : string
 Function CCountOfChar( const Value : string; aChr : char) : integer
 Function UnquoteStr( const Value : string; Quote : Char) : string
 Function QuoteStr( const Value : string; Quote : Char) : string
 Procedure HeadersToList( const Value : TStrings)
 Procedure ListToHeaders( const Value : TStrings)
 Function SwapBytes( Value : integer) : integer
 Function ReadStrFromStream( const Stream : TStream; len : integer) : AnsiString
 Procedure WriteStrToStream( const Stream : TStream; Value : AnsiString)
 Function GetTempFile( const Dir, prefix : AnsiString) : AnsiString
 Function CPadString( const Value : AnsiString; len : integer; Pad : AnsiChar): AnsiString
 Function CXorString( Indata1, Indata2 : AnsiString) : AnsiString
 Function NormalizeHeader( Value : TStrings; var Index : Integer) : string
end;

procedure SIRegister_StCRC(CL: TPSPascalCompiler);
begin
 ('CrcBufSize','LongInt'( 2048);
 Function Adler32Prim( var Data, DataSize : Cardinal; CurCrc : LongInt) : LongInt
 Function Adler32OfStream( Stream : TStream; CurCrc : LongInt) : LongInt
 Function Adler32OfFile( FileName : AnsiString) : LongInt
 Function Crc16Prim( var Data, DataSize, CurCrc : Cardinal) : Cardinal
 Function Crc16OfStream( Stream : TStream; CurCrc : Cardinal) : Cardinal
 Function Crc16OfFile( FileName : AnsiString) : Cardinal
 Function Crc32Prim( var Data, DataSize : Cardinal; CurCrc : LongInt) : LongInt
 Function Crc32OfStream( Stream : TStream; CurCrc : LongInt) : LongInt
 Function Crc32OfFile( FileName : AnsiString) : LongInt
 Function InternetSumPrim( var Data, DataSize, CurCrc : Cardinal) : Cardinal
 Function InternetSumOfStream( Stream : TStream; CurCrc : Cardinal) : Cardinal
 Function InternetSumOfFile( FileName : AnsiString) : Cardinal
 Function Kermit16Prim( var Data, DataSize, CurCrc : Cardinal) : Cardinal
 Function Kermit16OfStream( Stream : TStream; CurCrc : Cardinal) : Cardinal
 Function Kermit16OfFile( FileName : AnsiString) : Cardinal
end;

procedure SIRegister_ComObj(cl: TPSPascalCompiler);
begin
 function CreateOleObject(const ClassName: String): IDispatch;
 function GetActiveOleObject(const ClassName: String): IDispatch;
 function ProgIDToClassID(const ProgID: string): TGUID;
 function ClassIDToProgID(const ClassID: TGUID): string;
 function CreateClassID: string;
 function CreateGUIDString: string;
 function CreateGUIDID: string;
 procedure OleError(ErrorCode: longint)
 procedure OleCheck(Result: HResult);
end;

 Function xCreateOleObject( const ClassName : string) : Variant //or IDispatch
 Function xGetActiveOleObject( const ClassName : string) : Variant
 //Function DllGetClassObject( const CLSID : TCLSID; const IID : TIID; var Obj) : HResult
 Function DllCanUnloadNow : HResult
 Function DllRegisterServer : HResult
 Function DllUnregisterServer : HResult
 Function VarFromInterface( Unknown : IUnknown) : Variant
 Function VarToInterface( const V : Variant) : IDispatch
 Function VarToAutoObject( const V : Variant) : TAutoObject
 //Procedure
DispInvoke(Dispatch:IDispatch;CallDesc:PCallDesc;DispIDs:PDispIDList;Params:Pointer;Res:PVariant);
 //Procedure DispInvokeError( Status : HResult; const ExcepInfo : TExcepInfo)
 Procedure OleError( ErrorCode : HResult)
 Procedure OleCheck( Result : HResult)
 Function StringToClassID( const S : string) : TCLSID
 Function ClassIDToString( const ClassID : TCLSID) : string
 Function xProgIDToClassID( const ProgID : string) : TCLSID
 Function xClassIDToProgID( const ClassID : TCLSID) : string
 Function xWideCompareStr( const S1, S2 : WideString) : Integer
 Function xWideSameStr( const S1, S2 : WideString) : Boolean
 Function xGUIDToString( const ClassID : TGUID) : string
 Function xStringToGUID( const S : string) : TGUID
 Function xGetModuleName( Module : HMODULE) : string
 Function xAcquireExceptionObject : TObject
 Function xIfThen( AValue : Boolean; const ATrue : Integer; const AFalse : Integer) : Integer
 Function xUtf8Encode( const WS : WideString) : UTF8String
 Function xUtf8Decode( const S : UTF8String) : WideString
 Function xExcludeTrailingPathDelimiter( const S : string) : string
 Function xIncludeTrailingPathDelimiter( const S : string) : string
 Function XRTLHandleCOMException : HResult
 Procedure XRTLCheckArgument( Flag : Boolean)
 //Procedure XRTLCheckOutArgument( out Arg)
 Procedure XRTLInterfaceConnect(const Source:IUnknown;const IID:TIID;const Sink:IUnknown;var
Connection:Longint);
 Procedure XRTLInterfaceDisconnect(const Source: IUnknown; const IID:TIID;var Connection : Longint)
 Function XRTLRegisterActiveObject(const Unk:IUnknown;ClassID:TCLSID;Flags:DWORD;var
RegisterCookie:Int):HResult
 Function XRTLUnRegisterActiveObject( var RegisterCookie : Integer) : HResult
 //Function XRTLGetActiveObject( ClassID : TCLSID; RIID : TIID; out Obj) : HResult
 Procedure XRTLEnumActiveObjects( Strings : TStrings)
function   XRTLDefaultCategoryManager: IUnknown;
function   XRTLIsCategoryEmpty(CatID: TGUID; const CategoryManager: IUnknown = nil): Boolean;
// ICatRegister helper functions
function   XRTLCreateComponentCategory(CatID: TGUID; CatDescription: WideString;
                                       LocaleID: TLCID = LOCALE_USER_DEFAULT;
                                       const CategoryManager: IUnknown = nil): HResult;
function   XRTLRemoveComponentCategory(CatID: TGUID; CatDescription: WideString;
                                       LocaleID: TLCID = LOCALE_USER_DEFAULT;
                                       const CategoryManager: IUnknown = nil): HResult;
function   XRTLRegisterCLSIDInCategory(ClassID: TGUID; CatID: TGUID;
                                       const CategoryManager: IUnknown = nil): HResult;
function   XRTLUnRegisterCLSIDInCategory(ClassID: TGUID; CatID: TGUID;
                                         const CategoryManager: IUnknown = nil): HResult;
// ICatInformation helper functions
function   XRTLGetCategoryDescription(CatID: TGUID; var CatDescription: WideString;
                                      LocaleID: TLCID = LOCALE_USER_DEFAULT;
                                      const CategoryManager: IUnknown = nil): HResult;
function   XRTLGetCategoryList(Strings: TStrings; LocaleID: TLCID = LOCALE_USER_DEFAULT;
                               const CategoryManager: IUnknown = nil): HResult;
function   XRTLGetCategoryCLSIDList(CatID: TGUID; Strings: TStrings;
                                    const CategoryManager: IUnknown = nil): HResult;
function   XRTLGetCategoryProgIDList(CatID: TGUID; Strings: TStrings;
                                     const CategoryManager: IUnknown = nil): HResult;
function XRTLFetch(var AInput: WideString; const ADelim: WideString = ' ';
                   const ADelete: Boolean = True): WideString;
function XRTLRPos(const ASub, AIn: WideString; AStart: Integer = -1): Integer;
Function XRTLGetVariantAsString( const Value : Variant) : string
Function XRTLDateTimeToTimeZoneTime( DT : TDateTime; TimeZone : TXRTLTimeZone) : TDateTime
Function XRTLGetTimeZones : TXRTLTimeZones
Function XFileTimeToDateTime( FileTime : TFileTime) : TDateTime
Function DateTimeToFileTime( DateTime : TDateTime) : TFileTime
Function GMTNow : TDateTime
Function GMTToLocalTime( GMT : TDateTime) : TDateTime
Function LocalTimeToGMT( LocalTime : TDateTime) : TDateTime
Procedure XRTLNotImplemented
Procedure XRTLRaiseError( E : Exception)
Procedure XRTLInvalidOperation( ClassName:string; OperationName:string; Description: string)


```

```
procedure SIRegister_xrtl_util_Value(CL: TPSPascalCompiler);
begin
  SIRegister_IXRTLValue(CL);
  SIRegister_TXRTLValue(CL);
  //AddTypeS('PXRTLValueArray', '^TXRTLValueArray // will not work
  AddTypeS('TXRTLValueArray', 'array of IXRTLValue
Function XRTLValue( const AValue : Cardinal) : IXRTLValue;
Function XRTLSetValue( const IValue : IXRTLValue; const AValue : Cardinal) : Cardinal;
Function XRTLGetAsCardinal( const IValue : IXRTLValue) : Cardinal
Function XRTLGetAsCardinalDef( const IValue : IXRTLValue; const DefValue : Cardinal) : Cardinal
Function XRTLValue1( const AValue : Integer) : IXRTLValue;
Function XRTLSetValue1( const IValue : IXRTLValue; const AValue : Integer) : Integer;
Function XRTLGetAsInteger( const IValue : IXRTLValue) : Integer
Function XRTLGetAsIntegerDef( const IValue : IXRTLValue; const DefValue : Integer) : Integer
Function XRTLValue2( const AValue : Int64) : IXRTLValue;
Function XRTLSetValue2( const IValue : IXRTLValue; const AValue : Int64) : Int64;
Function XRTLGetAsInt64( const IValue : IXRTLValue) : Int64
Function XRTLGetAsInt64Def( const IValue : IXRTLValue; const DefValue : Int64) : Int64
Function XRTLValue3( const AValue : Single) : IXRTLValue;
Function XRTLSetValue3( const IValue : IXRTLValue; const AValue : Single) : Single;
Function XRTLGetAsSingle( const IValue : IXRTLValue) : Single
Function XRTLGetAsSingleDef( const IValue : IXRTLValue; const DefValue : Single) : Single
Function XRTLValue4( const AValue : Double) : IXRTLValue;
Function XRTLSetValue4( const IValue : IXRTLValue; const AValue : Double) : Double;
Function XRTLGetAsDouble( const IValue : IXRTLValue) : Double
Function XRTLGetAsDoubleDef( const IValue : IXRTLValue; const DefValue : Double) : Double
Function XRTLValue5( const AValue : Extended) : IXRTLValue;
Function XRTLSetValue5( const IValue : IXRTLValue; const AValue : Extended) : Extended;
Function XRTLGetAsExtended( const IValue : IXRTLValue) : Extended
Function XRTLGetAsExtendedDef( const IValue : IXRTLValue; const DefValue : Extended) : Extended
Function XRTLValue6( const AValue : IInterface) : IXRTLValue;
Function XRTLSetValue6( const IValue : IXRTLValue; const AValue : IInterface) : IInterface;
Function XRTLGetAsInterface( const IValue : IXRTLValue) : IInterface;
 //Function XRTLGetAsInterface1( const IValue : IXRTLValue; out Obj) : IInterface;
Function XRTLGetAsInterfaceDef( const IValue : IXRTLValue; const DefValue : IInterface) : IInterface;
Function XRTLValue7( const AValue : WideString) : IXRTLValue;
Function XRTLSetValue7( const IValue : IXRTLValue; const AValue : WideString) : WideString;
Function XRTLGetAsWideString( const IValue : IXRTLValue) : WideString
Function XRTLGetAsWideStringDef( const IValue : IXRTLValue; const DefValue : WideString) : WideString
Function XRTLValue8( const AValue : TObject; const AOwnValue : Boolean) : IXRTLValue;
Function XRTLSetValue8( const IValue : IXRTLValue; const AValue : TObject) : TObject;
Function XRTLGetAsObject( const IValue : IXRTLValue; const ADetachOwnership : Boolean) : TObject;
Function XRTLGetAsObjectDef(const IValue:IXRTLValue;const DefValue:TObject;const
ADetachOwnership:Boolean):TObject;
//Function XRTLValue9( const AValue : __Pointer) : IXRTLValue;
//Function XRTLSetValue9( const IValue : IXRTLValue; const AValue : __Pointer) : __Pointer;
//Function XRTLGetAsPointer( const IValue : IXRTLValue) : __Pointer
//Function XRTLGetAsPointerDef( const IValue : IXRTLValue; const DefValue : __Pointer) : __Pointer
Function XRTLValueV( const AValue : Variant) : IXRTLValue;
Function XRTLSetValueV( const IValue : IXRTLValue; const AValue : Variant) : Variant;
Function XRTLGetAsVariant( const IValue : IXRTLValue) : Variant
Function XRTLGetAsVariantDef( const IValue : IXRTLValue; const DefValue : Variant) : Variant
Function XRTLValue10( const AValue : Currency) : IXRTLValue;
Function XRTLSetValue10( const IValue : IXRTLValue; const AValue : Currency) : Currency;
Function XRTLGetAsCurrency( const IValue : IXRTLValue) : Currency
Function XRTLGetAsCurrencyDef( const IValue : IXRTLValue; const DefValue : Currency) : Currency
Function XRTLValue11( const AValue : Comp) : IXRTLValue;
Function XRTLSetValue11( const IValue : IXRTLValue; const AValue : Comp) : Comp;
Function XRTLGetAsComp( const IValue : IXRTLValue) : Comp
Function XRTLGetAsCompDef( const IValue : IXRTLValue; const DefValue : Comp) : Comp
Function XRTLValue12( const AValue : TClass) : IXRTLValue;
Function XRTLSetValue12( const IValue : IXRTLValue; const AValue : TClass) : TClass;
Function XRTLGetAsClass( const IValue : IXRTLValue) : TClass
Function XRTLGetAsClassDef( const IValue : IXRTLValue; const DefValue : TClass) : TClass
Function XRTLValue13( const AValue : TGUID) : IXRTLValue;
Function XRTLSetValue13( const IValue : IXRTLValue; const AValue : TGUID) : TGUID;
Function XRTLGetAsGUID( const IValue : IXRTLValue) : TGUID
Function XRTLGetAsGUIDDef( const IValue : IXRTLValue; const DefValue : TGUID) : TGUID
Function XRTLValue14( const AValue : Boolean) : IXRTLValue;
Function XRTLSetValue14( const IValue : IXRTLValue; const AValue : Boolean) : Boolean;
Function XRTLGetAsBoolean( const IValue : IXRTLValue) : Boolean
Function XRTLGetAsBooleanDef( const IValue : IXRTLValue; const DefValue : Boolean) : Boolean
end;

//****************************unit uPSI_GR32;*******************************************

 Function Color32( WinColor : TColor) : TColor32;
 Function Color321( R, G, B : Byte; A : Byte) : TColor32;
 Function Color322( Index : Byte; var Palette : TPalette32) : TColor32;
 Function Gray32( Intensity : Byte; Alpha : Byte) : TColor32
 Function WinColor( Color32 : TColor32) : TColor
 Function ArrayOfColor32( Colors : array of TColor32) : TArrayOfColor32
 Procedure Color32ToRGB( Color32 : TColor32; var R, G, B : Byte)
 Procedure Color32ToRGBA( Color32 : TColor32; var R, G, B, A : Byte)
 Function Color32Components( R, G, B, A : Boolean) : TColor32Components
 Function RedComponent( Color32 : TColor32) : Integer
 Function GreenComponent( Color32 : TColor32) : Integer
 Function BlueComponent( Color32 : TColor32) : Integer
 Function AlphaComponent( Color32 : TColor32) : Integer
 Function Intensity( Color32 : TColor32) : Integer
 Function SetAlpha( Color32 : TColor32; NewAlpha : Integer) : TColor32
 Function HSLtoRGB( H, S, L : Single) : TColor32;
 Procedure RGBtoHSL( RGB : TColor32; out H, S, L : Single);
 Function HSLtoRGB1( H, S, L : Integer) : TColor32;
 Procedure RGBtoHSL1( RGB : TColor32; out H, S, L : Byte);
 Function WinPalette( const P : TPalette32) : HPALETTE
 Function FloatPoint( X, Y : Single) : TFloatPoint;
 Function FloatPoint1( const P : TPoint) : TFloatPoint;
 Function FloatPoint2( const FXP : TFixedPoint) : TFloatPoint;
 Function FixedPoint( X, Y : Integer) : TFixedPoint;
 Function FixedPoint1( X, Y : Single) : TFixedPoint;
 Function FixedPoint2( const P : TPoint) : TFixedPoint;
 Function FixedPoint3( const FP : TFloatPoint) : TFixedPoint;
  AddTypeS('TRectRounding', '( rrClosest, rrOutside, rrInside )
 Function MakeRect( const L, T, R, B : Integer) : TRect;
 Function MakeRect1( const FR : TFloatRect; Rounding : TRectRounding) : TRect;
 Function MakeRect2( const FXR : TRect; Rounding : TRectRounding) : TRect;
 Function GFixedRect( const L, T, R, B : TFixed) : TRect;
 Function FixedRect1( const ARect : TRect) : TRect;
 Function FixedRect2( const FR : TFloatRect) : TRect;
 Function GFloatRect( const L, T, R, B : TFloat) : TFloatRect;
 Function FloatRect1( const ARect : TRect) : TFloatRect;
 Function FloatRect2( const FXR : TRect) : TFloatRect;
 Function GIntersectRect( out Dst : TRect; const R1, R2 : TRect) : Boolean;
 Function IntersectRect1( out Dst : TFloatRect; const FR1, FR2 : TFloatRect) : Boolean;
 Function GUnionRect( out Rect : TRect; const R1, R2 : TRect) : Boolean;
 Function UnionRect1( out Rect : TFloatRect; const R1, R2 : TFloatRect) : Boolean;
 Function GEqualRect( const R1, R2 : TRect) : Boolean;
 Function EqualRect1( const R1, R2 : TFloatRect) : Boolean;
 Procedure GInflateRect( var R : TRect; Dx, Dy : Integer);
 Procedure InflateRect1( var FR : TFloatRect; Dx, Dy : TFloat);
 Procedure GOffsetRect( var R : TRect; Dx, Dy : Integer);
 Procedure OffsetRect1( var FR : TFloatRect; Dx, Dy : TFloat);
 Function IsRectEmpty( const R : TRect) : Boolean;
 Function IsRectEmpty1( const FR : TFloatRect) : Boolean;
 Function GPtInRect( const R : TRect; const P : TPoint) : Boolean;
 Function PtInRect1( const R : TFloatRect; const P : TPoint) : Boolean;
 Function PtInRect2( const R : TRect; const P : TFloatPoint) : Boolean;
 Function PtInRect3( const R : TFloatRect; const P : TFloatPoint) : Boolean;
 Function EqualRectSize( const R1, R2 : TRect) : Boolean;
 Function EqualRectSize1( const R1, R2 : TFloatRect) : Boolean;
 Function MessageBeep( uType : UINT) : BOOL
 Function ShowCursor( bShow : BOOL) : Integer
 Function SetCursorPos( X, Y : Integer) : BOOL
 Function SetCursor( hCursor : HICON) : HCURSOR
 Function GetCursorPos( var lpPoint : TPoint) : BOOL
 //Function ClipCursor( lpRect : PRect) : BOOL
 Function GetClipCursor( var lpRect : TRect) : BOOL
 Function GetCursor : HCURSOR
 Function CreateCaret( hWnd : HWND; hBitmap : HBITMAP; nWidth, nHeight : Integer) : BOOL
 Function GetCaretBlinkTime : UINT
 Function SetCaretBlinkTime( uMSeconds : UINT) : BOOL
 Function DestroyCaret : BOOL
 Function HideCaret( hWnd : HWND) : BOOL
 Function ShowCaret( hWnd : HWND) : BOOL
 Function SetCaretPos( X, Y : Integer) : BOOL
 Function GetCaretPos( var lpPoint : TPoint) : BOOL
 Function ClientToScreen( hWnd : HWND; var lpPoint : TPoint) : BOOL
 Function ScreenToClient( hWnd : HWND; var lpPoint : TPoint) : BOOL
 Function MapWindowPoints(hWndFrom,hWndTo:HWND; var lpPoints, cPoints : UINT) : Integer
 Function WindowFromPoint( Point : TPoint) : HWND
 Function ChildWindowFromPoint( hWndParent : HWND; Point : TPoint) : HWND


procedure SIRegister_GR32_Math(CL: TPSPascalCompiler);
begin
 Function FixedFloor( A : TFixed) : Integer
 Function FixedCeil( A : TFixed) : Integer
 Function FixedMul( A, B : TFixed) : TFixed
 Function FixedDiv( A, B : TFixed) : TFixed
 Function OneOver( Value : TFixed) : TFixed
 Function FixedRound( A : TFixed) : Integer
 Function FixedSqr( Value : TFixed) : TFixed
 Function FixedSqrtLP( Value : TFixed) : TFixed
 Function FixedSqrtHP( Value : TFixed) : TFixed
 Function FixedCombine( W, X, Y : TFixed) : TFixed
 Procedure GRSinCos( const Theta : TFloat; out Sin, Cos : TFloat);
 Procedure GRSinCos1( const Theta, Radius : Single; out Sin, Cos : Single);
 Function GRHypot( const X, Y : TFloat) : TFloat;
 Function Hypot1( const X, Y : Integer) : Integer;
 Function FastSqrt( const Value : TFloat) : TFloat
 Function FastSqrtBab1( const Value : TFloat) : TFloat
 Function FastSqrtBab2( const Value : TFloat) : TFloat
 Function FastInvSqrt( const Value : Single) : Single;
 Function MulDiv( Multiplicand, Multiplier, Divisor : Integer) : Integer
 Function GRIsPowerOf2( Value : Integer) : Boolean
 Function PrevPowerOf2( Value : Integer) : Integer
 Function NextPowerOf2( Value : Integer) : Integer
 Function Average( A, B : Integer) : Integer
 Function GRSign( Value : Integer) : Integer
 Function FloatMod( x, y : Double) : Double
end;

procedure SIRegister_GR32_LowLevel(CL: TPSPascalCompiler);
begin
 Function Clamp( const Value : Integer) : Integer;
 Procedure GRFillWord( var X, Count : Cardinal; Value : Longword)
 Function StackAlloc( Size : Integer) : Pointer
 Procedure StackFree( P : Pointer)
 Procedure Swap( var A, B : Pointer);
 Procedure Swap1( var A, B : Integer);
 Procedure Swap2( var A, B : TFixed);
 Procedure Swap3( var A, B : TColor32);
 Procedure TestSwap( var A, B : Integer);
 Procedure TestSwap1( var A, B : TFixed);
 Function TestClip( var A, B : Integer; const Size : Integer) : Boolean;
 Function TestClip1( var A, B : Integer; const Start, Stop : Integer) : Boolean;
 Function GRConstrain( const Value, Lo, Hi : Integer) : Integer;
 Function Constrain1( const Value, Lo, Hi : Single) : Single;
 Function SwapConstrain( const Value:Integer; Constrain1,Constrain2:Integer) : Integer
 Function GRMin( const A, B, C : Integer) : Integer;
 Function GRMax( const A, B, C : Integer) : Integer;
 Function Clamp( Value, Max : Integer) : Integer;
 Function Clamp1( Value, Min, Max : Integer) : Integer;
 Function Wrap( Value, Max : Integer) : Integer;
 Function Wrap1( Value, Min, Max : Integer) : Integer;
 Function Wrap3( Value, Max : Single) : Single;;
 Function WrapPow2( Value, Max : Integer) : Integer;
 Function WrapPow21( Value, Min, Max : Integer) : Integer;
 Function Mirror( Value, Max : Integer) : Integer;
 Function Mirror1( Value, Min, Max : Integer) : Integer;
 Function MirrorPow2( Value, Max : Integer) : Integer;
 Function MirrorPow21( Value, Min, Max : Integer) : Integer;
 Function GetOptimalWrap( Max : Integer) : TWrapProc;
 Function GetOptimalWrap1( Min, Max : Integer) : TWrapProcEx;
 Function GetOptimalMirror( Max : Integer) : TWrapProc;
 Function GetOptimalMirror1( Min, Max : Integer) : TWrapProcEx;
 Function GetWrapProc( WrapMode : TWrapMode) : TWrapProc;
 Function GetWrapProc1( WrapMode : TWrapMode; Max : Integer) : TWrapProc;
 Function GetWrapProcEx( WrapMode : TWrapMode) : TWrapProcEx;
 Function GetWrapProcEx1( WrapMode : TWrapMode; Min, Max : Integer):TWrapProcEx;
 Function Div255( Value : Cardinal) : Cardinal
 Function SAR_4( Value : Integer) : Integer
 Function SAR_8( Value : Integer) : Integer
 Function SAR_9( Value : Integer) : Integer
 Function SAR_11( Value : Integer) : Integer
 Function SAR_12( Value : Integer) : Integer
 Function SAR_13( Value : Integer) : Integer
 Function SAR_14( Value : Integer) : Integer
 Function SAR_15( Value : Integer) : Integer
 Function SAR_16( Value : Integer) : Integer
 Function ColorSwap( WinColor : TColor) : TColor32
end;

procedure SIRegister_GR32_Filters(CL: TPSPascalCompiler);
begin
 AddTypeS('TLogicalOperator', '( loXOR, loAND, loOR )
 Procedure CopyComponents( Dst, Src : TCustomBitmap32; Components : TColor32Components);
 Procedure CopyComponents1(Dst:TCustomBmap32;DstX,
DstY:Int;Src:TCustomBmap32;SrcRect:TRect;Components:TColor32Comp;
 Procedure AlphaToGrayscale( Dst, Src : TCustomBitmap32)
 Procedure ColorToGrayscale( Dst, Src : TCustomBitmap32; PreserveAlpha : Boolean)
 Procedure IntensityToAlpha( Dst, Src : TCustomBitmap32)
 Procedure Invert( Dst, Src : TCustomBitmap32; Components : TColor32Components)
 Procedure InvertRGB( Dst, Src : TCustomBitmap32)
 Procedure ApplyLUT( Dst, Src : TCustomBitmap32; const LUT : TLUT8; PreserveAlpha : Boolean)
 Procedure ChromaKey( ABitmap : TCustomBitmap32; TrColor : TColor32)
 Function CreateBitmask( Components : TColor32Components) : TColor32
 Procedure ApplyBitmask(Dst: TCustomBitmap32; DstX,DstY:Integer; Src:TCustomBitmap32; SrcRect : TRect;
Bitmask : TColor32; LogicalOperator : TLogicalOperator);
 Procedure
ApplyBitmask1(ABitmap:TCustomBitmap32;ARect:TRect;Bitmask:TColor32;LogicalOperator:TLogicalOperator);
 Procedure CheckParams( Dst, Src : TCustomBitmap32; ResizeDst : Boolean)
end;


procedure SIRegister_JclNTFS(CL: TPSPascalCompiler);
begin
  AddClassN(FindClass('TOBJECT'),'EJclNtfsError
  AddTypeS('TFileCompressionState', '( fcNoCompression, fcDefaultCompression, fcLZNT1Compression )
 Function NtfsGetCompression( const FileName : string; var State : Short) : Boolean;
 Function NtfsGetCompression1( const FileName : string) : TFileCompressionState;
 Function NtfsSetCompression( const FileName : string; const State : Short) : Boolean
 Procedure NtfsSetFileCompression( const FileName : string; const State : TFileCompressionState)
 Procedure NtfsSetDirectoryTreeCompression(const Directory: string; const State : TFileCompressionState)
 Procedure NtfsSetDefaultFileCompression(const Directory: string; const State:TFileCompressionState)
 Procedure NtfsSetPathCompression(const Path:string;const State:TFileCompressionState;Recursive:Boolean;
  //AddTypeS('TNtfsAllocRanges', 'record Entries : Integer; Data : PFileAlloca'
   //+'tedRangeBuffer; MoreData : Boolean; end
 Function NtfsSetSparse( const FileName : string) : Boolean
 Function NtfsZeroDataByHandle( const Handle : THandle; const First, Last : Int64) : Boolean
 Function NtfsZeroDataByName( const FileName : string; const First, Last : Int64) : Boolean
 //Function NtfsQueryAllocRanges(const FileName:string;Offset,Count:Int64;var
Ranges:TNtfsAllocRanges):Boolean;
 //Function NtfsGetAllocRangeEntry( const Ranges : TNtfsAllocRanges;
Index:Integer):TFileAllocatedRangeBuffer
 Function NtfsSparseStreamsSupported( const Volume : string) : Boolean
 Function NtfsGetSparse( const FileName : string) : Boolean
 Function NtfsDeleteReparsePoint( const FileName : string; ReparseTag : DWORD) : Boolean
 Function NtfsSetReparsePoint( const FileName : string; var ReparseData, Size : Longword) : Boolean
 //Function NtfsGetReparsePoint(const FileName:string; var ReparseData:TReparseGuidDataBuffer):Boolean
 Function NtfsGetReparseTag( const Path : string; var Tag : DWORD) : Boolean
 Function NtfsReparsePointsSupported( const Volume : string) : Boolean
 Function NtfsFileHasReparsePoint( const Path : string) : Boolean
 Function NtfsIsFolderMountPoint( const Path : string) : Boolean
 Function NtfsMountDeviceAsDrive( const Device : string; Drive : Char) : Boolean
 Function NtfsMountVolume( const Volume : Char; const MountPoint : string) : Boolean
  AddTypeS('TOpLock', '( olExclusive, olReadOnly, olBatch, olFilter )
 Function NtfsOpLockAckClosePending( Handle : THandle; Overlapped : TOverlapped) : Boolean
 Function NtfsOpLockBreakAckNo2( Handle : THandle; Overlapped : TOverlapped) : Boolean
 Function NtfsOpLockBreakAcknowledge( Handle : THandle; Overlapped : TOverlapped) : Boolean
 Function NtfsOpLockBreakNotify( Handle : THandle; Overlapped : TOverlapped) : Boolean
 Function NtfsRequestOpLock( Handle : THandle; Kind : TOpLock; Overlapped : TOverlapped) : Boolean
 Function NtfsCreateJunctionPoint( const Source, Destination : string) : Boolean
 Function NtfsDeleteJunctionPoint( const Source : string) : Boolean
 Function NtfsGetJunctionPointDestination( const Source : string; var Destination : string) : Boolean
  AddTypeS('TStreamId', '( siInvalid, siStandard, siExtendedAttribute, siSec'
   +'urity, siAlternate, siHardLink, siProperty, siObjectIdentifier, siReparsePoints, siSparseFile )
  AddTypeS('TStreamIds', 'set of TStreamId
  AddTypeS('TInternalFindStreamData', 'record FileHandle : THandle; Context '
   +': ___Pointer; StreamIds : TStreamIds; end
  AddTypeS('TFindStreamData', 'record internal : TInternalFindStreamData; At'
   +'tributes : DWORD; StreamID : TStreamId; Name : WideString; Size : Int64; end
 Function NtfsFindFirstStream(const FileName:string;StreamIds:TStreamIds;var Data:TFindStreamData):Boolean;
 Function NtfsFindNextStream( var Data : TFindStreamData) : Boolean
 Function NtfsFindStreamClose( var Data : TFindStreamData) : Boolean
 Function NtfsCreateHardLink( const LinkFileName, ExistingFileName : string) : Boolean
  AddTypeS('TNtfsHardLinkInfo', 'record LinkCount : Cardinal; FileIndex : Int64; end
 Function NtfsGetHardLinkInfo( const FileName : string; var Info : TNtfsHardLinkInfo) : Boolean
 Function NtfsFindHardLinks(const Path:string;const FileIndexHigh,FileIndexLow:Cardinal;const
List:TStrings):Bool;
 Function NtfsDeleteHardLinks( const FileName : string) : Boolean
 Function JclAppInstances : TJclAppInstances;
 Function JclAppInstances1( const UniqueAppIdGuidStr : string) : TJclAppInstances;
 Function ReadMessageCheck( var Message: TMessage;const IgnoredOriginatorWnd: HWND) : TJclAppInstDataKind
 Procedure ReadMessageData( const Message : TMessage; var Data : ___Pointer; var Size : Integer)
 Procedure ReadMessageString( const Message : TMessage; var S : string)
 Procedure ReadMessageStrings( const Message : TMessage; const Strings : TStrings)


(*----------------------------------------------------------------------------*)
procedure SIRegister_JclGraphics(CL: TPSPascalCompiler);
begin
  FindClass('TOBJECT'),'EJclGraphicsError
  TDynDynIntegerArrayArray', 'array of TDynIntegerArray
  TDynPointArray', 'array of TPoint
  TDynDynPointArrayArray', 'array of TDynPointArray
  TPointF', 'record X : Single; Y : Single; end
  TDynPointArrayF', 'array of TPointF
  TDrawMode2', '( dmOpaque, dmBlend )
  TStretchFilter2', '( sfNearest, sfLinear, sfSpline )
  TConversionKind', '( ckRed, ckGreen, ckBlue, ckAlpha, ckUniformRGB, ckWeightedRGB )
  TResamplingFilter', '( rfBox, rfTriangle, rfHermite, rfBell, rfSpline, rfLanczos3, rfMitchell )
  TMatrix3d', 'record array[0..2,0..2] of extended end
  TDynDynPointArrayArrayF', 'array of TDynPointArrayF
  TScanLine', 'array of Integer
  TScanLines', 'array of TScanLine
  TColorChannel', '( ccRed, ccGreen, ccBlue, ccAlpha )
  TGradientDirection', '( gdVertical, gdHorizontal )
  TPolyFillMode', '( fmAlternate, fmWinding )
  TJclRegionCombineOperator', '( coAnd, coDiff, coOr, coXor )
  TJclRegionBitmapMode', '( rmInclude, rmExclude )
  TJclRegionKind', '( rkNull, rkSimple, rkComplex, rkError )
  SIRegister_TJclDesktopCanvas(CL);
  FindClass('TOBJECT'),'TJclRegion
  SIRegister_TJclRegionInfo(CL);
  SIRegister_TJclRegion(CL);
  SIRegister_TJclThreadPersistent(CL);
  SIRegister_TJclCustomMap(CL);
  SIRegister_TJclBitmap32(CL);
  SIRegister_TJclByteMap(CL);
  SIRegister_TJclTransformation(CL);
  SIRegister_TJclLinearTransformation(CL);
 Procedure Stretch(NewWidth,
NewHeight:Card;Filter:TResamplingFilter;Radius:Single;Source:TGraphic;Target:TBitmap);
 Procedure Stretch1(NewWidth,NewHeight:Cardinal;Filter:TResamplingFilter;Radius:Single;Bitmap:TBitmap);
 Procedure DrawBitmap( DC : HDC; Bitmap : HBitMap; X, Y, Width, Height : Integer)
 Function GetAntialiasedBitmap( const Bitmap : TBitmap) : TBitmap
 Procedure BitmapToJPeg( const FileName : string)
 Procedure JPegToBitmap( const FileName : string)
 Function ExtractIconCount( const FileName : string) : Integer
 Function BitmapToIconJ( Bitmap : HBITMAP; cx, cy : Integer) : HICON
 Function IconToBitmapJ( Icon : HICON) : HBITMAP
 Procedure BlockTransfer(Dst:TJclBitmap32;DstX:Int;DstY:Int;Src:TJclBitmap32;SrcRect:TRect;CombineOp:TDrawMode)
 Procedure StretchTransfer(Dst:TJclBitmap32; DstRect:TRect;Src:TJclBitmap32;SrcRect:TRect;StretchFilter:TStretchFilter; CombineOp : TDrawMode)
 Procedure Transform( Dst, Src : TJclBitmap32; SrcRect : TRect; Transformation : TJclTransformation)
 Procedure SetBorderTransparent( ABitmap : TJclBitmap32; ARect : TRect)
 Function FillGradient(DC:HDC; ARect:TRect; ColorCount:Integer; StartColor,EndColor:TColor;ADirection : TGradientDirection) : Boolean;
 Function CreateRegionFromBitmap(Bitmap: TBitmap; RegionColor:TColor; RegionBitmapMode:TJclRegionBitmapMode): HRGN
 Procedure ScreenShot( bm : TBitmap; Left, Top, Width, Height : Integer; Window : HWND);
 Procedure ScreenShot1( bm : TBitmap);
 Procedure PolyLineTS( Bitmap : TJclBitmap32; const Points : TDynPointArray; Color : TColor32)
 Procedure PolyLineAS( Bitmap : TJclBitmap32; const Points : TDynPointArray; Color : TColor32)
 Procedure PolyLineFS( Bitmap : TJclBitmap32; const Points : TDynPointArrayF; Color : TColor32)
 Procedure PolygonTS( Bitmap : TJclBitmap32; const Points : TDynPointArray; Color : TColor32)
 Procedure PolygonAS( Bitmap : TJclBitmap32; const Points : TDynPointArray; Color : TColor32)
 Procedure PolygonFS( Bitmap : TJclBitmap32; const Points : TDynPointArrayF; Color : TColor32)
 Procedure PolyPolygonTS(Bitmap:TJclBitmap32;const Points:TDynDynPointArrayArray;Color:TColor32);
 Procedure PolyPolygonAS(Bitmap:TJclBitmap32;const Points:TDynDynPointArrayArray;Color:TColor32);
 Procedure PolyPolygonFS(Bitmap:TJclBitmap32;const Points:TDynDynPointArrayArrayF;Color:TColor32);
 Procedure AlphaToGrayscale( Dst, Src : TJclBitmap32)
 Procedure IntensityToAlpha( Dst, Src : TJclBitmap32)
 Procedure Invert( Dst, Src : TJclBitmap32)
 Procedure InvertRGB( Dst, Src : TJclBitmap32)
 Procedure ColorToGrayscale( Dst, Src : TJclBitmap32)
 Procedure ApplyLUT( Dst, Src : TJclBitmap32; const LUT : TLUT8)
 Procedure SetGamma( Gamma : Single)
end;

(*----------------------------------------------------------------------------*)
procedure SIRegister_JclSynch(CL: TPSPascalCompiler);
begin
 Function LockedAdd( var Target : Integer; Value : Integer) : Integer
 Function LockedCompareExchange( var Target : Integer; Exch, Comp : Integer) : Integer;
 Function LockedCompareExchange1( var Target : ___Pointer; Exch, Comp : ___Pointer) : Pointer;
 Function LockedDec( var Target : Integer) : Integer
 Function LockedExchange( var Target : Integer; Value : Integer) : Integer
 Function LockedExchangeAdd( var Target : Integer; Value : Integer) : Integer
 Function LockedExchangeDec( var Target : Integer) : Integer
 Function LockedExchangeInc( var Target : Integer) : Integer
 Function LockedExchangeSub( var Target : Integer; Value : Integer) : Integer
 Function LockedInc( var Target : Integer) : Integer
 Function LockedSub( var Target : Integer; Value : Integer) : Integer
  TJclWaitResult', '( wrAbandoned, wrError, wrIoCompletion, wrSignaled, wrTimeout )
  SIRegister_TJclDispatcherObject(CL);
 Function WaitForMultipleObjects(const Objects:array of TJclDispatcherObject;WaitAll:Bool;TimeOut:Cardinal):Cardinal;
 Function WaitAlertableForMultipleObjects(const Objects : array of TJclDispatcherObject; WaitAll:Bool; TimeOut : Cardinal):Cardinal
  SIRegister_TJclCriticalSection(CL);
  SIRegister_TJclCriticalSectionEx(CL);
  SIRegister_TJclEvent(CL);
  SIRegister_TJclWaitableTimer(CL);
  SIRegister_TJclSemaphore(CL);
  SIRegister_TJclMutex(CL);
  POptexSharedInfo', '^TOptexSharedInfo // will not work
  TOptexSharedInfo', 'record SpinCount:Int; LockCount: Int; ThreadId:Longword; RecursionCount:Int; end
  SIRegister_TJclOptex(CL);
  TMrewPreferred', '( mpReaders, mpWriters, mpEqual )
  TMrewThreadInfo', 'record ThreadId : Longword; RecursionCount: Integer; Reader : Boolean; end
  TMrewThreadInfoArray', 'array of TMrewThreadInfo
  SIRegister_TJclMultiReadExclusiveWrite(CL);
  PMetSectSharedInfo', '^TMetSectSharedInfo // will not work
  TMetSectSharedInfo', 'record Initialized : LongBool; SpinLock : '
   +'Longint; ThreadsWaiting : Longint; AvailableCount : Longint; MaximumCount : Longint; end
  PMeteredSection', '^TMeteredSection // will not work
  TMeteredSection', 'record Event : THandle; FileMap : THandle; SharedInfo : PMetSectSharedInfo; end
  SIRegister_TJclMeteredSection(CL);
  TEventInfo', 'record EventType : Longint; Signaled : LongBool; end
  TMutexInfo', 'record SignalState : Longint; Owned : Boolean; Abandoned : Boolean; end
  TSemaphoreCounts', 'record CurrentCount : Longint; MaximumCount: Longint; end
  TTimerInfo', 'record Remaining : TLargeInteger; Signaled : LongBool; end
 Function QueryCriticalSection( CS : TJclCriticalSection; var Info : TRTLCriticalSection) : Boolean
 Function QueryEvent( Handle : THandle; var Info : TEventInfo) : Boolean
 Function QueryMutex( Handle : THandle; var Info : TMutexInfo) : Boolean
 Function QuerySemaphore( Handle : THandle; var Info : TSemaphoreCounts) : Boolean
 Function QueryTimer( Handle : THandle; var Info : TTimerInfo) : Boolean
  FindClass('TOBJECT'),'EJclWin32HandleObjectError
  FindClass('TOBJECT'),'EJclDispatcherObjectError
  FindClass('TOBJECT'),'EJclCriticalSectionError
  FindClass('TOBJECT'),'EJclEventError
  FindClass('TOBJECT'),'EJclWaitableTimerError
  FindClass('TOBJECT'),'EJclSemaphoreError
  FindClass('TOBJECT'),'EJclMutexError
  FindClass('TOBJECT'),'EJclMeteredSectionError
end;


//***************************unit uPSI_mORMotReport;
Procedure SetCurrentPrinterAsDefault
Function CurrentPrinterName : string
Function mCurrentPrinterPaperSize : string
Procedure UseDefaultPrinter

procedure SIRegisterTSTREAM(Cl: TPSPascalCompiler);
begin
  with FindClass('TOBJECT'), 'TStream') do begin
    IsAbstract := True;
   //RegisterMethod('Function Read( var Buffer, Count : Longint) : Longint
   // RegisterMethod('Function Write( const Buffer, Count : Longint) : Longint
    function Read(Buffer:String;Count:LongInt):LongInt
    function Write(Buffer:String;Count:LongInt):LongInt
    function ReadString(Buffer:String;Count:LongInt):LongInt    //FileStream
    function WriteString(Buffer:String;Count:LongInt):LongInt
    function ReadInt(Buffer:integer;Count:LongInt):LongInt
    function WriteInt(Buffer:integer;Count:LongInt):LongInt
    function ReadByteArray(Buffer:TByteArray;Count:LongInt):LongInt');
    function WriteByteArray(Buffer:TByteArray;Count:LongInt):LongInt');

    procedure ReadAB(Buffer: TByteArray;Count:LongInt)
    procedure WriteAB(Buffer: TByteArray;Count:LongInt)
    procedure ReadABD(Buffer: TByteDynArray;Count:LongInt)
    procedure WriteABD(Buffer: TByteDynArray;Count:LongInt)
    procedure ReadAC(Buffer: TCharArray;Count:LongInt)
    procedure WriteAC(Buffer: TCharArray;Count:LongInt)
    procedure ReadACD(Buffer: TCharDynArray;Count:LongInt)
    procedure WriteACD(Buffer: TCharDynArray;Count:LongInt)

    function Seek(Offset:LongInt;Origin:Word):LongInt
    procedure ReadBuffer(Buffer:String;Count:LongInt)
    procedure WriteBuffer(Buffer:String;Count:LongInt)
    procedure ReadBufferInt(Buffer:Integer;Count:LongInt)');
    procedure WriteBufferInt(Buffer:Integer;Count:LongInt)');
    procedure ReadBufferFloat(Buffer:Double;Count:LongInt)');
    Procedure WriteBufferFloat(Buffer:Double;Count:LongInt)');

    procedure ReadBufferAB(Buffer: TByteArray;Count:LongInt)
    procedure WriteBufferAB(Buffer: TByteArray;Count:LongInt)
    procedure ReadBufferABD(Buffer: TByteDynArray;Count:LongInt)
    procedure WriteBufferABD(Buffer: TByteDynArray;Count:LongInt)
    procedure ReadBufferAW(Buffer: TWordArray;Count:LongInt)
    procedure WriteBufferAW(Buffer: TWordArray;Count:LongInt)
    procedure ReadBufferAWD(Buffer: TWordDynArray;Count:LongInt)
    procedure WriteBufferAWD(Buffer: TWordDynArray;Count:LongInt)
    procedure ReadBufferAW(Buffer: TWordArray;Count:LongInt)
    procedure WriteBufferAW(Buffer: TWordArray;Count:LongInt)
    procedure ReadBufferAC(Buffer: TCharArray;Count:LongInt)
    procedure WriteBufferAC(Buffer: TCharArray;Count:LongInt)
    procedure ReadBufferACD(Buffer: TCharDynArray;Count:LongInt)
    procedure WriteBufferACD(Buffer: TCharDynArray;Count:LongInt)

    procedure ReadBufferP(Buffer: PChar;Count:LongInt)
    procedure WriteBufferP(Buffer: PChar;Count:LongInt)
    procedure ReadBufferO(Buffer: TObject;Count:LongInt)');
    procedure WriteBufferO(Buffer: TObject;Count:LongInt)');
    //READBUFFERAC
    function InstanceSize: Longint
    Procedure FixupResourceHeader( FixupInfo : Integer)
    Procedure ReadResHeader

    {$IFDEF DELPHI4UP}
    function CopyFrom(Source:TStream;Count:Int64):LongInt
    {$ELSE}
    function CopyFrom(Source:TStream;Count:Integer):LongInt
    {$ENDIF}
    RegisterProperty('Position', 'LongInt', iptrw);
    RegisterProperty('Size', 'LongInt', iptrw);
  end;
end;


{ ******************************************************************
  Unit DMATH - Interface for DMATH.DLL
  ****************************************************************** }
// see more docs/dmath_manual.pdf

Function InitEval : Integer
Procedure SetVariable( VarName : Char; Value : Float)
Procedure SetFunction( FuncName : String; Wrapper : TWrapper)
Function Eval( ExpressionString : String) : Float
```

```

unit dmath; //types are in built, others are external in DLL
interface
{$IFDEF DELPHI}
uses
  StdCtrls, Graphics;
{$ENDIF}
{ ------------------------------------------------------------------
  Types and constants
  ------------------------------------------------------------------ }
{$i types.inc}
{ ------------------------------------------------------------------
  Error handling
  ------------------------------------------------------------------ }
procedure SetErrCode(ErrCode : Integer); external 'dmath';
{ Sets the error code }
function DefaultVal(ErrCode : Integer; DefVal : Float) : Float; external 'dmath';
{ Sets error code and default function value }
function MathErr : Integer; external 'dmath';
{ Returns the error code }
{ ------------------------------------------------------------------
  Dynamic arrays
  ------------------------------------------------------------------ }
procedure SetAutoInit(AutoInit : Boolean); external 'dmath';
{ Sets the auto-initialization of arrays }
procedure DimVector(var V : TVector; Ub : Integer); external 'dmath';
{ Creates floating point vector V[0..Ub] }
procedure DimIntVector(var V : TIntVector; Ub : Integer); external 'dmath';
{ Creates integer vector V[0..Ub] }
procedure DimCompVector(var V : TCompVector; Ub : Integer); external 'dmath';
{ Creates complex vector V[0..Ub] }
procedure DimBoolVector(var V : TBoolVector; Ub : Integer); external 'dmath';
{ Creates boolean vector V[0..Ub] }
procedure DimStrVector(var V : TStrVector; Ub : Integer); external 'dmath';
{ Creates string vector V[0..Ub] }
procedure DimMatrix(var A : TMatrix; Ub1, Ub2 : Integer); external 'dmath';
{ Creates floating point matrix A[0..Ub1, 0..Ub2] }
procedure DimIntMatrix(var A : TIntMatrix; Ub1, Ub2 : Integer); external 'dmath';
{ Creates integer matrix A[0..Ub1, 0..Ub2] }
procedure DimCompMatrix(var A : TCompMatrix; Ub1, Ub2 : Integer); external 'dmath';
{ Creates complex matrix A[0..Ub1, 0..Ub2] }
procedure DimBoolMatrix(var A : TBoolMatrix; Ub1, Ub2 : Integer); external 'dmath';
{ Creates boolean matrix A[0..Ub1, 0..Ub2] }
procedure DimStrMatrix(var A : TStrMatrix; Ub1, Ub2 : Integer); external 'dmath';
{ Creates string matrix A[0..Ub1, 0..Ub2] }
{ ------------------------------------------------------------------
  Minimum, maximum, sign and exchange
  ------------------------------------------------------------------ }
function FMin(X, Y : Float) : Float; external 'dmath';
{ Minimum of 2 reals }
function FMax(X, Y : Float) : Float; external 'dmath';
{ Maximum of 2 reals }
function IMin(X, Y : Integer) : Integer; external 'dmath';
{ Minimum of 2 integers }
function IMax(X, Y : Integer) : Integer; external 'dmath';
{ Maximum of 2 integers }
function Sgn(X : Float) : Integer; external 'dmath';
{ Sign (returns 1 if X = 0) }
function Sgn0(X : Float) : Integer; external 'dmath';
{ Sign (returns 0 if X = 0) }
function DSgn(A, B : Float) : Float; external 'dmath';
{ Sgn(B) * |A| }
procedure FSwap(var X, Y : Float); external 'dmath';
{ Exchange 2 reals }
procedure ISwap(var X, Y : Integer); external 'dmath';
{ Exchange 2 integers }
{ ------------------------------------------------------------------
  Rounding functions
  ------------------------------------------------------------------ }
function RoundN(X : Float; N : Integer) : Float; external 'dmath';
{ Rounds X to N decimal places }
function Ceil(X : Float) : Integer; external 'dmath';
{ Ceiling function }
function Floor(X : Float) : Integer; external 'dmath';
{ Floor function }
{ ------------------------------------------------------------------
  Logarithms, exponentials and power
  ------------------------------------------------------------------ }
function Expo(X : Float) : Float; external 'dmath';
{ Exponential }
function Exp2(X : Float) : Float; external 'dmath';
{ 2^X }
function Exp10(X : Float) : Float; external 'dmath';
{ 10^X }
function Log(X : Float) : Float; external 'dmath';
{ Natural log }
function Log2(X : Float) : Float; external 'dmath';
{ Log, base 2 }
function Log10(X : Float) : Float; external 'dmath';
```

```
{ Decimal log }
function LogA(X, A : Float) : Float; external 'dmath';
{ Log, base A }
function IntPower(X : Float; N : Integer) : Float; external 'dmath';
{ X^N }
function Power(X, Y : Float) : Float; external 'dmath';
{ X^Y, X >= 0 }
{ ------------------------------------------------------------------
  Trigonometric functions
  ------------------------------------------------------------------ }
function Pythag(X, Y : Float) : Float; external 'dmath';
{ Sqrt(X^2 + Y^2) }
function FixAngle(Theta : Float) : Float; external 'dmath';
{ Set Theta in -Pi..Pi }
function Tan(X : Float) : Float; external 'dmath';
{ Tangent }
function ArcSin(X : Float) : Float; external 'dmath';
{ Arc sinus }
function ArcCos(X : Float) : Float; external 'dmath';
{ Arc cosinus }
function ArcTan2(Y, X : Float) : Float; external 'dmath';
{ Angle (Ox, OM) with M(X,Y) }
{ ------------------------------------------------------------------
  Hyperbolic functions
  ------------------------------------------------------------------ }
function Sinh(X : Float) : Float; external 'dmath';
{ Hyperbolic sine }
function Cosh(X : Float) : Float; external 'dmath';
{ Hyperbolic cosine }
function Tanh(X : Float) : Float; external 'dmath';
{ Hyperbolic tangent }
function ArcSinh(X : Float) : Float; external 'dmath';
{ Inverse hyperbolic sine }
function ArcCosh(X : Float) : Float; external 'dmath';
{ Inverse hyperbolic cosine }
function ArcTanh(X : Float) : Float; external 'dmath';
{ Inverse hyperbolic tangent }
procedure SinhCosh(X : Float; var SinhX, CoshX : Float); external 'dmath';
{ Sinh & Cosh }
{ ------------------------------------------------------------------
  Gamma function and related functions
  ------------------------------------------------------------------ }
function Fact(N : Integer) : Float; external 'dmath';
{ Factorial }
function SgnGamma(X : Float) : Integer; external 'dmath';
{ Sign of Gamma function }
function Gamma(X : Float) : Float; external 'dmath';
{ Gamma function }
function LnGamma(X : Float) : Float; external 'dmath';
{ Logarithm of Gamma function }
function Stirling(X : Float) : Float; external 'dmath';
{ Stirling's formula for the Gamma function }
function StirLog(X : Float) : Float; external 'dmath';
{ Approximate Ln(Gamma) by Stirling's formula, for X >= 13 }
function DiGamma(X : Float ) : Float; external 'dmath';
{ Digamma function }
function TriGamma(X : Float ) : Float; external 'dmath';
{ Trigamma function }
function IGamma(A, X : Float) : Float; external 'dmath';
{ Incomplete Gamma function}
function JGamma(A, X : Float) : Float; external 'dmath';
{ Complement of incomplete Gamma function }
function InvGamma(A, P : Float) : Float; external 'dmath';
{ Inverse of incomplete Gamma function }
function Erf(X : Float) : Float; external 'dmath';
{ Error function }
function Erfc(X : Float) : Float; external 'dmath';
{ Complement of error function }
{ ------------------------------------------------------------------
  Beta function and related functions
  ------------------------------------------------------------------ }
function Beta(X, Y : Float) : Float; external 'dmath';
{ Beta function }
function IBeta(A, B, X : Float) : Float; external 'dmath';
{ Incomplete Beta function }
function InvBeta(A, B, Y : Float) : Float; external 'dmath';
{ Inverse of incomplete Beta function }
{ ------------------------------------------------------------------
  Lambert's function
  ------------------------------------------------------------------ }
function LambertW(X : Float; UBranch, Offset : Boolean) : Float; external 'dmath';
 ------------------------------------------------------------------
  Binomial distribution
  ------------------------------------------------------------------ }
function Binomial(N, K : Integer) : Float; external 'dmath';
{ Binomial coefficient C(N,K) }
function PBinom(N : Integer; P : Float; K : Integer) : Float; external 'dmath';
{ Probability of binomial distribution }
function FBinom(N : Integer; P : Float; K : Integer) : Float; external 'dmath';
```

```
{ Cumulative probability for binomial distrib. }
{ ------------------------------------------------------------------
  Poisson distribution
  ------------------------------------------------------------------ }
function PPoisson(Mu : Float; K : Integer) : Float; external 'dmath';
{ Probability of Poisson distribution }
function FPoisson(Mu : Float; K : Integer) : Float; external 'dmath';
{ Cumulative probability for Poisson distrib. }
{ ------------------------------------------------------------------
  Exponential distribution
  ------------------------------------------------------------------ }
function DExpo(A, X : Float) : Float; external 'dmath';
{ Density of exponential distribution with parameter A }
function FExpo(A, X : Float) : Float; external 'dmath';
{ Cumulative probability function for exponential dist. with parameter A }
{ ------------------------------------------------------------------
  Standard normal distribution
  ------------------------------------------------------------------ }
function DNorm(X : Float) : Float; external 'dmath';
{ Density of standard normal distribution }
function FNorm(X : Float) : Float; external 'dmath';
{ Cumulative probability for standard normal distrib. }
function PNorm(X : Float) : Float; external 'dmath';
{ Prob(|U| > X) for standard normal distrib. }
function InvNorm(P : Float) : Float; external 'dmath';
{ Inverse of standard normal distribution }
{ ------------------------------------------------------------------
  Student's distribution
  ------------------------------------------------------------------ }
function DStudent(Nu : Integer; X : Float) : Float; external 'dmath';
{ Density of Student distribution with Nu d.o.f. }
function FStudent(Nu : Integer; X : Float) : Float; external 'dmath';
{ Cumulative probability for Student distrib. with Nu d.o.f. }
function PStudent(Nu : Integer; X : Float) : Float; external 'dmath';
{ Prob(|t| > X) for Student distrib. with Nu d.o.f. }
function InvStudent(Nu : Integer; P : Float) : Float; external 'dmath';
{ Inverse of Student's t-distribution function }
{ ------------------------------------------------------------------
  Khi-2 distribution
  ------------------------------------------------------------------ }
function DKhi2(Nu : Integer; X : Float) : Float; external 'dmath';
{ Density of Khi-2 distribution with Nu d.o.f. }
function FKhi2(Nu : Integer; X : Float) : Float; external 'dmath';
{ Cumulative prob. for Khi-2 distrib. with Nu d.o.f. }
function PKhi2(Nu : Integer; X : Float) : Float; external 'dmath';
{ Prob(Khi2 > X) for Khi-2 distrib. with Nu d.o.f. }
function InvKhi2(Nu : Integer; P : Float) : Float; external 'dmath';
{ Inverse of Khi-2 distribution function }
{ ------------------------------------------------------------------
  Fisher-Snedecor distribution
  ------------------------------------------------------------------ }
function DSnedecor(Nu1, Nu2 : Integer; X : Float) : Float; external 'dmath';
{ Density of Fisher-Snedecor distribution with Nu1 and Nu2 d.o.f. }
function FSnedecor(Nu1, Nu2 : Integer; X : Float) : Float; external 'dmath';
{ Cumulative prob. for Fisher-Snedecor distrib. with Nu1 and Nu2 d.o.f. }
function PSnedecor(Nu1, Nu2 : Integer; X : Float) : Float; external 'dmath';
{ Prob(F > X) for Fisher-Snedecor distrib. with Nu1 and Nu2 d.o.f. }
function InvSnedecor(Nu1, Nu2 : Integer; P : Float) : Float; external 'dmath';
{ Inverse of Snedecor's F-distribution function }
{ ------------------------------------------------------------------
  Beta distribution
  ------------------------------------------------------------------ }
function DBeta(A, B, X : Float) : Float; external 'dmath';
{ Density of Beta distribution with parameters A and B }
function FBeta(A, B, X : Float) : Float; external 'dmath';
{ Cumulative probability for Beta distrib. with param. A and B }
{ ------------------------------------------------------------------
  Gamma distribution
  ------------------------------------------------------------------ }
function DGamma(A, B, X : Float) : Float; external 'dmath';
{ Density of Gamma distribution with parameters A and B }
function FGamma(A, B, X : Float) : Float; external 'dmath';
{ Cumulative probability for Gamma distrib. with param. A and B }
{ ------------------------------------------------------------------
  Expression evaluation
  ------------------------------------------------------------------ }
function InitEval : Integer; external 'dmath';
{ Initializes built-in functions and returns their number }
function Eval(ExpressionString : String) : Float; external 'dmath';
{ Evaluates an expression at run-time }
procedure SetVariable(VarName : Char; Value : Float); external 'dmath';
{ Assigns a value to a variable }
procedure SetFunction(FuncName : String; Wrapper : TWrapper); external 'dmath';
{ Adds a function to the parser }
{ ------------------------------------------------------------------
  Matrices and linear equations
  ------------------------------------------------------------------ }
procedure GaussJordan(A            : TMatrix;
                      Lb, Ub1, Ub2 : Integer;
```

```
                       var Det      : Float); external 'dmath';
{ Transforms a matrix according to the Gauss-Jordan method }
procedure LinEq(A       : TMatrix;
                B       : TVector;
                Lb, Ub  : Integer;
                var Det : Float); external 'dmath';
{ Solves a linear system according to the Gauss-Jordan method }
procedure Cholesky(A, L : TMatrix; Lb, Ub : Integer); external 'dmath';
{ Cholesky factorization of a positive definite symmetric matrix }
procedure LU_Decomp(A : TMatrix; Lb, Ub : Integer); external 'dmath';
{ LU decomposition }
procedure LU_Solve(A      : TMatrix;
                   B      : TVector;
                   Lb, Ub : Integer;
                   X      : TVector); external 'dmath';
{ Solution of linear system from LU decomposition }
procedure QR_Decomp(A            : TMatrix;
                    Lb, Ub1, Ub2 : Integer;
                    R            : TMatrix); external 'dmath';
{ QR decomposition }
procedure QR_Solve(Q, R         : TMatrix;
                   B            : TVector;
                   Lb, Ub1, Ub2 : Integer;
                   X            : TVector); external 'dmath';
{ Solution of linear system from QR decomposition }
procedure SV_Decomp(A            : TMatrix;
                    Lb, Ub1, Ub2 : Integer;
                    S            : TVector;
                    V            : TMatrix); external 'dmath';
{ Singular value decomposition }
procedure SV_SetZero(S      : TVector;
                     Lb, Ub : Integer;
                     Tol    : Float); external 'dmath';
{ Set lowest singular values to zero }
procedure SV_Solve(U            : TMatrix;
                   S            : TVector;
                   V            : TMatrix;
                   B            : TVector;
                   Lb, Ub1, Ub2 : Integer;
                   X            : TVector); external 'dmath';
{ Solution of linear system from SVD }
procedure SV_Approx(U            : TMatrix;
                    S            : TVector;
                    V            : TMatrix;
                    Lb, Ub1, Ub2 : Integer;
                    A            : TMatrix); external 'dmath';
{ Matrix approximation from SVD }
procedure EigenVals(A      : TMatrix;
                    Lb, Ub : Integer;
                    Lambda : TCompVector); external 'dmath';
{ Eigenvalues of a general square matrix }
procedure EigenVect(A      : TMatrix;
                    Lb, Ub : Integer;
                    Lambda : TCompVector;
                    V      : TMatrix); external 'dmath';
{ Eigenvalues and eigenvectors of a general square matrix }
procedure EigenSym(A      : TMatrix;
                   Lb, Ub : Integer;
                   Lambda : TVector;
                   V      : TMatrix); external 'dmath';
{ Eigenvalues and eigenvectors of a symmetric matrix (SVD method) }
procedure Jacobi(A               : TMatrix;
                 Lb, Ub, MaxIter : Integer;
                 Tol             : Float;
                 Lambda          : TVector;
                 V               : TMatrix); external 'dmath';
{ Eigenvalues and eigenvectors of a symmetric matrix (Jacobi method) }
{ ------------------------------------------------------------------
  Optimization
  ------------------------------------------------------------------ }
procedure MinBrack(Func                   : TFunc;
                   var A, B, C, Fa, Fb, Fc : Float); external 'dmath';
{ Brackets a minimum of a function }
procedure GoldSearch(Func           : TFunc;
                     A, B           : Float;
                     MaxIter        : Integer;
                     Tol            : Float;
                     var Xmin, Ymin : Float); external 'dmath';
{ Minimization of a function of one variable (golden search) }
procedure LinMin(Func      : TFuncNVar;
                 X, DeltaX : TVector;
                 Lb, Ub    : Integer;
                 var R     : Float;
                 MaxIter   : Integer;
                 Tol       : Float;
                 var F_min : Float); external 'dmath';
{ Minimization of a function of several variables along a line }
procedure Newton(Func      : TFuncNVar;
                 HessGrad  : THessGrad;
                 X         : TVector;
                 Lb, Ub    : Integer;
                 MaxIter   : Integer;
                 Tol       : Float;
                 var F_min : Float;
                 G         : TVector;
                 H_inv     : TMatrix;
                 var Det   : Float); external 'dmath';
{ Minimization of a function of several variables (Newton's method) }
procedure SaveNewton(FileName : string); external 'dmath';
{ Save Newton iterations in a file }
procedure Marquardt(Func      : TFuncNVar;
                    HessGrad  : THessGrad;
                    X         : TVector;
                    Lb, Ub    : Integer;
                    MaxIter   : Integer;
                    Tol       : Float;
                    var F_min : Float;
                    G         : TVector;
                    H_inv     : TMatrix;
                    var Det   : Float); external 'dmath';
{ Minimization of a function of several variables (Marquardt's method) }
procedure SaveMarquardt(FileName : string); external 'dmath';
{ Save Marquardt iterations in a file }
procedure BFGS(Func      : TFuncNVar;
               Gradient  : TGradient;
               X         : TVector;
               Lb, Ub    : Integer;
               MaxIter   : Integer;
               Tol       : Float;
               var F_min : Float;
               G         : TVector;
               H_inv     : TMatrix); external 'dmath';
{ Minimization of a function of several variables (BFGS method) }
procedure SaveBFGS(FileName : string); external 'dmath';
{ Save BFGS iterations in a file }
procedure Simplex(Func      : TFuncNVar;
                  X         : TVector;
                  Lb, Ub    : Integer;
                  MaxIter   : Integer;
                  Tol       : Float;
                  var F_min : Float); external 'dmath';
{ Minimization of a function of several variables (Simplex) }
procedure SaveSimplex(FileName : string); external 'dmath';
{ Save Simplex iterations in a file }
{ ------------------------------------------------------------------
  Nonlinear equations
  ------------------------------------------------------------------ }
procedure RootBrack(Func             : TFunc;
                    var X, Y, FX, FY : Float); external 'dmath';
{ Brackets a root of function Func between X and Y }
procedure Bisect(Func     : TFunc;
                 var X, Y : Float;
                 MaxIter  : Integer;
                 Tol      : Float;
                 var F    : Float); external 'dmath';
{ Bisection method }
procedure Secant(Func     : TFunc;
                 var X, Y : Float;
                 MaxIter  : Integer;
                 Tol      : Float;
                 var F    : Float); external 'dmath';
{ Secant method }
procedure NewtEq(Func, Deriv : TFunc;
                 var X       : Float;
                 MaxIter     : Integer;
                 Tol         : Float;
                 var F       : Float); external 'dmath';
{ Newton-Raphson method for a single nonlinear equation }
procedure NewtEqs(Equations : TEquations;
                  Jacobian  : TJacobian;
                  X, F      : TVector;
                  Lb, Ub    : Integer;
                  MaxIter   : Integer;
                  Tol       : Float); external 'dmath';
{ Newton-Raphson method for a system of nonlinear equations }
procedure Broyden(Equations : TEquations;
                  X, F      : TVector;
                  Lb, Ub    : Integer;
                  MaxIter   : Integer;
                  Tol       : Float); external 'dmath';
{ Broyden's method for a system of nonlinear equations }
{ ------------------------------------------------------------------
  Polynomials and rational fractions
  ------------------------------------------------------------------ }
function Poly(X    : Float;
              Coef : TVector;
              Deg  : Integer) : Float; external 'dmath';
{ Evaluates a polynomial }
```

```
function RFrac(X          : Float;
               Coef       : TVector;
               Deg1, Deg2 : Integer) : Float; external 'dmath';
{ Evaluates a rational fraction }
function RootPol1(A, B  : Float;
                  var X : Float) : Integer; external 'dmath';
{ Solves the linear equation A + B * X = 0 }
function RootPol2(Coef : TVector;
                  Z    : TCompVector) : Integer; external 'dmath';
{ Solves a quadratic equation }
function RootPol3(Coef : TVector;
                  Z    : TCompVector) : Integer; external 'dmath';
{ Solves a cubic equation }
function RootPol4(Coef : TVector;
                  Z    : TCompVector) : Integer; external 'dmath';
{ Solves a quartic equation }
function RootPol(Coef : TVector;
                 Deg  : Integer;
                 Z    : TCompVector) : Integer; external 'dmath';
{ Solves a polynomial equation }
function SetRealRoots(Deg : Integer;
                      Z   : TCompVector;
                      Tol : Float) : Integer; external 'dmath';
{ Set the imaginary part of a root to zero }
procedure SortRoots(Deg : Integer;
                    Z   : TCompVector); external 'dmath';
{ Sorts the roots of a polynomial }
{ ------------------------------------------------------------------
  Numerical integration and differential equations
  ------------------------------------------------------------------ }
function TrapInt(X, Y : TVector; N : Integer) : Float; external 'dmath';
{ Integration by trapezoidal rule }
function GausLeg(Func : TFunc; A, B : Float) : Float; external 'dmath';
{ Integral from A to B }
function GausLeg0(Func : TFunc; B : Float) : Float; external 'dmath';
{ Integral from 0 to B }
function Convol(Func1, Func2 : TFunc; T : Float) : Float; external 'dmath';
{ Convolution product at time T }
procedure ConvTrap(Func1,Func2:TFunc; T,Y:TVector; N:Integer);external 'dmath';
{ Convolution by trapezoidal rule }
procedure RKF45(F                    : TDiffEqs;
                Neqn                 : Integer;
                Y, Yp                : TVector;
                var T                : Float;
                Tout, RelErr, AbsErr : Float;
                var Flag             : Integer); external 'dmath';
{ Integration of a system of differential equations }
{ ------------------------------------------------------------------
  Fast Fourier Transform
  ------------------------------------------------------------------ }
procedure FFT(NumSamples        : Integer;
              InArray, OutArray : TCompVector); external 'dmath';
{ Fast Fourier Transform }
procedure IFFT(NumSamples        : Integer;
               InArray, OutArray : TCompVector); external 'dmath';
{ Inverse Fast Fourier Transform }
procedure FFT_Integer(NumSamples     : Integer;
                      RealIn, ImagIn : TIntVector;
                      OutArray       : TCompVector); external 'dmath';
{ Fast Fourier Transform for integer data }
procedure FFT_Integer_Cleanup; external 'dmath';
{ Clear memory after a call to FFT_Integer }
procedure CalcFrequency(NumSamples,
                        FrequencyIndex : Integer;
                        InArray        : TCompVector;
                        var FFT        : Complex); external 'dmath';
{ Direct computation of Fourier transform }
{ ------------------------------------------------------------------
  Random numbers
  ------------------------------------------------------------------ }
procedure SetRNG(RNG : RNG_Type); external 'dmath';
{ Select generator }
procedure InitGen(Seed : RNG_IntType); external 'dmath';
{ Initialize generator }
function IRanGen : RNG_IntType; external 'dmath';
{ 32-bit random integer in [-2^31 .. 2^31 - 1] }
function IRanGen31 : RNG_IntType; external 'dmath';
{ 31-bit random integer in [0 .. 2^31 - 1] }
function RanGen1 : Float; external 'dmath';
{ 32-bit random real in [0,1] }
function RanGen2 : Float; external 'dmath';
{ 32-bit random real in [0,1) }
function RanGen3 : Float; external 'dmath';
{ 32-bit random real in (0,1) }
function RanGen53 : Float; external 'dmath';
{ 53-bit random real in [0,1) }
procedure InitMWC(Seed : RNG_IntType); external 'dmath';
{ Initializes the 'Multiply with carry' random number generator }
function IRanMWC : RNG_IntType; external 'dmath';
```

```
{ Returns a 32 bit random number in [-2^31 ; 2^31-1] }
procedure InitMT(Seed : RNG_IntType); external 'dmath';
{ Initializes Mersenne Twister generator with a seed }
procedure InitMTbyArray(InitKey   : array of RNG_LongType;
                        KeyLength : Word); external 'dmath';
{ Initialize MT generator with an array InitKey[0..(KeyLength - 1)] }
function IRanMT : RNG_IntType; external 'dmath';
{ Random integer from MT generator }
procedure InitUVAGbyString(KeyPhrase : string); external 'dmath';
{ Initializes the UVAG generator with a string }
procedure InitUVAG(Seed : RNG_IntType); external 'dmath';
{ Initializes the UVAG generator with an integer }
function IRanUVAG : RNG_IntType; external 'dmath';
{ Random integer from UVAG generator }
function RanGaussStd : Float; external 'dmath';
{ Random number from standard normal distribution }
function RanGauss(Mu, Sigma : Float) : Float; external 'dmath';
{ Random number from normal distrib. with mean Mu and S. D. Sigma }
procedure RanMult(M      : TVector; L      : TMatrix;
                  Lb, Ub : Integer;
                  X      : TVector); external 'dmath';
{ Random vector from multinormal distribution (correlated) }
procedure RanMultIndep(M, S   : TVector;
                       Lb, Ub : Integer;
                       X      : TVector); external 'dmath';
{ Random vector from multinormal distribution (uncorrelated) }
procedure InitMHParams(NCycles, MaxSim, SavedSim : Integer); external 'dmath';
{ Initializes Metropolis-Hastings parameters }
procedure GetMHParams(var NCycles, MaxSim,SavedSim:Integer); external 'dmath';
{ Returns Metropolis-Hastings parameters }
procedure Hastings(Func      : TFuncNVar;
                   T         : Float;
                   X         : TVector;
                   V         : TMatrix;
                   Lb, Ub    : Integer;
                   Xmat      : TMatrix;
                   X_min     : TVector;
                   var F_min : Float); external 'dmath';
{ Simulation of a probability density function by Metropolis-Hastings }
procedure InitSAParams(NT, NS, NCycles : Integer; RT : Float); external 'dmath';
{ Initializes Simulated Annealing parameters }
procedure SA_CreateLogFile(FileName : String); external 'dmath';
{ Initializes log file }
procedure SimAnn(Func          : TFuncNVar;
                 X, Xmin, Xmax : TVector;
                 Lb, Ub        : Integer;
                 var F_min     : Float); external 'dmath';
{ Minimization of a function of several var. by simulated annealing }
procedure InitGAParams(NP, NG : Integer; SR, MR, HR : Float); external 'dmath';
{ Initializes Genetic Algorithm parameters }
procedure GA_CreateLogFile(FileName : String); external 'dmath';
{ Initializes log file }
procedure GenAlg(Func          : TFuncNVar;
                 X, Xmin, Xmax : TVector;
                 Lb, Ub        : Integer;
                 var F_min     : Float); external 'dmath';
{ Minimization of a function of several var. by genetic algorithm }
{ ------------------------------------------------------------------
  Statistics
  ------------------------------------------------------------------ }
function Mean(X : TVector; Lb, Ub : Integer) : Float; external 'dmath';
{ Mean of sample X }
function Min(X : TVector; Lb, Ub : Integer) : Float; external 'dmath';
{ Minimum of sample X }
function Max(X : TVector; Lb, Ub : Integer) : Float; external 'dmath';
{ Maximum of sample X }
function Median(X : TVector; Lb, Ub : Integer; Sorted : Boolean) : Float; external 'dmath';
{ Median of sample X }
function StDev(X : TVector; Lb, Ub : Integer; M : Float) : Float; external 'dmath';
{ Standard deviation estimated from sample X }
function StDevP(X : TVector; Lb, Ub : Integer; M : Float) : Float; external 'dmath';
{ Standard deviation of population }
function Correl(X, Y : TVector; Lb, Ub : Integer) : Float; external 'dmath';
{ Correlation coefficient }
function Skewness(X : TVector; Lb, Ub : Integer; M,Sigma: Float): Float; external 'dmath';
{ Skewness of sample X }
function Kurtosis(X : TVector; Lb, Ub : Integer; M,Sigma: Float): Float; external 'dmath';
{ Kurtosis of sample X }
procedure QSort(X : TVector; Lb, Ub : Integer); external 'dmath';
{ Quick sort (ascending order) }
procedure DQSort(X : TVector; Lb, Ub : Integer); external 'dmath';
{ Quick sort (descending order) }
procedure Interval(X1, X2             : Float;
                   MinDiv, MaxDiv     : Integer;
                   var Min, Max, Step : Float); external 'dmath';
{ Determines an interval for a set of values }
procedure AutoScale(X : TVector; Lb, Ub : Integer; Scale : TScale;
                    var XMin, XMax, XStep : Float); external 'dmath';
{ Finds an appropriate scale for plotting the data in X[Lb..Ub] }
```

```
procedure StudIndep(N1, N2         : Integer;
                    M1, M2, S1, S2 : Float;
                    var T          : Float;
                    var DoF        : Integer); external 'dmath';
{ Student t-test for independent samples }
procedure StudPaired(X, Y    : TVector;
                     Lb, Ub  : Integer;
                     var T   : Float;
                     var DoF : Integer); external 'dmath';
{ Student t-test for paired samples }
procedure AnOVa1(Ns               : Integer;
                 N                : TIntVector;
                 M, S             : TVector;
                 var V_f, V_r, F  : Float;
                 var DoF_f, DoF_r : Integer); external 'dmath';
{ One-way analysis of variance }
procedure AnOVa2(NA, NB, Nobs : Integer;
                 M, S         : TMatrix;
                 V, F         : TVector;
                 DoF          : TIntVector); external 'dmath';
{ Two-way analysis of variance }
procedure Snedecor(N1, N2         : Integer;
                   S1, S2         : Float;
                   var F          : Float;
                   var DoF1, DoF2 : Integer); external 'dmath';
{ Snedecor's F-test (comparison of two variances) }
procedure Bartlett(Ns       : Integer;
                   N        : TIntVector;
                   S        : TVector;
                   var Khi2 : Float;
                   var DoF  : Integer); external 'dmath';
{ Bartlett's test (comparison of several variances) }
procedure Mann_Whitney(N1, N2     : Integer;
                       X1, X2     : TVector;
                       var U, Eps : Float); external 'dmath';
{ Mann-Whitney test}
procedure Wilcoxon(X, Y       : TVector;
                   Lb, Ub     : Integer;
                   var Ndiff  : Integer;
                   var T, Eps : Float); external 'dmath';
{ Wilcoxon test }
procedure Kruskal_Wallis(Ns      : Integer;
                         N       : TIntVector;
                         X       : TMatrix;
                         var H   : Float;
                         var DoF : Integer); external 'dmath';
{ Kruskal-Wallis test }
procedure Khi2_Conform(N_cls    : Integer;
                       N_estim  : Integer;
                       Obs      : TIntVector;
                       Calc     : TVector;
                       var Khi2 : Float;
                       var DoF  : Integer); external 'dmath';
{ Khi-2 test for conformity }
procedure Khi2_Indep(N_lin    : Integer;
                     N_col    : Integer;
                     Obs      : TIntMatrix;
                     var Khi2 : Float;
                     var DoF  : Integer); external 'dmath';
{ Khi-2 test for independence }
procedure Woolf_Conform(N_cls   : Integer;
                        N_estim : Integer;
                        Obs     : TIntVector;
                        Calc    : TVector;
                        var G   : Float;
                        var DoF : Integer); external 'dmath';
{ Woolf's test for conformity }
procedure Woolf_Indep(N_lin   : Integer;
                      N_col   : Integer;
                      Obs     : TIntMatrix;
                      var G   : Float;
                      var DoF : Integer); external 'dmath';
{ Woolf's test for independence }
procedure DimStatClassVector(var C : TStatClassVector;
                             Ub    : Integer); external 'dmath';
{ Allocates an array of statistical classes: C[0..Ub] }
procedure Distrib(X       : TVector;
                  Lb, Ub  : Integer;
                  A, B, H : Float;
                  C       : TStatClassVector); external 'dmath';
{ Distributes an array X[Lb..Ub] into statistical classes }
{ ------------------------------------------------------------------
  Linear / polynomial regression
  ------------------------------------------------------------------ }
procedure LinFit(X, Y   : TVector;
                 Lb, Ub : Integer;
                 B      : TVector;
                 V      : TMatrix); external 'dmath';
{ Linear regression : Y = B(0) + B(1) * X }
```

```
procedure WLinFit(X, Y, S : TVector;
                  Lb, Ub  : Integer;
                  B       : TVector;
                  V       : TMatrix); external 'dmath';
{ Weighted linear regression : Y = B(0) + B(1) * X }
procedure SVDLinFit(X, Y   : TVector;
                    Lb, Ub : Integer;
                    SVDTol : Float;
                    B      : TVector;
                    V      : TMatrix); external 'dmath';
{ Unweighted linear regression by singular value decomposition }
procedure WSVDLinFit(X, Y, S : TVector;
                     Lb, Ub  : Integer;
                     SVDTol  : Float;
                     B       : TVector;
                     V       : TMatrix); external 'dmath';
{ Weighted linear regression by singular value decomposition }
procedure MulFit(X            : TMatrix;
                 Y            : TVector;
                 Lb, Ub, Nvar : Integer;
                 ConsTerm     : Boolean;
                 B            : TVector;
                 V            : TMatrix); external 'dmath';
{ Multiple linear regression by Gauss-Jordan method }
procedure WMulFit(X            : TMatrix;
                  Y, S         : TVector;
                  Lb, Ub, Nvar : Integer;
                  ConsTerm     : Boolean;
                  B            : TVector;
                  V            : TMatrix); external 'dmath';
{ Weighted multiple linear regression by Gauss-Jordan method }
procedure SVDFit(X            : TMatrix;
                 Y            : TVector;
                 Lb, Ub, Nvar : Integer;
                 ConsTerm     : Boolean;
                 SVDTol       : Float;
                 B            : TVector;
                 V            : TMatrix); external 'dmath';
{ Multiple linear regression by singular value decomposition }
procedure WSVDFit(X            : TMatrix;
                  Y, S         : TVector;
                  Lb, Ub, Nvar : Integer;
                  ConsTerm     : Boolean;
                  SVDTol       : Float;
                  B            : TVector;
                  V            : TMatrix); external 'dmath';
{ Weighted multiple linear regression by singular value decomposition }
procedure PolFit(X, Y        : TVector;
                 Lb, Ub, Deg : Integer;
                 B           : TVector;
                 V           : TMatrix); external 'dmath';
{ Polynomial regression by Gauss-Jordan method }
procedure WPolFit(X, Y, S     : TVector;
                  Lb, Ub, Deg : Integer;
                  B           : TVector;
                  V           : TMatrix); external 'dmath';
{ Weighted polynomial regression by Gauss-Jordan method }
procedure SVDPolFit(X, Y        : TVector;
                    Lb, Ub, Deg : Integer;
                    SVDTol      : Float;
                    B           : TVector;
                    V           : TMatrix); external 'dmath';
{ Unweighted polynomial regression by singular value decomposition }
procedure WSVDPolFit(X, Y, S     : TVector;
                     Lb, Ub, Deg : Integer;
                     SVDTol      : Float;
                     B           : TVector;
                     V           : TMatrix); external 'dmath';
{ Weighted polynomial regression by singular value decomposition }
procedure RegTest(Y, Ycalc : TVector;
                  LbY, UbY : Integer;
                  V        : TMatrix;
                  LbV, UbV : Integer;
                  var Test : TRegTest); external 'dmath';
{ Test of unweighted regression }
procedure WRegTest(Y, Ycalc, S : TVector;
                   LbY, UbY    : Integer;
                   V           : TMatrix;
                   LbV, UbV    : Integer;
                   var Test    : TRegTest); external 'dmath';
{ Test of weighted regression }
{ ------------------------------------------------------------------
  Nonlinear regression
  ------------------------------------------------------------------ }
procedure SetOptAlgo(Algo : TOptAlgo); external 'dmath';
{ Sets the optimization algorithm for nonlinear regression }
function GetOptAlgo : TOptAlgo; external 'dmath';
{ Returns the optimization algorithm }
procedure SetMaxParam(N : Byte); external 'dmath';
```

```
{ Sets the maximum number of regression parameters for nonlinear regression }
function GetMaxParam : Byte; external 'dmath';
{ Returns the maximum number of regression parameters for nonlinear regression }
procedure SetParamBounds(I : Byte; ParamMin, ParamMax : Float); external 'dmath';
{ Sets the bounds on the I-th regression parameter }
procedure GetParamBounds(I : Byte; var ParamMin,ParamMax:Float); external 'dmath';
{ Returns the bounds on the I-th regression parameter }
procedure NLFit(RegFunc   : TRegFunc;
                DerivProc : TDerivProc;
                X, Y      : TVector;
                Lb, Ub    : Integer;
                MaxIter   : Integer;
                Tol       : Float;
                B         : TVector;
                FirstPar,
                LastPar   : Integer;
                V         : TMatrix); external 'dmath';
{ Unweighted nonlinear regression }
procedure WNLFit(RegFunc   : TRegFunc;
                 DerivProc : TDerivProc;
                 X, Y, S   : TVector;
                 Lb, Ub    : Integer;
                 MaxIter   : Integer;
                 Tol       : Float;
                 B         : TVector;
                 FirstPar,
                 LastPar   : Integer;
                 V         : TMatrix); external 'dmath';
{ Weighted nonlinear regression }
procedure SetMCFile(FileName : String); external 'dmath';
{ Set file for saving MCMC simulations }
procedure SimFit(RegFunc   : TRegFunc;
                 X, Y      : TVector;
                 Lb, Ub    : Integer;
                 B         : TVector;
                 FirstPar,
                 LastPar   : Integer;
                 V         : TMatrix); external 'dmath';
{ Simulation of unweighted nonlinear regression by MCMC }
procedure WSimFit(RegFunc   : TRegFunc;
                  X, Y, S   : TVector;
                  Lb, Ub    : Integer;
                  B         : TVector;
                  FirstPar,
                  LastPar   : Integer;
                  V         : TMatrix); external 'dmath';
{ Simulation of weighted nonlinear regression by MCMC }
{ ------------------------------------------------------------------
  Nonlinear regression models
  ------------------------------------------------------------------ }
procedure FracFit(X, Y       : TVector;
                  Lb, Ub     : Integer;
                  Deg1, Deg2 : Integer;
                  ConsTerm   : Boolean;
                  MaxIter    : Integer;
                  Tol        : Float;
                  B          : TVector;
                  V          : TMatrix); external 'dmath';
{ Unweighted fit of rational fraction }
procedure WFracFit(X, Y, S    : TVector;
                   Lb, Ub     : Integer;
                   Deg1, Deg2 : Integer;
                   ConsTerm   : Boolean;
                   MaxIter    : Integer;
                   Tol        : Float;
                   B          : TVector;
                   V          : TMatrix); external 'dmath';
{ Weighted fit of rational fraction }

function FracFit_Func(X : Float; B : TVector) : Float; external 'dmath';
{ Returns the value of the rational fraction at point X }
procedure ExpFit(X, Y         : TVector;
                 Lb, Ub, Nexp : Integer;
                 ConsTerm     : Boolean;
                 MaxIter      : Integer;
                 Tol          : Float;
                 B            : TVector;
                 V            : TMatrix); external 'dmath';
{ Unweighted fit of sum of exponentials }
procedure WExpFit(X, Y, S      : TVector;
                  Lb, Ub, Nexp : Integer;
                  ConsTerm     : Boolean;
                  MaxIter      : Integer;
                  Tol          : Float;
                  B            : TVector;
                  V            : TMatrix); external 'dmath';
{ Weighted fit of sum of exponentials }
function ExpFit_Func(X : Float; B : TVector) : Float; external 'dmath';
{ Returns the value of the regression function at point X }
```

```
procedure IncExpFit(X, Y     : TVector;
                    Lb, Ub   : Integer;
                    ConsTerm : Boolean;
                    MaxIter  : Integer;
                    Tol      : Float;
                    B        : TVector;
                    V        : TMatrix); external 'dmath';
{ Unweighted fit of model of increasing exponential }
procedure WIncExpFit(X, Y, S  : TVector;
                     Lb, Ub   : Integer;
                     ConsTerm : Boolean;
                     MaxIter  : Integer;
                     Tol      : Float;
                     B        : TVector;
                     V        : TMatrix); external 'dmath';
{ Weighted fit of increasing exponential }
function IncExpFit_Func(X : Float; B : TVector) : Float; external 'dmath';
{ Returns the value of the regression function at point X }
procedure ExpLinFit(X, Y    : TVector;
                    Lb, Ub  : Integer;
                    MaxIter : Integer;
                    Tol     : Float;
                    B       : TVector;
                    V       : TMatrix); external 'dmath';
{ Unweighted fit of the "exponential + linear" model }
procedure WExpLinFit(X, Y, S : TVector;
                     Lb, Ub  : Integer;
                     MaxIter : Integer;
                     Tol     : Float;
                     B       : TVector;
                     V       : TMatrix); external 'dmath';
{ Weighted fit of the "exponential + linear" model }

function ExpLinFit_Func(X : Float; B : TVector) : Float; external 'dmath';
{ Returns the value of the regression function at point X }
procedure MichFit(X, Y    : TVector;
                  Lb, Ub  : Integer;
                  MaxIter : Integer;
                  Tol     : Float;
                  B       : TVector;
                  V       : TMatrix); external 'dmath';
{ Unweighted fit of Michaelis equation }
procedure WMichFit(X, Y, S : TVector;
                   Lb, Ub  : Integer;
                   MaxIter : Integer;
                   Tol     : Float;
                   B       : TVector;
                   V       : TMatrix); external 'dmath';
{ Weighted fit of Michaelis equation }
function MichFit_Func(X : Float; B : TVector) : Float; external 'dmath';
{ Returns the value of the Michaelis equation at point X }
procedure MintFit(X, Y    : TVector;
                  Lb, Ub  : Integer;
                  MintVar : TMintVar;
                  Fit_S0  : Boolean;
                  MaxIter : Integer;
                  Tol     : Float;
                  B       : TVector;
                  V       : TMatrix); external 'dmath';
{ Unweighted fit of the integrated Michaelis equation }
procedure WMintFit(X, Y, S : TVector;
                   Lb, Ub  : Integer;
                   MintVar : TMintVar;
                   Fit_S0  : Boolean;
                   MaxIter : Integer;
                   Tol     : Float;
                   B       : TVector;
                   V       : TMatrix); external 'dmath';
{ Weighted fit of the integrated Michaelis equation }
function MintFit_Func(X : Float; B : TVector) : Float; external 'dmath';
{ Returns the value of the integrated Michaelis equation at point X }
procedure HillFit(X, Y    : TVector;
                  Lb, Ub  : Integer;
                  MaxIter : Integer;
                  Tol     : Float;
                  B       : TVector;
                  V       : TMatrix); external 'dmath';
{ Unweighted fit of Hill equation }
procedure WHillFit(X, Y, S : TVector;
                   Lb, Ub  : Integer;
                   MaxIter : Integer;
                   Tol     : Float;
                   B       : TVector;
                   V       : TMatrix); external 'dmath';
{ Weighted fit of Hill equation }
function HillFit_Func(X : Float; B : TVector) : Float; external 'dmath';
{ Returns the value of the Hill equation at point X }
procedure LogiFit(X, Y     : TVector;
                  Lb, Ub   : Integer;
                  ConsTerm : Boolean;
                  General  : Boolean;
                  MaxIter  : Integer;
                  Tol      : Float;
                  B        : TVector;
                  V        : TMatrix); external 'dmath';
{ Unweighted fit of logistic function }
procedure WLogiFit(X, Y, S  : TVector;
                   Lb, Ub   : Integer;
                   ConsTerm : Boolean;
                   General  : Boolean;
                   MaxIter  : Integer;
                   Tol      : Float;
                   B        : TVector;
                   V        : TMatrix); external 'dmath';
{ Weighted fit of logistic function }
function LogiFit_Func(X : Float; B : TVector) : Float; external 'dmath';
{ Returns the value of the logistic function at point X }
procedure PKFit(X, Y    : TVector;
                Lb, Ub  : Integer;
                MaxIter : Integer;
                Tol     : Float;
                B       : TVector;
                V       : TMatrix); external 'dmath';
{ Unweighted fit of the acid-base titration curve }
procedure WPKFit(X, Y, S : TVector;
                 Lb, Ub  : Integer;
                 MaxIter : Integer;
                 Tol     : Float;
                 B       : TVector;
                 V       : TMatrix); external 'dmath';
{ Weighted fit of the acid-base titration curve }
function PKFit_Func(X : Float; B : TVector) : Float; external 'dmath';
{ Returns the value of the acid-base titration function at point X }
procedure PowFit(X, Y    : TVector;
                 Lb, Ub  : Integer;
                 MaxIter : Integer;
                 Tol     : Float;
                 B       : TVector;
                 V       : TMatrix); external 'dmath';
{ Unweighted fit of power function }
procedure WPowFit(X, Y, S : TVector;
                  Lb, Ub  : Integer;
                  MaxIter : Integer;
                  Tol     : Float;
                  B       : TVector;
                  V       : TMatrix); external 'dmath';
{ Weighted fit of power function }

function PowFit_Func(X : Float; B : TVector) : Float; external 'dmath';
{ Returns the value of the power function at point X }
procedure GammaFit(X, Y    : TVector;
                   Lb, Ub  : Integer;
                   MaxIter : Integer;
                   Tol     : Float;
                   B       : TVector;
                   V       : TMatrix); external 'dmath';
{ Unweighted fit of gamma distribution function }
procedure WGammaFit(X, Y, S : TVector;
                    Lb, Ub  : Integer;
                    MaxIter : Integer;
                    Tol     : Float;
                    B       : TVector;
                    V       : TMatrix); external 'dmath';
{ Weighted fit of gamma distribution function }
function GammaFit_Func(X : Float; B : TVector) : Float; external 'dmath';
{ Returns the value of the gamma distribution function at point X }
{ ------------------------------------------------------------------
  Principal component analysis
  ------------------------------------------------------------------ }
procedure VecMean(X            : TMatrix;
                  Lb, Ub, Nvar : Integer;
                  M            : TVector); external 'dmath';
{ Computes the mean vector M from matrix X }
procedure VecSD(X            : TMatrix;
                Lb, Ub, Nvar : Integer;
                M, S         : TVector); external 'dmath';
{ Computes the vector of standard deviations S from matrix X }
procedure MatVarCov(X            : TMatrix;
                    Lb, Ub, Nvar : Integer;
                    M            : TVector;
                    V            : TMatrix); external 'dmath';
{ Computes the variance-covariance matrix V from matrix X }
procedure MatCorrel(V    : TMatrix;
                    Nvar : Integer;
                    R    : TMatrix); external 'dmath';
{ Computes the correlation matrix R from the var-cov matrix V }
procedure PCA(R      : TMatrix;
              Nvar   : Integer;
              Lambda : TVector;
              C, Rc  : TMatrix); external 'dmath';
{ Performs a principal component analysis of the correlation matrix R }
procedure ScaleVar(X            : TMatrix;
                   Lb, Ub, Nvar : Integer;
                   M, S         : TVector;
                   Z            : TMatrix); external 'dmath';
{ Scales a set of variables by subtracting means and dividing by SD's }
procedure PrinFac(Z            : TMatrix;
                  Lb, Ub, Nvar : Integer;
                  C, F         : TMatrix); external 'dmath';
{ Computes principal factors }
{ -------------------------------------------------------------------
  Strings
  ------------------------------------------------------------------- }
function LTrim(S : String) : String; external 'dmath';
{ Removes leading blanks }
function RTrim(S : String) : String; external 'dmath';
{ Removes trailing blanks }
function Trim(S : String) : String; external 'dmath';
{ Removes leading and trailing blanks }
function StrChar(N : Byte; C : Char) : String; external 'dmath';
{ Returns a string made of character C repeated N times }
function RFill(S : String; L : Byte) : String; external 'dmath';
{ Completes string S with trailing blanks for a total length L }
function LFill(S : String; L : Byte) : String; external 'dmath';
{ Completes string S with leading blanks for a total length L }
function CFill(S : String; L : Byte) : String; external 'dmath';
{ Centers string S on a total length L }
function Replace(S : String; C1, C2 : Char) : String; external 'dmath';
{ Replaces in string S all the occurences of C1 by C2 }
function Extract(S : String; var Index : Byte; Delim : Char) : String; external 'dmath';
{ Extracts a field from a string }
procedure Parse(S : String; Delim:Char; Field:TStrVector; var N:Byte); external 'dmath';
{ Parses a string into its constitutive fields }
procedure SetFormat(NumLength,MaxDec:Integer;FloatPoint,NSZero:Bool); external 'dmath';
{ Sets the numeric format }
function FloatStr(X : Float) : String; external 'dmath';
{ Converts a real to a string according to the numeric format }
function IntStr(N : LongInt) : String; external 'dmath';
{ Converts an integer to a string }
function CompStr(Z : Complex) : String; external 'dmath';
{ Converts a complex number to a string }
{$IFDEF DELPHI}
function StrDec(S : String) : String; external 'dmath';
{ Set decimal separator to the symbol defined in SysUtils }
function IsNumeric(var S : String; var X : Float) : Boolean; external 'dmath';
{ Test if a string represents a number and returns it in X }
function ReadNumFromEdit(Edit : TEdit) : Float; external 'dmath';
{ Reads a floating point number from an Edit control }
procedure WriteNumToFile(var F : Text; X : Float); external 'dmath';
{ Writes a floating point number in a text file }
{$ENDIF}
{ -------------------------------------------------------------------
  BGI / Delphi graphics
  ------------------------------------------------------------------- }
function InitGraphics
{$IFDEF DELPHI}
(Width, Height : Integer) : Boolean;
{$ELSE}
(Pilot, Mode : Integer; BGIPath : String) : Boolean; {$ENDIF} external 'dmath';
{ Enters graphic mode }
procedure SetWindow({$IFDEF DELPHI}Canvas : TCanvas;{$ENDIF}
                    X1, X2, Y1,Y2 : Integer; GraphBorder:Boolean); external 'dmath';
{ Sets the graphic window }
procedure SetOxScale(Scale                : TScale;
                     OxMin, OxMax, OxStep : Float); external 'dmath';
{ Sets the scale on the Ox axis }
procedure SetOyScale(Scale                : TScale;
                     OyMin, OyMax, OyStep : Float); external 'dmath';
{ Sets the scale on the Oy axis }
procedure GetOxScale(var Scale                : TScale;
                     var OxMin, OxMax, OxStep : Float); external 'dmath';
{ Returns the scale on the Ox axis }
procedure GetOyScale(var Scale                : TScale;
                     var OyMin, OyMax, OyStep : Float); external 'dmath';
{ Returns the scale on the Oy axis }
procedure SetGraphTitle(Title : String); external 'dmath'; { Sets the title for the graph }
procedure SetOxTitle(Title : String); external 'dmath'; { Sets the title for the Ox axis }
procedure SetOyTitle(Title : String); external 'dmath'; { Sets the title for the Oy axis }
function GetGraphTitle : String; external 'dmath'; { Returns the title for the graph }
function GetOxTitle : String; external 'dmath'; { Returns the title for the Ox axis }
function GetOyTitle : String; external 'dmath'; { Returns the title for the Oy axis }
{$IFNDEF DELPHI}
procedure SetTitleFont(FontIndex, Width, Height : Integer); external 'dmath';
{ Sets the font for the main graph title }
procedure SetOxFont(FontIndex, Width, Height : Integer); external 'dmath';
{ Sets the font for the Ox axis (title and labels) }
procedure SetOyFont(FontIndex, Width, Height : Integer); external 'dmath';
```

```
{ Sets the font for the Oy axis (title and labels) }
procedure SetLgdFont(FontIndex, Width, Height : Integer); external 'dmath';
{ Sets the font for the legends }
procedure SetClipping(Clip : Boolean); external 'dmath';
{ Determines whether drawings are clipped at the current viewport
  boundaries, according to the value of the Boolean parameter Clip }
{$ENDIF}
procedure PlotOxAxis{$IFDEF DELPHI}(Canvas : TCanvas){$ENDIF}; external 'dmath';
{ Plots the horizontal axis }
procedure PlotOyAxis{$IFDEF DELPHI}(Canvas : TCanvas){$ENDIF}; external 'dmath';
{ Plots the vertical axis }
procedure PlotGrid({$IFDEF DELPHI}Canvas:TCanvas;{$ENDIF} Grid:TGrid); external 'dmath';
{ Plots a grid on the graph }
procedure WriteGraphTitle{$IFDEF DELPHI}(Canvas : TCanvas){$ENDIF}; external 'dmath';
{ Writes the title of the graph }
procedure SetMaxCurv(NCurv : Byte); external 'dmath';
{ Sets the maximum number of curves and re-initializes their parameters }
procedure SetPointParam
{$IFDEF DELPHI}
(CurvIndex, Symbol, Size : Integer; Color : TColor);
{$ELSE}
(CurvIndex, Symbol, Size, Color : Integer); {$ENDIF} external 'dmath';
{ Sets the point parameters for curve # CurvIndex }
procedure SetLineParam
{$IFDEF DELPHI}
(CurvIndex : Integer; Style : TPenStyle; Width : Integer; Color : TColor);
{$ELSE}
(CurvIndex, Style, Width, Color : Integer); {$ENDIF} external 'dmath';
{ Sets the line parameters for curve # CurvIndex }
procedure SetCurvLegend(CurvIndex : Integer; Legend : String); external 'dmath';
{ Sets the legend for curve # CurvIndex }
procedure SetCurvStep(CurvIndex, Step : Integer); external 'dmath';
{ Sets the step for curve # CurvIndex }
function GetMaxCurv : Byte; external 'dmath'; { Returns the maximum number of curves }
procedure GetPointParam
{$IFDEF DELPHI}
(CurvIndex : Integer; var Symbol, Size : Integer; var Color : TColor);
{$ELSE}
(CurvIndex : Integer; var Symbol, Size, Color : Integer); {$ENDIF} external 'dmath';
{ Returns the point parameters for curve # CurvIndex }
procedure GetLineParam
{$IFDEF DELPHI}
(CurvIndex : Integer; var Style : TPenStyle; var Width : Integer; var Color : TColor);
{$ELSE}
(CurvIndex : Integer; var Style, Width, Color : Integer); {$ENDIF} external 'dmath';
{ Returns the line parameters for curve # CurvIndex }
function GetCurvLegend(CurvIndex : Integer) : String; external 'dmath';
{ Returns the legend for curve # CurvIndex }
function GetCurvStep(CurvIndex : Integer) : Integer; external 'dmath';
{ Returns the step for curve # CurvIndex }
{$IFDEF DELPHI}
procedure PlotPoint(Canvas    : TCanvas;
                    X, Y      : Float; CurvIndex : Integer); external 'dmath';
{$ELSE}
procedure PlotPoint(Xp, Yp, CurvIndex : Integer); external 'dmath';
{$ENDIF}
{ Plots a point on the screen }
procedure PlotCurve({$IFDEF DELPHI}Canvas : TCanvas;{$ENDIF}
                    X, Y                 : TVector;
                   Lb, Ub, CurvIndex     : Integer); external 'dmath';
{ Plots a curve }
procedure PlotCurveWithErrorBars({$IFDEF DELPHI}Canvas : TCanvas;{$ENDIF}
                                 X, Y, S              : TVector;
                                 Ns, Lb, Ub, CurvIndex : Integer); external 'dmath';
{ Plots a curve with error bars }
procedure PlotFunc({$IFDEF DELPHI}Canvas : TCanvas;{$ENDIF}
                   Func                 : TFunc;
                   Xmin, Xmax           : Float;
                   {$IFDEF DELPHI}Npt    : Integer;{$ENDIF}
                   CurvIndex            : Integer); external 'dmath';
{ Plots a function }
procedure WriteLegend({$IFDEF DELPHI}Canvas : TCanvas;{$ENDIF}
                      NCurv                 : Integer;
                      ShowPoints, ShowLines : Boolean); external 'dmath';
{ Writes the legends for the plotted curves }
procedure ConRec({$IFDEF DELPHI}Canvas    : TCanvas;{$ENDIF}
                 Nx, Ny, Nc             : Integer;
                 X, Y, Z                : TVector;
                 F                      : TMatrix); external 'dmath';
{ Contour plot }
function Xpixel(X : Float):Integer; external 'dmath'; {Converts user abscissa X to screen coordinate }
function Ypixel(Y : Float):Integer; external 'dmath'; {Converts user ordinate Y to screen coordinate }
function Xuser(X : Integer):Float; external 'dmath';  {Converts screen coordinate X to user abscissa }
function Yuser(Y : Integer):Float; external 'dmath';  {Converts screen coordinate Y to user ordinate }
{$IFNDEF DELPHI}
procedure LeaveGraphics; external 'dmath';
{ Quits graphic mode }
{$ENDIF}
{ ------------------------------------------------------------------
```

```
  LaTeX graphics
  ------------------------------------------------------------------ }
function TeX_InitGraphics(FileName          : String; PgWidth, PgHeight : Integer;
                           Header            : Boolean) : Boolean; external 'dmath';
{ Initializes the LaTeX file }
procedure TeX_SetWindow(X1, X2, Y1, Y2 : Integer; GraphBorder : Boolean); external 'dmath';
{ Sets the graphic window }
procedure TeX_LeaveGraphics(Footer : Boolean); external 'dmath'; { Close the LaTeX file }
procedure TeX_SetOxScale(Scale : TScale; OxMin, OxMax, OxStep : Float); external 'dmath';
{ Sets the scale on the Ox axis }
procedure TeX_SetOyScale(Scale : TScale; OyMin, OyMax, OyStep : Float); external 'dmath';
{ Sets the scale on the Oy axis }
procedure TeX_SetGraphTitle(Title : String); external 'dmath'; { Sets the title for the graph }
procedure TeX_SetOxTitle(Title : String); external 'dmath'; { Sets the title for the Ox axis }
procedure TeX_SetOyTitle(Title : String); external 'dmath'; { Sets the title for the Oy axis }
procedure TeX_PlotOxAxis; external 'dmath'; { Plots the horizontal axis }
procedure TeX_PlotOyAxis; external 'dmath'; { Plots the vertical axis }
procedure TeX_PlotGrid(Grid : TGrid); external 'dmath'; { Plots a grid on the graph }
procedure TeX_WriteGraphTitle; external 'dmath'; Writes the title of the graph }
procedure TeX_SetMaxCurv(NCurv : Byte); external 'dmath';
{ Sets the maximum number of curves and re-initializes their parameters }
procedure TeX_SetPointParam(CurvIndex, Symbol, Size : Integer); external 'dmath';
{ Sets the point parameters for curve # CurvIndex }
procedure TeX_SetLineParam(CurvIndex, Style : Integer;
                            Width : Float; Smooth : Boolean); external 'dmath';
{ Sets the line parameters for curve # CurvIndex }
procedure TeX_SetCurvLegend(CurvIndex : Integer; Legend : String); external 'dmath';
{ Sets the legend for curve # CurvIndex }
procedure TeX_SetCurvStep(CurvIndex, Step : Integer); external 'dmath';
{ Sets the step for curve # CurvIndex }
procedure TeX_PlotCurve(X, Y : TVector; Lb, Ub, CurvIndex : Integer); external 'dmath';
{ Plots a curve }
procedure TeX_PlotCurveWithErrorBars(X, Y, S : TVector;
                                      Ns, Lb, Ub, CurvIndex : Integer); external 'dmath';
{ Plots a curve with error bars }
procedure TeX_PlotFunc(Func : TFunc; X1, X2 : Float;
                        Npt : Integer; CurvIndex : Integer); external 'dmath';
{ Plots a function }
procedure TeX_WriteLegend(NCurv : Integer; ShowPoints, ShowLines : Boolean); external 'dmath';
{ Writes the legends for the plotted curves }
procedure TeX_ConRec(Nx, Ny, Nc : Integer; X, Y, Z  : TVector; F : TMatrix); external 'dmath';
{ Contour plot }
function Xcm(X : Float) : Float; external 'dmath'; { Converts user coordinate X to cm }
function Ycm(Y : Float) : Float; external 'dmath'; { Converts user coordinate Y to cm }

//*****************************************************unit uPSI_SynPdf;
 Function RawUTF8ToPDFString( const Value : RawUTF8) : PDFString
 Function _DateTimeToPdfDate( ADate : TDateTime) : TPdfDate
 Function _PdfDateToDateTime( const AText : TPdfDate) : TDateTime
 Function PdfRect( Left, Top, Right, Bottom : Single) : TPdfRect;
 Function PdfRect1( const Box : TPdfBox) : TPdfRect;
 Function PdfBox( Left, Top, Width, Height : Single) : TPdfBox
 //Function _GetCharCount( Text : PAnsiChar) : integer
 //Procedure L2R( W : PWideChar; L : integer)
 Function PdfCoord( MM : single) : integer
 Function CurrentPrinterPaperSize : TPDFPaperSize
 Function CurrentPrinterRes : TPoint
 Procedure GDICommentBookmark( MetaHandle : HDC; const aBookmarkName : RawUTF8)
 Procedure GDICommentOutline( MetaHandle : HDC; const aTitle : RawUTF8; aLevel : Integer)
 Procedure GDICommentLink( MetaHandle:HDC; const aBookmarkName:RawUTF8; const aRect : TRect)
 Const('Usp10','String 'usp10.dll
  AddTypeS('TScriptState_enum', '( r0, r1, r2, r3, r4, fOverrideDirection, fInhibitSymSwap,
   'fCharShape, fDigitSubstitute,fInhibitLigate,fDisplayZWG, fArabicNumContext, fGcpClusters )
  AddTypeS('TScriptState_set', 'set of TScriptState_enum
//***********************************************************************

procedure SIRegister_PMrand(CL: TPSPascalCompiler);  //ParkMiller
begin
 Procedure PMrandomize( I : word)
 Function PMrandom : longint
 Function Rrand : extended
 Function Irand( N : word) : word
 Function Brand( P : extended) : boolean
 Function Nrand : extended
end;

procedure SIRegister_Spring_Cryptography_Utils(CL: TPSPascalCompiler);
begin
  Function Endian( x : LongWord) : LongWord
  Function Endian64( x : Int64) : Int64
  Function spRol( x : LongWord; y : Byte) : LongWord
  Function spRor( x : LongWord; y : Byte) : LongWord
  Function Ror64( x : Int64; y : Byte) : Int64
end;

procedure SIRegister_MapReader(CL: TPSPascalCompiler);
begin
 Procedure ClearModules
 Procedure ReadMapFile( Fname : string)
 Function AddressInfo( Address : dword) : string
end;

procedure SIRegister_LibTar(CL: TPSPascalCompiler);
begin
  TTarPermission', '( tpReadByOwner, tpWriteByOwner, tpExecuteByOw'
   +'ner, tpReadByGroup, tpWriteByGroup, tpExecuteByGroup, tpReadByOther, tpWri'
   +'teByOther, tpExecuteByOther )
  TTarPermissions', 'set of TTarPermission
  TFileType', '( ftNormal, ftLink, ftSymbolicLink, ftCharacter, ft'
   +'Block, ftDirectory, ftFifo, ftContiguous, ftDumpDir, ftMultiVolume, ftVolumeHeader;
  TTarMode', '( tmSetUid, tmSetGid, tmSaveText )
  TTarModes', 'set of TTarMode
  TTarDirRec', 'record Name : STRING; Size : INT64; DateTime : TDa'
   +'teTime; Permissions : TTarPermissions; FileType : TFileType; LinkName : ST'
   +'RING; UID : INTEGER; GID : INTEGER; UserName : STRING; GroupName : STRING;'
   +' ChecksumOK : BOOLEAN; Mode : TTarModes; Magic : STRING; MajorDevNo : INTE'
   +'GER; MinorDevNo : INTEGER; FilePos : INT64; end
  SIRegister_TTarArchive(CL);
  SIRegister_TTarWriter(CL);
 Function PermissionString( Permissions : TTarPermissions) : STRING
 Function ConvertFilename( Filename : STRING) : STRING
 Function FileTimeGMT( FileName : STRING) : TDateTime;
 Function FileTimeGMT1( SearchRec : TSearchRec) : TDateTime;
 Procedure ClearDirRec( var DirRec : TTarDirRec)
end;


//*****************************************************unit uPSI_TlHelp32;
procedure SIRegister_TlHelp32(CL: TPSPascalCompiler);
begin
 Const('MAX_MODULE_NAME32','LongInt'( 255);
 Function CreateToolhelp32Snapshot( dwFlags, th32ProcessID : DWORD) : THandle
 Const('TH32CS_SNAPHEAPLIST','LongWord( $00000001);
 Const('TH32CS_SNAPPROCESS','LongWord').SetUInt( $00000002);
 Const('TH32CS_SNAPTHREAD','LongWord').SetUInt( $00000004);
 Const('TH32CS_SNAPMODULE','LongWord').SetUInt( $00000008);
 Const('TH32CS_INHERIT','LongWord').SetUInt( $80000000);
  tagHEAPLIST32','record dwSize:DWORD;th32ProcessID:DWORD;th32HeapID:DWORD;dwFlags:DWORD;end';
  AddTypeS('HEAPLIST32', 'tagHEAPLIST32
  AddTypeS('THeapList32', 'tagHEAPLIST32
 Const('HF32_DEFAULT','LongInt'( 1);
 Const('HF32_SHARED','LongInt'( 2);
 Function Heap32ListFirst( hSnapshot : THandle; var lphl : THeapList32) : BOOL
 Function Heap32ListNext( hSnapshot : THandle; var lphl : THeapList32) : BOOL
 AddTypeS('tagHEAPENTRY32', 'record dwSize : DWORD; hHandle : THandle; dwAd'
   +'dress : DWORD; dwBlockSize : DWORD; dwFlags : DWORD; dwLockCount : DWORD; '
   +'dwResvd : DWORD; th32ProcessID : DWORD; th32HeapID : DWORD; end
  AddTypeS('HEAPENTRY32', 'tagHEAPENTRY32
  AddTypeS('THeapEntry32', 'tagHEAPENTRY32
 Const('LF32_FIXED','LongWord').SetUInt( $00000001);
 Const('LF32_FREE','LongWord').SetUInt( $00000002);
 Const('LF32_MOVEABLE','LongWord').SetUInt( $00000004);
 Function Heap32First( var lphe : THeapEntry32; th32ProcessID, th32HeapID : DWORD) : BOOL
 Function Heap32Next( var lphe : THeapEntry32) : BOOL
 DWORD; var lpNumberOfBytesRead : DWORD) : BOOL
  AddTypeS('tagTHREADENTRY32', 'record dwSize : DWORD; cntUsage : DWORD; th3'
   +'2ThreadID : DWORD; th32OwnerProcessID : DWORD; tpBasePri : Longint; tpDelt'
   +'aPri : Longint; dwFlags : DWORD; end
  AddTypeS('THREADENTRY32', 'tagTHREADENTRY32
  AddTypeS('TThreadEntry32', 'tagTHREADENTRY32
 Function Thread32First( hSnapshot : THandle; var lpte : TThreadEntry32) : BOOL
 Function Thread32Next( hSnapshot : THandle; var lpte : TThreadENtry32) : BOOL
end;
 Const('EW_RESTARTWINDOWS','LongWord').SetUInt( $0042);
 Const('EW_REBOOTSYSTEM','LongWord( $0043);
 Const('EW_EXITANDEXECAPP','LongWord( $0044);
 Const('ENDSESSION_LOGOFF','LongWord').SetUInt( DWORD ( $80000000 ));
 Const('EWX_LOGOFF','LongInt'( 0);
 Const('EWX_SHUTDOWN','LongInt'( 1);
 Const('EWX_REBOOT','LongInt'( 2);
 Const('EWX_FORCE','LongInt'( 4);
 Const('EWX_POWEROFF','LongInt'( 8);
 Const('EWX_FORCEIFHUNG','LongWord').SetUInt( $10);
 Function GET_APPCOMMAND_LPARAM( const lParam : LongInt) : Shortint
 Function GET_DEVICE_LPARAM( const lParam : LongInt) : Word
 Function GET_MOUSEORKEY_LPARAM( const lParam : LongInt) : Word
 Function GET_FLAGS_LPARAM( const lParam : LongInt) : Word
 Function GET_KEYSTATE_LPARAM( const lParam : LongInt) : Word
 Function GetWindowWord( hWnd : HWND; nIndex : Integer) : Word
 Function SetWindowWord( hWnd : HWND; nIndex : Integer; wNewWord : Word) : Word
 Function GetWindowLong( hWnd : HWND; nIndex : Integer) : Longint
 Function SetWindowLong( hWnd : HWND; nIndex : Integer; dwNewLong : Longint) : Longint
 Function GetClassWord( hWnd : HWND; nIndex : Integer) : Word
 Function SetClassWord( hWnd : HWND; nIndex : Integer; wNewWord : Word) : Word
 Function GetClassLong( hWnd : HWND; nIndex : Integer) : DWORD
 Function SetClassLong( hWnd : HWND; nIndex : Integer; dwNewLong : Longint) : DWORD
 Function GetDesktopWindow : HWND
 Function GetParent( hWnd : HWND) : HWND
 Function SetParent( hWndChild, hWndNewParent : HWND) : HWND
 Function GetTopWindow( hWnd : HWND) : HWND
 Function GetNextWindow( hWnd : HWND; uCmd : UINT) : HWND
 Function GetWindow( hWnd : HWND; uCmd : UINT) : HWND
 //Delphi DFM
 Function LoadDFMFile2Strings(const AFile:string; AStrings:TStrings; var WasText:boolean):integer
 Function SaveStrings2DFMFile( AStrings : TStrings; const AFile : string) : integer
 procedure GetHighlighters(AOwner: TComponent; AHighlighters: TStringList; AppendToList: boolean);
 function GetHighlightersFilter(AHighlighters: TStringList): string;
 function GetHighlighterFromFileExt(AHighlighters: TStringList;Extension: string):TSynCustomHighlighter;
 Function ShowOwnedPopups( hWnd : HWND; fShow : BOOL) : BOOL
 Function OpenIcon( hWnd : HWND) : BOOL
 Function CloseWindow( hWnd : HWND) : BOOL
 Function MoveWindow( hWnd : HWND; X, Y, nWidth, nHeight : Integer; bRepaint : BOOL) : BOOL
 Function SetWindowPos(hWnd: HWND;hWndInsertAfter:HWND; X,Y,cx,cy : Integer; uFlags : UINT) : BOOL
 Function IsWindowVisible( hWnd : HWND) : BOOL
 Function IsIconic( hWnd : HWND) : BOOL
 Function AnyPopup : BOOL
 Function BringWindowToTop( hWnd : HWND) : BOOL
 Function IsZoomed( hWnd : HWND) : BOOL
 Function IsWindow( hWnd : HWND) : BOOL
 Function IsMenu( hMenu : HMENU) : BOOL
 Function IsChild( hWndParent, hWnd : HWND) : BOOL
 Function DestroyWindow( hWnd : HWND) : BOOL
 Function ShowWindow( hWnd : HWND; nCmdShow : Integer) : BOOL
 Function AnimateWindow( hWnd : HWND; dwTime : DWORD; dwFlags : DWORD) : BOOL
 Function ShowWindowAsync( hWnd : HWND; nCmdShow : Integer) : BOOL
 Function FlashWindow( hWnd : HWND; bInvert : BOOL) : BOOL
 Function IsWindowUnicode( hWnd : HWND) : BOOL
 Function EnableWindow( hWnd : HWND; bEnable : BOOL) : BOOL
 Function IsWindowEnabled( hWnd : HWND) : BOOL

procedure SIRegister_IDECmdLine(CL: TPSPascalCompiler);
begin
 const('ShowSetupDialogOptLong','String '--setup
 PrimaryConfPathOptLong','String '--primary-config-path=
 PrimaryConfPathOptShort','String '--pcp=
 SecondaryConfPathOptLong','String '--secondary-config-path=
 SecondaryConfPathOptShort','String '--scp=
 NoSplashScreenOptLong','String '--no-splash-screen
 NoSplashScreenOptShort','String '--nsc
 StartedByStartLazarusOpt','String '--started-by-startlazarus
 SkipLastProjectOpt','String '--skip-last-project
 DebugLogOpt','String '--debug-log=
 DebugLogOptEnable','String '--debug-enable=
 LanguageOpt','String '--language=
 LazarusDirOpt','String '--lazarusdir=
 Procedure ParseCommandLine(aCmdLineParams:TStrings;out IDEPid:Int;out ShowSplashScreen:boolean);
 Function GetCommandLineParameters( aCmdLineParams : TStrings; isStartLazarus:Boolean) : string
 Function ExtractPrimaryConfigPath( aCmdLineParams : TStrings) : string
 Function IsHelpRequested : Boolean
 Function IsVersionRequested : boolean
 Function GetLanguageSpecified : string
 Function ParamIsOption( ParamIndex : integer; const Option : string) : boolean
 Function ParamIsOptionPlusValue(ParamIndex:integer;const Option:string;out AValue:string):bool;
 Procedure ParseNoGuiCmdLineParams
 Function ExtractCmdLineFilenames : TStrings
end;


procedure SIRegister_LazFileUtils(CL: TPSPascalCompiler);
begin
 Function CompareFilenames( const Filename1, Filename2 : string) : integer
 Function CompareFilenamesIgnoreCase( const Filename1, Filename2 : string) : integer
 Function CompareFileExt( const Filename, Ext : string; CaseSensitive : boolean) : integer;
 Function CompareFileExt1( const Filename, Ext : string) : integer;
 Function CompareFilenameStarts( const Filename1, Filename2 : string) : integer
 Function CompareFilenames(Filename1:PChar;Len1:integer; Filename2:PChar;Len2:integer):integer
 Function CompareFilenamesP( Filename1, Filename2 : PChar; IgnoreCase : boolean) : integer
 Function DirPathExists( DirectoryName : string) : boolean
 Function DirectoryIsWritable( const DirectoryName : string) : boolean
 Function ExtractFileNameOnly( const AFilename : string) : string
 Function FilenameIsAbsolute( const TheFilename : string) : boolean
 Function FilenameIsWinAbsolute( const TheFilename : string) : boolean
 Function FilenameIsUnixAbsolute( const TheFilename : string) : boolean
 Function ForceDirectory( DirectoryName : string) : boolean
 Procedure CheckIfFileIsExecutable( const AFilename : string)
 Procedure CheckIfFileIsSymlink( const AFilename : string)
 Function FileIsText( const AFilename : string) : boolean
 Function FileIsText2( const AFilename : string; out FileReadable : boolean) : boolean
 Function FilenameIsTrimmed( const TheFilename : string) : boolean
 Function FilenameIsTrimmed2( StartPos : PChar; NameLen : integer) : boolean
 Function TrimFilename( const AFilename : string) : string
 Function ResolveDots( const AFilename : string) : string
 Procedure ForcePathDelims( var FileName : string)
 Function GetForcedPathDelims( const FileName : string) : String
 Function CleanAndExpandFilename( const Filename : string) : string
 Function CleanAndExpandDirectory( const Filename : string) : string
 Function TrimAndExpandFilename( const Filename : string; const BaseDir : string) : string
 Function TrimAndExpandDirectory( const Filename : string; const BaseDir : string) : string
 Function TryCreateRelativePath(const Dest,Source:String; UsePointDirectory:bool;
AlwaysRequireSharedBaseFolder : Boolean; out RelPath : String) : Boolean
 Function CreateRelativePath( const Filename,BaseDirectory:string; UsePointDirectory:boolean;
AlwaysRequireSharedBaseFolder: Boolean) : string
 Function FileIsInPath( const Filename, Path : string) : boolean
 Function AppendPathDelim( const Path : string) : string
 Function ChompPathDelim( const Path : string) : string
 Function CreateAbsoluteSearchPath( const SearchPath, BaseDirectory : string) : string
 Function CreateRelativeSearchPath( const SearchPath, BaseDirectory : string) : string
 Function MinimizeSearchPath( const SearchPath : string) : string
 Function FindPathInSearchPath(APath:PChar;APathLen:int;SearchPath:PChar;SearchPathLen:int):PChar;
 (*Function FileExistsUTF8( const Filename : string) : boolean
 Function FileAgeUTF8( const FileName : string) : Longint
 Function DirectoryExistsUTF8( const Directory : string) : Boolean
 Function ExpandFileNameUTF8( const FileName : string; BaseDir : string) : string
 Function FindFirstUTF8(const Path:string; Attr: Longint; out Rslt : TSearchRec) : Longint
 Function FindNextUTF8( var Rslt : TSearchRec) : Longint
 Procedure FindCloseUTF8( var F : TSearchrec)
 Function FileSetDateUTF8( const FileName : String; Age : Longint) : Longint
 Function FileGetAttrUTF8( const FileName : String) : Longint
 Function FileSetAttrUTF8( const Filename : String; Attr : longint) : Longint
 Function DeleteFileUTF8( const FileName : String) : Boolean
 Function RenameFileUTF8( const OldName, NewName : String) : Boolean
 Function FileSearchUTF8( const Name, DirList : String; ImplicitCurrentDir : Boolean) : String
 Function FileIsReadOnlyUTF8( const FileName : String) : Boolean
 Function GetCurrentDirUTF8 : String
 Function SetCurrentDirUTF8( const NewDir : String) : Boolean
 Function CreateDirUTF8( const NewDir : String) : Boolean
 Function RemoveDirUTF8( const Dir : String) : Boolean
 Function ForceDirectoriesUTF8( const Dir : string) : Boolean
 Function FileOpenUTF8( const FileName : string; Mode : Integer) : THandle
 Function FileCreateUTF8( const FileName : string) : THandle;
 Function FileCreateUTF81( const FileName : string; Rights : Cardinal) : THandle;
 Function FileCreateUtf82( const FileName : String; ShareMode : Integer; Rights : Cardinal) : THandle;
 Function FileSizeUtf8( const Filename : string) : int64
 Function GetFileDescription( const AFilename : string) : string
 Function GetAppConfigDirUTF8( Global : Boolean; Create : boolean) : string
 Function GetAppConfigFileUTF8( Global : Boolean; SubDir : boolean; CreateDir : boolean) : string
 Function GetTempFileNameUTF8( const Dir, Prefix : String) : String*)
 Function IsUNCPath( const Path : String) : Boolean
 Function ExtractUNCVolume( const Path : String) : String
 Function ExtractFileRoot( FileName : String) : String
 Function GetDarwinSystemFilename( Filename : string) : string
 Procedure SplitCmdLineParams( const Params : string; ParamList : TStrings; ReadBackslash : boolean)
 Function StrToCmdLineParam( const Param : string) : string
 Function MergeCmdLineParams( ParamList : TStrings) : string
 Procedure InvalidateFileStateCache( const Filename : string)
 Function FindAllFiles(const SearchPath:String;SearchMask:String;SearchSubDirs:Boolean):TStringList);
 Function FindAllDirectories( const SearchPath : string; SearchSubDirs : Boolean) : TStringList
 Function ReadFileToString( const Filename : string) : string
 type
  TCopyFileFlag = ( cffOverwriteFile,
    cffCreateDestDirectory, cffPreserveTime );
  TCopyFileFlags = set of TCopyFileFlag;*)
  TCopyFileFlag', '(cffOverwriteFile, cffCreateDestDirectory, cffPreserveTime)
  TCopyFileFlags', 'set of TCopyFileFlag
  Function CopyDirTree( const SourceDir, TargetDir : string; Flags : TCopyFileFlags) : Boolean
end;

procedure SIRegister_lazMasks(CL: TPSPascalCompiler);
begin
  TMaskCharType', '( mcChar, mcCharSet, mcAnyChar, mcAnyText )
  SIRegister_TMask(CL);
  SIRegister_TParseStringList(CL);
  SIRegister_TMaskList(CL);
 Function MatchesMask( const FileName, Mask : String; const CaseSensitive : Boolean) : Boolean
 Function MatchesWindowsMask( const FileName, Mask : String; const CaseSensitive : Boolean) : Bool;
 Function MatchesMaskList(const FileName,Mask:String;Separator:Char;const CaseSensitive:Boolean):Bool;
 Function MatchesWindowsMaskList(const FileName,Mask:String;Separat:Char;const CaseSensitive:Bool):Bool;
end;

procedure SIRegister_JvShellHook(CL: TPSPascalCompiler);
begin
  //PShellHookInfo', '^TShellHookInfo // will not work
  TShellHookInfo', 'record hwnd : THandle; rc : TRect; end
  SHELLHOOKINFO', 'TShellHookInfo
  LPSHELLHOOKINFO', 'PShellHookInfo
  TJvShellHookEvent', 'Procedure ( Sender : TObject; var Msg : TMessage)
  SIRegister_TJvShellHook(CL);
 Function InitJvShellHooks : Boolean
 Procedure UnInitJvShellHooks
end;

procedure SIRegister_JvExControls(CL: TPSPascalCompiler);
begin
  TDlgCode', '( dcWantAllKeys, dcWantArrows, dcWantChars, dcButton'
   +', dcHasSetSel, dcWantTab, dcNative )
  TDlgCodes', 'set of TDlgCode
 'dcWantMessage',' dcWantAllKeys);
  SIRegister_IJvExControl(CL);
  SIRegister_IJvDenySubClassing(CL);
  SIRegister_TStructPtrMessage(CL);
 Procedure SetDotNetFrameColors( FocusedColor, UnfocusedColor : TColor)
 Procedure DrawDotNetControl( Control : TWinControl; AColor : TColor; InControl : Boolean);
 Procedure DrawDotNetControl1( DC : HDC; R : TRect; AColor : TColor; UseFocusedColor : Boolean);
 Procedure HandleDotNetHighlighting(Control:TWinControl;const Msg:TMessage;MouseOver:Boolean;Color:TColor);
 Function CreateWMMessage( Msg : Integer; WParam : Integer; LParam : Longint) : TMessage;
 Function CreateWMMessage1( Msg : Integer; WParam : Integer; LParam : TControl) : TMessage;
 Function SmallPointToLong( const Pt : TSmallPoint) : Longint
 Function ShiftStateToKeyData( Shift : TShiftState) : Longint
 Function GetFocusedControl( AControl : TControl) : TWinControl
 Function DlgcToDlgCodes( Value : Longint) : TDlgCodes
 Function DlgCodesToDlgc( Value : TDlgCodes) : Longint
 Procedure GetHintColor( var HintInfo : THintInfo; AControl : TControl; HintColor : TColor)
 Function DispatchIsDesignMsg( Control : TControl; var Msg : TMessage) : Boolean
  SIRegister_TJvExControl(CL);
  SIRegister_TJvExWinControl(CL);
  SIRegister_TJvExCustomControl(CL);
  SIRegister_TJvExGraphicControl(CL);
  SIRegister_TJvExHintWindow(CL);
  SIRegister_TJvExPubGraphicControl(CL);
end;

(*----------------------------------------------------------------------------*)
procedure SIRegister_EncdDecd(CL: TPSPascalCompiler);
begin
 Procedure EncodeStream( Input, Output : TStream)
 Procedure DecodeStream( Input, Output : TStream)
 Function EncodeString1( const Input : string) : string
 Function DecodeString1( const Input : string) : string
end;

(*----------------------------------------------------------------------------*)
procedure SIRegister_SockAppReg(CL: TPSPascalCompiler);
begin
  SIRegister_TWebAppRegInfo(CL);
  SIRegister_TWebAppRegList(CL);
 Procedure GetRegisteredWebApps( AList : TWebAppRegList)
 Procedure RegisterWebApp( const AFileName, AProgID : string)
 Procedure UnregisterWebApp( const AProgID : string)
 Function FindRegisteredWebApp( const AProgID : string) : string
 Function CreateRegistry( InitializeNewFile : Boolean) : TCustomIniFile
 'sUDPPort','String 'UDPPort
end;

procedure SIRegister_PJEnvVars(CL: TPSPascalCompiler);
begin
 // TStringDynArray', 'array of string
 Function GetEnvVarValue( const VarName : string) : string
 Function SetEnvVarValue( const VarName, VarValue : string) : Integer
 Function DeleteEnvVar( const VarName : string) : Integer
 Function CreateEnvBlock(const NewEnv:TStrings;const IncludeCurrent:Bool;const Buffer:string;const
BufSize:Int):Int;
 Function ExpandEnvVars( const Str : string) : string
 Function GetAllEnvVars( const Vars : TStrings) : Integer
 Procedure GetAllEnvVarNames( const Names : TStrings);
 Function GetAllEnvVarNames1 : TStringDynArray;
 Function EnvBlockSize : Integer
  TPJEnvVarsEnum', 'Procedure ( const VarName : string; Data : TObject)
  SIRegister_TPJEnvVarsEnumerator(CL);
  SIRegister_TPJEnvVars(CL);
  FindClass('TOBJECT'),'EPJEnvVars
  FindClass('TOBJECT'),'EPJEnvVars
 //Procedure Register
end;

(*----------------------------------------------------------------------------*)
procedure SIRegister_PJConsoleApp(CL: TPSPascalCompiler);
begin
 'cOneSecInMS','LongInt'( 1000);
 //'cDefTimeSlice','LongInt'( 50);
 //'cDefMaxExecTime',' cOneMinInMS);
 'cAppErrorMask','LongInt'( 1 shl 29);
 Function IsApplicationError( const ErrCode : LongWord) : Boolean
  TPJConsoleAppPriority', '( cpDefault, cpHigh, cpNormal, cpIdle, cpRealTime )
  TPJConsoleColors', 'record Foreground : TPJConsoleColor; Background:TPJConsoleColor; end;
 Function MakeConsoleColors( const AForeground, ABackground : TPJConsoleColor):TPJConsoleColors;
 Function MakeConsoleColors1( const AForeground, ABackground : TColor) : TPJConsoleColors;
 Function MakeConsoleColors2( const AForeground, ABackground : TAlphaColor) : TPJConsoleColors;
 Function MakeSize( const ACX, ACY : LongInt) : TSize
  SIRegister_TPJCustomConsoleApp(CL);
  SIRegister_TPJConsoleApp(CL);
end;

procedure SIRegister_ip_misc(CL: TPSPascalCompiler);
begin
 INVALID_IP_ADDRESS','LongWord').SetUInt( $ffffffff);
```

```
  t_encoding', '( uuencode, base64, mime )
 Function internet_date( date : TDateTime) : string
 Function lookup_hostname( const hostname : string) : longint
 Function my_hostname : string
 Function my_ip_address : longint
 Function ip2string( ip_address : longint) : string
 Function resolve_hostname( ip : longint) : string
 Function address_from( const s : string; count : integer) : string
 Function encode_base64( data : TStream) : TStringList
 Function decode_base64( source : TStringList) : TMemoryStream
 Function posn( const s, t : string; count : integer) : integer
 Function poscn( c : char; const s : string; n : integer) : integer
 Function filename_of( const s : string) : string
 //Function trim( const s : string) : string
 //Procedure setlength( var s : string; l : byte)
 Function TimeZoneBias : longint
 Function eight2seven_quoteprint( const s : string) : string
 Function eight2seven_german( const s : string) : string
 Function seven2eight_quoteprint( const s : string) : string end;
  type in_addr', 'record s_bytes : array[1..4] of byte; end;
 Function socketerror : cint
 Function fpsocket( domain : cint; xtype : cint; protocol : cint) : cint
 Function fprecv( s : cint; buf : ___pointer; len : size_t; flags : cint) : ssize_t
 Function fpsend( s : cint; msg : ___pointer; len : size_t; flags : cint) : ssize_t
 //Function fpbind( s : cint; addrx : psockaddr; addrlen : tsocklen) : cint
 Function fplisten( s : cint; backlog : cint) : cint
 //Function fpaccept( s : cint; addrx : psockaddr; addrlen : plongint) : cint
 //Function fpconnect( s : cint; name : psockaddr; namelen : tsocklen) : cint
 //Function fpgetsockname( s : cint; name : psockaddr; namelen : psocklen) : cint
 Function NetAddrToStr( Entry : in_addr) : String
 Function HostAddrToStr( Entry : in_addr) : String
 Function StrToHostAddr( IP : String) : in_addr
 Function StrToNetAddr( IP : String) : in_addr
 SOL_SOCKET','LongWord').SetUInt( $ffff);
  cint8', 'shortint
  cuint8', 'byte
  cchar', 'cint8
  cschar', 'cint8
  cuchar', 'cuint8
  cint16', 'smallint
  cuint16', 'word
  cshort', 'cint16
  csshort', 'cint16
  cushort', 'cuint16
  cint32', 'longint
  cuint32', 'longword
  cint', 'cint32
  csint', 'cint32
  cuint', 'cuint32
  csigned', 'cint
  cunsigned', 'cuint
  cint64', 'int64
  clonglong', 'cint64
  cslonglong', 'cint64
  cbool', 'longbool
  cfloat', 'single
  cdouble', 'double
  clongdouble', 'extended

procedure SIRegister_uLkJSON(CL: TPSPascalCompiler);
begin
  TlkJSONtypes','(jsBase,jsNumber,jsString,jsBoolean,jsNull,jsList,jsObject )
  SIRegister_TlkJSONdotnetclass(CL);
  SIRegister_TlkJSONbase(CL);
  SIRegister_TlkJSONnumber(CL);
  SIRegister_TlkJSONstring(CL);
  SIRegister_TlkJSONboolean(CL);
  SIRegister_TlkJSONnull(CL);
  TlkJSONFuncEnum', 'Procedure ( ElName : string; Elem : TlkJSONba'
   +'se; data : TObject; var Continue : Boolean)
  SIRegister_TlkJSONcustomlist(CL);
  SIRegister_TlkJSONlist(CL);
  SIRegister_TlkJSONobjectmethod(CL);
  TlkHashItem', 'record hash : cardinal; index : Integer; end
  TlkHashFunction', 'Function ( const ws : WideString) : cardinal
  SIRegister_TlkHashTable(CL);
  SIRegister_TlkBalTree(CL);
  SIRegister_TlkJSONobject(CL);
  SIRegister_TlkJSON(CL);
  SIRegister_TlkJSONstreamed(CL);
 Function GenerateReadableText( vObj : TlkJSONbase; var vLevel : Integer): string
end;

procedure SIRegister_ZSysUtils(CL: TPSPascalCompiler);
begin
  TZListSortCompare', 'Function (Item1, Item2 : TObject): Integer
  SIRegister_TZSortedList(CL);
 Function zFirstDelimiter( const Delimiters, Str : string) : Integer
 Function zLastDelimiter( const Delimiters, Str : string) : Integer
 //Function MemLCompUnicode( P1, P2 : PWideChar; Len : Integer) : Boolean
 //Function MemLCompAnsi( P1, P2 : PAnsiChar; Len : Integer) : Boolean
 Function zStartsWith( const Str, SubStr : WideString) : Boolean;
 Function StartsWith1( const Str, SubStr : RawByteString) : Boolean;
 Function EndsWith( const Str, SubStr : WideString) : Boolean;
 Function EndsWith1( const Str, SubStr : RawByteString) : Boolean;
 Function SQLStrToFloatDef( Str : RawByteString; Def : Extended) : Extended;
 Function SQLStrToFloatDef1( Str : String; Def : Extended) : Extended;
 Function SQLStrToFloat( const Str : AnsiString) : Extended
 //Function BufferToStr( Buffer : PWideChar; Length : LongInt) : string;
 //Function BufferToStr1( Buffer : PAnsiChar; Length : LongInt) : string;
 Function BufferToBytes( Buffer : TObject; Length : LongInt) : TByteDynArray
 Function StrToBoolEx( Str : string) : Boolean
 Function BoolToStrEx( Bool : Boolean) : String
 Function IsIpAddr( const Str : string) : Boolean
 Function zSplitString( const Str, Delimiters : string) : TStrings
 Procedure PutSplitString( List : TStrings; const Str, Delimiters : string)
 Procedure AppendSplitString( List : TStrings; const Str, Delimiters : string)
 Function ComposeString( List : TStrings; const Delimiter : string) : string
 Function FloatToSQLStr( Value : Extended) : string
 Procedure PutSplitStringEx( List : TStrings; const Str, Delimiter : string)
 Function SplitStringEx( const Str, Delimiter : string) : TStrings
 Procedure AppendSplitStringEx( List : TStrings; const Str, Delimiter : string)
 Function zBytesToStr( const Value : TByteDynArray) : AnsiString
 Function zStrToBytes( const Value : AnsiString) : TByteDynArray;
 Function StrToBytes1( const Value : UTF8String) : TByteDynArray;
 Function StrToBytes2( const Value : RawByteString) : TByteDynArray;
 Function StrToBytes3( const Value : WideString) : TByteDynArray;
 Function StrToBytes4( const Value : UnicodeString) : TByteDynArray;
 Function BytesToVar( const Value : TByteDynArray) : Variant
 Function VarToBytes( const Value : Variant) : TByteDynArray
 Function AnsiSQLDateToDateTime( const Value : string) : TDateTime
 Function TimestampStrToDateTime( const Value : string) : TDateTime
 Function DateTimeToAnsiSQLDate( Value : TDateTime; WithMMSec : Boolean) : string
 Function EncodeCString( const Value : string) : string
 Function DecodeCString( const Value : string) : string
 Function zReplaceChar( const Source, Target : Char; const Str : string) : string
 Function MemPas( Buffer : PChar; Length : LongInt) : string
 Procedure DecodeSQLVersioning(const FullVersion:Int;out MajorVersion:Int;out MinorVersion:Int;out
SubVersion:Int);
 Function EncodeSQLVersioning(const MajorVersion:Integer;const MinorVersion:Integer;const
SubVersion:Integer):Int;
 Function FormatSQLVersion( const SQLVersion : Integer) : String
 Function ZStrToFloat( Value : AnsiChar) : Extended;
 Function ZStrToFloat1( Value : AnsiString) : Extended;
 Procedure ZSetString( const Src : AnsiChar; var Dest : AnsiString);
 Procedure ZSetString1( const Src : AnsiChar; const Len : Cardinal; var Dest : AnsiString);
 Procedure ZSetString2( const Src : AnsiChar; var Dest : UTF8String);
 Procedure ZSetString3( const Src : AnsiChar; const Len : Cardinal; var Dest : UTF8String);
 Procedure ZSetString4( const Src : AnsiChar; const Len : Cardinal; var Dest : WideString);
 Procedure ZSetString5( const Src : AnsiChar; var Dest : RawByteString);
 Procedure ZSetString6( const Src : AnsiChar; const Len : Cardinal; var Dest : RawByteString);
end;

unit uPSI_ZEncoding;
 Function StringToAnsiEx( const s : String; const FromCP, ToCP : Word) : RawByteString
 Function AnsiToStringEx( const s : RawByteString; const FromCP, ToCP : Word) : String
 Function ZRawToUnicode( const S : RawByteString; const CP : Word) : WideString
 Function ZUnicodeToRaw( const US : WideString; CP : Word) : RawByteString
 Function ZConvertAnsiToRaw( const Src : AnsiString; const RawCP : Word) : RawByteString
 Function ZConvertRawToAnsi( const Src : RawByteString; const RawCP : Word) : AnsiString
 Function ZConvertAnsiToUTF8( const Src : AnsiString) : UTF8String
 Function ZConvertUTF8ToAnsi( const Src : UTF8String) : AnsiString
 Function ZConvertRawToUTF8( const Src : RawByteString; const CP : Word) : UTF8String
 Function ZConvertUTF8ToRaw( const Src : UTF8String; const CP : Word) : RawByteString
 Function ZConvertRawToString( const Src : RawByteString; const RawCP, StringCP : Word) : String
 Function ZConvertStringToRaw( const Src : String; const StringCP, RawCP : Word) : RawByteString
 Function ZConvertStringToRawWithAutoEncode(const Src:String;const StringCP,RawCP:Word):RawByteString;
 Function ZConvertUTF8ToString( const Src : UTF8String; const StringCP : Word) : String
 Function ZConvertStringToUTF8( const Src : String; const StringCP : Word) : UTF8String
 Function ZConvertStringToUTF8WithAutoEncode( const Src : String; const StringCP: Word): UTF8String
 Function ZConvertStringToAnsi( const Src : String; const StringCP : Word) : AnsiString
 Function ZConvertStringToAnsiWithAutoEncode( const Src : String; const StringCP: Word): AnsiString
 Function ZConvertAnsiToString( const Src : AnsiString; const StringCP : Word) : String
 Function ZConvertUnicodeToString( const Src : WideString; const StringCP : Word) : String
 Function ZConvertUnicodeToString_CPUTF8( const Src : WideString; const StringCP : Word) : String
 Function ZConvertStringToUnicode( const Src : String; const StringCP : Word) : WideString
 Function ZConvertString_CPUTF8ToUnicode( const Src : String; const StringCP : Word) : WideString
 Function ZConvertStringToUnicodeWithAutoEncode( const Src: String; const StringCP:Word):WideString
 Function ZMoveAnsiToRaw( const Src : AnsiString; const RawCP : Word) : RawByteString
 Function ZMoveRawToAnsi( const Src : RawByteString; const RawCP : Word) : AnsiString
 Function ZMoveAnsiToUTF8( const Src : AnsiString) : UTF8String
 Function ZMoveUTF8ToAnsi( const Src : UTF8String) : AnsiString
 Function ZMoveRawToUTF8( const Src : RawByteString; const CP : Word) : UTF8String
 Function ZMoveUTF8ToRaw( const Src : UTF8String; const CP : Word) : RawByteString
 Function ZMoveStringToAnsi( const Src : String; const StringCP : Word) : AnsiString
 Function ZMoveAnsiToString( const Src : AnsiString; const StringCP : Word) : String
 Function ZMoveRawToString( const Src : RawByteString; const RawCP, StringCP : Word) : String
 Function ZMoveStringToRaw( const Src : String; const StringCP, RawCP : Word) : RawByteString
 Function ZMoveUTF8ToString( const Src : UTF8String; StringCP : Word) : String
 Function ZMoveStringToUTF8( const Src : String; const StringCP : Word) : UTF8String
 Function ZUnknownRawToUnicode( const S : RawByteString; const CP : Word) : WideString
 Function ZUnknownRawToUnicodeWithAutoEncode( const S : RawByteString; const CP : Word) : WideString
 Function ZUnicodeToUnknownRaw( const US : WideString; CP : Word) : RawByteString
 Function ZDefaultSystemCodePage : Word
 Function ZCompatibleCodePages( const CP1, CP2 : Word) : Boolean


procedure SIRegister_BoldComUtils(CL: TPSPascalCompiler);
begin
 'RPC_C_AUTHN_LEVEL_DEFAULT','LongInt'( 0);
 ('RPC_C_AUTHN_LEVEL_NONE','LongInt'( 1);
 ('RPC_C_AUTHN_LEVEL_CONNECT','LongInt'( 2);
 ('RPC_C_AUTHN_LEVEL_CALL','LongInt'( 3);
 ('RPC_C_AUTHN_LEVEL_PKT','LongInt'( 4);
 ('RPC_C_AUTHN_LEVEL_PKT_INTEGRITY','LongInt'( 5);
 ('RPC_C_AUTHN_LEVEL_PKT_PRIVACY','LongInt'( 6);
 {('alDefault','1 RPC_C_AUTHN_LEVEL_DEFAULT);
 ('alNone','2 RPC_C_AUTHN_LEVEL_NONE);
 ('alConnect','3 RPC_C_AUTHN_LEVEL_CONNECT);
 ('alCall','4 RPC_C_AUTHN_LEVEL_CALL);
 ('alPacket','5 RPC_C_AUTHN_LEVEL_PKT);
 ('alPacketIntegrity','6 RPC_C_AUTHN_LEVEL_PKT_INTEGRITY);
 ('alPacketPrivacy','7 RPC_C_AUTHN_LEVEL_PKT_PRIVACY);}
 ('RPC_C_IMP_LEVEL_DEFAULT','LongInt'( 0);
 ('RPC_C_IMP_LEVEL_ANONYMOUS','LongInt'( 1);
 ('RPC_C_IMP_LEVEL_IDENTIFY','LongInt'( 2);
 ('RPC_C_IMP_LEVEL_IMPERSONATE','LongInt'( 3);
 ('RPC_C_IMP_LEVEL_DELEGATE','LongInt'( 4);
 {('ilDefault','0 RPC_C_IMP_LEVEL_DEFAULT);
 ('ilAnonymous','1 RPC_C_IMP_LEVEL_ANONYMOUS);
 ('ilIdentiry','2 RPC_C_IMP_LEVEL_IDENTIFY);
 ('ilImpersonate','3 RPC_C_IMP_LEVEL_IMPERSONATE);
 ('ilDelegate','4 RPC_C_IMP_LEVEL_DELEGATE);}
 ('EOAC_NONE','LongWord').SetUInt( $0);
 ('EOAC_DEFAULT','LongWord').SetUInt( $800);
 ('EOAC_MUTUAL_AUTH','LongWord').SetUInt( $1);
 ('EOAC_STATIC_CLOACKING','LongWord').SetUInt( $20);
 ('EOAC_DYNAMIC_CLOAKING','LongWord').SetUInt( $40);
 ('EOAC_ANY_AUTHORITY','LongWord').SetUInt( $80);
 ('RPC_C_AUTHN_WINNT','LongInt'( 10);
 ('RPC_C_AUTHNZ_NONE','LongInt'( 0);
 ('RPC_C_AUTHNZ_NAME','LongInt'( 1);
 ('RPC_C_AUTHNZ_DCE','LongInt'( 2);
  FindClass('TOBJECT'),'EBoldCom
 Function BoldVariantIsType( V : OleVariant; TypeCode : Integer) : Boolean
 Function BoldMemoryToVariant( const Buffer, BufSize : Integer) : OleVariant
 Function BoldStreamToVariant( Stream : TStream) : OleVariant
 Function BoldStringsToVariant( Strings : TStrings) : OleVariant
 Function BoldVariantToMemory( V : OleVariant; var Buffer, BufSize : Integer) : Integer
 Function BoldVariantToStream( V : OleVariant; Stream : TStream) : Integer
 Function BoldVariantArrayOfArraysOfStringToStrings( V : OleVariant; Strings : TStrings) : Integer
 Function BoldVariantIsNamedValues( V : OleVariant) : Boolean
 Function BoldCreateNamedValues(const Names:array of string;const Values:array of OleVariant):OleVariant;
 Function BoldGetNamedValue( Data : OleVariant; const Name : string) : OleVariant
 Procedure BoldSetNamedValue( Data : OleVariant; const Name : string; Value : OleVariant)
 Function BoldCreateGUID : TGUID
 Function BoldCreateComObject( const ClsId, IId : TGUID; out Obj : variant; out Res : HResult) : Boolean
 Function BoldCreateRemoteComObject(const HostName:string;const ClsId,IId:TGUID;out Obj:variant;out
Res:HRes):Bool;
 Procedure BoldInitializeComSecurity( AuthenticationLevel, ImpersonationLevel : longint)
 Procedure BoldSetSecurityForInterface(AuthenticationLevel,ImpersonationLevel:longint;Unk:IUnknown);
end;

(*----------------------------------------------------------------------------*)
procedure SIRegister_BoldIsoDateTime(CL: TPSPascalCompiler);
begin
 Function ParseISODate( s : string) : TDateTime
 Function ParseISODateTime( s : string) : TDateTime
 Function ParseISOTime( str : string) : TDateTime
end;

(*----------------------------------------------------------------------------*)
procedure SIRegister_BoldGUIDUtils(CL: TPSPascalCompiler);
begin
 Function BoldCreateGUIDAsString( StripBrackets : Boolean) : string
 Function BoldCreateGUIDWithBracketsAsString : string
end;

procedure SIRegister_BoldFileHandler(CL: TPSPascalCompiler);
begin
  FindClass('TOBJECT'),'TBoldFileHandler
  FindClass('TOBJECT'),'TBoldDiskFileHandler
  //TBoldFileHandlerClass', 'class of TBoldFileHandler
  TBoldInitializeFileContents', 'Procedure ( StringList : TStringList)
  SIRegister_TBoldFileHandler(CL);
  SIRegister_TBoldDiskFileHandler(CL);
 Procedure BoldCloseAllFilehandlers
 Procedure BoldRemoveUnchangedFilesFromEditor
 Function BoldFileHandlerList : TBoldObjectArray
 Function BoldFileHandlerForFile(path,FileName:String; ModuleType:TBoldModuleType;ShowInEditor:Bool;
OnInitializeFileContents : TBoldInitializeFileContents) : TBoldFileHandler
end;

procedure SIRegister_BoldWinINet(CL: TPSPascalCompiler);
begin
  PCharArr', 'array of PChar
 Function BoldInternetOpen(Agent:String;
AccessType:integer;Proxy:string;ProxyByPass:String;Flags:integer):ptr);
 Function BoldInternetOpenUrl(iNet:Pointer;URL: string; Headers:String;Flags,Context:cardinal):Pointer
 Function BoldInternetReadFile(hFile:Pointer;Buff:Ptr;NumbOfBytesToRead:Card;var
NumberOfBytesRead:Card):LongBool;
 Function BoldInternetCloseHandle( HINet : Pointer) : LongBool
 Function BoldHttpQueryInfo( hRequest : Pointer; InfoLevel : Cardinal; Buffer : Pointer; BufferLength :
Cardinal; Reserved : Cardinal) : LongBool
 Function BoldInternetQueryDataAvailable( hFile : Pointer; var NumberOfBytesAvailable : Cardinal; flags :
Cardinal; Context : Cardinal) : LongBool
 Function BoldHttpOpenRequest(hConnect: Pointer; Verb, ObjectName, Version, Referrer : String; AcceptTypes
: PCharArr; Flags, Context : Cardinal) : Pointer
 Function BoldHttpSendRequest(hRequest:Ptr;Headers:string;Optional:Ptr;OptionalLength:Cardinal): LongBool
 Function BoldInternetErrorDlg(hWnd:HWND;hRequest:HINTERNET;dwError,dwFlags:DWORD;var lppvData:Ptr):DWORD
 Function BoldInternetAttemptConnect( dwReserved : DWORD) : DWORD
 Function BoldInternetConnect(hInet: HINTERNET;ServerName:string; nServerPort:INTERNET_PORT;
Username:string; Password : string; dwService : DWORD; dwFlags : DWORD; dwContext : DWORD):HINTERNET
 Function BoldInternetCrackUrl(Url:PChar;UrlLength,dwFlags:DWORD;var lpUrlComponents:TURLComponents):BOOL;
end;

procedure SIRegister_BoldQueryUserDlg(CL: TPSPascalCompiler);
begin
  TBoldQueryResult', '( qrYesAll, qrYes, qrNo, qrNoAll )
  SIRegister_TfrmBoldQueryUser(CL);
 Function QueryUser( const Title, Query : string) : TBoldQueryResult
end;

(*----------------------------------------------------------------------------*)
procedure SIRegister_BoldQueue(CL: TPSPascalCompiler);
begin
 //('befIsInDisplayList',' BoldElementFlag0);
 //('befStronglyDependedOfPrioritized',' BoldElementFlag1);
 //('befFollowerSelected',' BoldElementFlag2);
  FindClass('TOBJECT'),'TBoldQueue
  FindClass('TOBJECT'),'TBoldQueueable
  TBoldQueueDisplayMode', '( dmDisplayOne, dmDisplayAll )
  SIRegister_TBoldQueueable(CL);
  SIRegister_TBoldQueue(CL);
 Function BoldQueueFinalized : Boolean
 Function BoldInstalledQueue : TBoldQueue
end;

procedure SIRegister_Barcode(CL: TPSPascalCompiler);
begin
 const mmPerInch','Extended').setExtended( 25.4);
  TBarcodeType', '( bcCode_2_5_interleaved, bcCode_2_5_industrial,'
   +' bcCode_2_5_matrix, bcCode39, bcCode39Extended, bcCode128A, bcCode128B, bc'
   +'Code128C, bcCode93, bcCode93Extended, bcCodeMSI, bcCodePostNet, bcCodeCoda'
   +'bar, bcCodeEAN8, bcCodeEAN13, bcCodeUPC_A, bcCodeUPC_E0, bcCodeUPC_E1, bcC'
   +'odeUPC_Supp2, bcCodeUPC_Supp5, bcCodeEAN128A, bcCodeEAN128B, bcCodeEAN128C
  TBarLineType', '( white, black, black_half )
  TBarcodeOption', '( bcoNone, bcoCode, bcoTyp, bcoBoth )
  TShowTextPosition', '( stpTopLeft, stpTopRight, stpTopCenter, st'
   +'pBottomLeft, stpBottomRight, stpBottomCenter )
  TCheckSumMethod', '( csmNone, csmModulo10 )
  SIRegister_TAsBarcode(CL);
 Function CheckSumModulo10( const data : string) : string
 Function ConvertMmToPixelsX( const Value : Double) : Integer
 Function ConvertMmToPixelsY( const Value : Double) : Integer
 Function ConvertInchToPixelsX( const Value : Double) : Integer
 Function ConvertInchToPixelsY( const Value : Double) : Integer
end;

procedure SIRegister_Geometry(CL: TPSPascalCompiler);  //OpenGL
begin
   THomogeneousByteVector', 'array[0..3] of Byte
  THomogeneousWordVector', 'array[0..3] of Word
  THomogeneousIntVector', 'array[0..3] of Integer
  THomogeneousFltVector', 'array[0..3] of single
  THomogeneousDblVector', 'array[0..3] of double
  THomogeneousExtVector', 'array[0..3] of extended
  TAffineByteVector', 'array[0..2] of Byte
  TAffineWordVector', 'array[0..2] of Word
  TAffineIntVector', 'array[0..2] of Integer
  TAffineFltVector', 'array[0..2] of single
  TAffineDblVector', 'array[0..2] of double
  TAffineExtVector', 'array[0..2] of extended
  THomogeneousByteMatrix', 'array[0..3] of THomogeneousByteVector
  THomogeneousWordMatrix', 'array[0..3] of THomogeneousWordVector
  THomogeneousIntMatrix', 'array[0..3] of THomogeneousIntVector
  THomogeneousFltMatrix', 'array[0..3] of THomogeneousFltVector
  THomogeneousDblMatrix', 'array[0..3] of THomogeneousDblVector
  THomogeneousExtMatrix', 'array[0..3] of THomogeneousExtVector
  TAffineByteMatrix', 'array[0..2] of TAffineByteVector
  TAffineWordMatrix', 'array[0..2] of TAffineWordVector
  TAffineIntMatrix', 'array[0..2] of TAffineIntVector
  TAffineFltMatrix', 'array[0..3] of TAffineFltVector
  TAffineDblMatrix', 'array[0..3] of TAffineDblVector
  TAffineExtMatrix', 'array[0..3] of TAffineExtVector
  TMatrix4b', 'THomogeneousByteMatrix
  TMatrix4w', 'THomogeneousWordMatrix
  TMatrix4i', 'THomogeneousIntMatrix
  TMatrix4f', 'THomogeneousFltMatrix
  TMatrix4d', 'THomogeneousDblMatrix
  TMatrix4e', 'THomogeneousExtMatrix
  TMatrix3b', 'TAffineByteMatrix
  TMatrix3w', 'TAffineWordMatrix
  TMatrix3i', 'TAffineIntMatrix
  TMatrix3f', 'TAffineFltMatrix
  TMatrix3d', 'TAffineDblMatrix
  TMatrix3e', 'TAffineExtMatrix
  //'PMatrix', '^TMatrix // will not work
  TMatrixGL', 'THomogeneousFltMatrix
  THomogeneousMatrix', 'THomogeneousFltMatrix
  TAffineMatrix', 'TAffineFltMatrix
  TQuaternion', 'record Vector : TVector4f; end
  TRectangle', 'record Left : integer; Top : integer; Width : inte'
  +'ger; Height : Integer; end
  TTransType', '( ttScaleX, ttScaleY, ttScaleZ, ttShearXY, ttShear'
   +'XZ, ttShearYZ, ttRotateX, ttRotateY, ttRotateZ, ttTranslateX, ttTranslateY'
   +', ttTranslateZ, ttPerspectiveX, ttPerspectiveY, ttPerspectiveZ, ttPerspectiveW )
 'EPSILON','Extended').setExtended( 1E-100);
 'EPSILON2','Extended').setExtended( 1E-50);
 Function VectorAddGL( V1, V2 : TVectorGL) : TVectorGL
 Function VectorAffineAdd( V1, V2 : TAffineVector) : TAffineVector
 Function VectorAffineCombine(V1,V2:TAffineVector; F1, F2 : Single) : TAffineVector
 Function VectorAffineDotProduct( V1, V2 : TAffineVector) : Single
 Function VectorAffineLerp( V1, V2 : TAffineVector; t : Single) : TAffineVector
 Function VectorAffineSubtract( V1, V2 : TAffineVector) : TAffineVector
 Function VectorAngle( V1, V2 : TAffineVector) : Single
 Function VectorCombine( V1, V2 : TVectorGL; F1, F2 : Single) : TVectorGL
 Function VectorCrossProduct( V1, V2 : TAffineVector) : TAffineVector
 Function VectorDotProduct( V1, V2 : TVectorGL) : Single
 Function VectorLength( V : array of Single) : Single
 Function VectorLerp( V1, V2 : TVectorGL; t : Single) : TVectorGL
 Procedure VectorNegate( V : array of Single)
 Function VectorNorm( V : array of Single) : Single
 Function VectorNormalize( V : array of Single) : Single
 Function VectorPerpendicular( V, N : TAffineVector) : TAffineVector
 Function VectorReflect( V, N : TAffineVector) : TAffineVector
 Procedure VectorRotate( var Vector : TVector4f; Axis : TVector3f; Angle : Single)
 Procedure VectorScale( V : array of Single; Factor : Single)
 Function VectorSubtractGL( V1, V2 : TVectorGL) : TVectorGL
 Function CreateRotationMatrixX( Sine, Cosine : Single) : TMatrixGL
 Function CreateRotationMatrixY( Sine, Cosine : Single) : TMatrixGL
 Function CreateRotationMatrixZ( Sine, Cosine : Single) : TMatrixGL
 Function CreateScaleMatrix( V : TAffineVector) : TMatrixGL
 Function CreateTranslationMatrix( V : TVectorGL) : TMatrixGL
 Procedure MatrixAdjoint( var M : TMatrixGL)
 Function MatrixAffineDeterminant( M : TAffineMatrix) : Single
 Procedure MatrixAffineTranspose( var M : TAffineMatrix)
 Function MatrixDeterminant( M : TMatrixGL) : Single
 Procedure MatrixInvert( var M : TMatrixGL)
 Function MatrixMultiply( M1, M2 : TMatrixGL) : TMatrixGL
 Procedure MatrixScale( var M : TMatrixGL; Factor : Single)
 Procedure MatrixTranspose( var M : TMatrixGL)
 Function QuaternionConjugate( Q : TQuaternion) : TQuaternion
 Function QuaternionFromPoints( V1, V2 : TAffineVector) : TQuaternion
 Function QuaternionMultiply( qL, qR : TQuaternion) : TQuaternion
 Function QuaternionSlerp( QStart,QEnd:TQuaternion; Spin:Integer; t:Single):TQuaternion
 Function QuaternionToMatrix( Q : TQuaternion) : TMatrixGL
 Procedure QuaternionToPoints( Q : TQuaternion; var ArcFrom, ArcTo : TAffineVector)
 Function ConvertRotation( Angles : TAffineVector) : TVectorGL
 Function CreateRotationMatrix( Axis : TVector3f; Angle : Single) : TMatrixGL
 //Function MatrixDecompose( M : TMatrixGL; var Tran : TTransformations) : Boolean
 Function VectorAffineTransform( V : TAffineVector; M : TAffineMatrix) : TAffineVector
 Function VectorTransform( V : TVector4f; M : TMatrixGL) : TVector4f;
 Function VectorTransform1( V : TVector3f; M : TMatrixGL) : TVector3f;
 Function MakeAffineDblVector( V : array of Double) : TAffineDblVector
 Function MakeDblVector( V : array of Double) : THomogeneousDblVector
 Function MakeAffineVector( V : array of Single) : TAffineVector
 Function MakeQuaternion( Imag : array of Single; Real : Single) : TQuaternion
 Function MakeVector( V : array of Single) : TVectorGL
 Function PointInPolygonGL( xp, yp : array of Single; x, y : Single) : Boolean
 Function VectorAffineDblToFlt( V : TAffineDblVector) : TAffineVector
 Function VectorDblToFlt( V : THomogeneousDblVector) : THomogeneousVector
 Function VectorAffineFltToDbl( V : TAffineVector) : TAffineDblVector
 Function VectorFltToDbl( V : TVectorGL) : THomogeneousDblVector
 Function ArcCosGL( X : Extended) : Extended
 Function ArcSinGL( X : Extended) : Extended
 Function ArcTan2GL( Y, X : Extended) : Extended
 Function CoTanGL( X : Extended) : Extended
 Function DegToRadGL( Degrees : Extended) : Extended
 Function RadToDegGL( Radians : Extended) : Extended
 Procedure SinCosGL( Theta : Extended; var Sin, Cos : Extended)
 Function TanGL( X : Extended) : Extended
 Function Turn( Matrix : TMatrixGL; Angle : Single) : TMatrixGL;
 Function Turn1( Matrix : TMatrixGL; MasterUp : TAffineVector; Angle: Single): TMatrixGL;
 Function Pitch( Matrix : TMatrixGL; Angle : Single) : TMatrixGL;
 Function Pitch1( Matrix : TMatrixGL; MasterRight:TAffineVector;Angle:Single):TMatrixGL;
 Function Roll( Matrix : TMatrixGL; Angle : Single) : TMatrixGL;
 Function Roll1( Matrix:TMatrixGL; MasterDirection:TAffineVector;Angle:Single):TMatrixGL;
end;


procedure SIRegister_JclRegistry(CL: TPSPascalCompiler);
begin
 Function RegCreateKey( const RootKey : HKEY; const Key, Value : string) : Longint
 Function RegDeleteEntry( const RootKey : HKEY; const Key, Name : string) : Boolean
 Function RegDeleteKeyTree( const RootKey : HKEY; const Key : string) : Boolean
 Function RegReadBool( const RootKey : HKEY; const Key, Name : string) : Boolean
 Function RegReadBoolDef( const RootKey : HKEY; const Key, Name : string; Def : Boolean) : Boolean
 Function RegReadInteger( const RootKey : HKEY; const Key, Name : string) : Integer
 Function RegReadIntegerDef( const RootKey : HKEY; const Key, Name : string; Def : Integer) : Integer
 Function RegReadString( const RootKey : HKEY; const Key, Name : string) : string
 Function RegReadStringDef( const RootKey : HKEY; const Key, Name, Def : string) : string
 Function RegReadDWORD( const RootKey : HKEY; const Key, Name : string) : Int64
 Function RegReadDWORDDef( const RootKey : HKEY; const Key, Name : string; Def : Int64) : Int64
 Procedure RegWriteBool( const RootKey : HKEY; const Key, Name : string; Value : Boolean)
 Procedure RegWriteInteger( const RootKey : HKEY; const Key, Name : string; Value : Integer)
 Procedure RegWriteString( const RootKey : HKEY; const Key, Name, Value : string)
 Procedure RegWriteDWORD( const RootKey : HKEY; const Key, Name : string; Value : Int64)
 Function RegGetValueNames( const RootKey : HKEY; const Key : string; const List : TStrings) : Boolean
 Function RegGetKeyNames( const RootKey : HKEY; const Key : string; const List : TStrings) : Boolean
 Function RegHasSubKeys( const RootKey : HKEY; const Key : string) : Boolean
 Function RegKeyExists( const RootKey : HKEY; const Key : string) : Boolean
  AddTypeS('TExecKind', '( ekMachineRun, ekMachineRunOnce, ekUserRun, ekUser'
   +'RunOnce, ekServiceRun, ekServiceRunOnce )
  AddClassN(FindClass('TOBJECT'),'EJclRegistryError
 Function UnregisterAutoExec( ExecKind : TExecKind; const Name : string) : Boolean
 Function RegisterAutoExec( ExecKind : TExecKind; const Name, Cmdline : string) : Boolean
 Function RegSaveList(const RootKey:HKEY;const Key:string; const ListName:string;const
Items:TStrings):Bool;
 Function RegLoadList(const RootKey:HKEY;const Key:string;const ListName:string;const
SaveTo:TStrings):Bool;
 Function RegDelList( const RootKey:HKEY;const Key:string; const ListName:string): Boolean
end;

procedure SIRegister_JclCOM(CL: TPSPascalCompiler);
begin
 CLSID_StdComponentCategoriesMgr','TGUID '{0002E005-0000-0000-C000-000000000046}
 CATID_SafeForInitializing','TGUID '{7DD95802-9882-11CF-9FA9-00AA006C42C4}
 CATID_SafeForScripting','TGUID '{7DD95801-9882-11CF-9FA9-00AA006C42C4}
 icMAX_CATEGORY_DESC_LEN','LongInt'( 128);
  FindClass('TOBJECT'),'EInvalidParam
 Function IsDCOMInstalled : Boolean
 Function IsDCOMEnabled : Boolean
 Function GetDCOMVersion : string
 Function GetMDACVersion : string
 Function MarshalInterThreadInterfaceInVarArray(const iid:TIID;unk:IUnknown;var
VarArray:OleVariant):HResult;
 Function MarshalInterProcessInterfaceInStream(const iid:TIID;unk:IUnknown;var stm:IStream):HResult;
 Function MarshalInterProcessInterfaceInVarArray(const iid:TIID;unk:IUnknown;var
VarArray:OleVariant):HResult;
 Function MarshalInterMachineInterfaceInStream( const iid:TIID;unk:IUnknown;var stm:IStream):HResult;
 Function MarshalInterMachineInterfaceInVarArray(const iid:TIID;unk:IUnknown;var
VarArray:OleVariant):HResult;
 Function CreateComponentCategory( const CatID : TGUID; const sDescription : string) : HResult
 Function RegisterCLSIDInCategory( const ClassID : TGUID; const CatID : TGUID) : HResult
 Function UnRegisterCLSIDInCategory( const ClassID : TGUID; const CatID : TGUID) : HResult
 Function ResetIStreamToStart( Stream : IStream) : Boolean
 Function SizeOfIStreamContents( Stream : IStream) : Largeint
 Function StreamToVariantArray( Stream : TStream) : OleVariant;
 Function StreamToVariantArray1( Stream : IStream) : OleVariant;
 Procedure VariantArrayToStream( VarArray : OleVariant; var Stream : TStream);
 Procedure VariantArrayToStream1( VarArray : OleVariant; var Stream : IStream);
end;


procedure SIRegister_JclUnitConv_mX2(CL: TPSPascalCompiler);
begin
 Const('CelsiusFreezingPoint','Extended').setExtended( 0.0);
 FahrenheitFreezingPoint','Extended').setExtended( 32.0);
 KelvinFreezingPoint','Extended').setExtended( 273.15);
 CelsiusAbsoluteZero','Extended').setExtended( - 273.15);
 FahrenheitAbsoluteZero','Extended').setExtended( - 459.67);
 KelvinAbsoluteZero','Extended').setExtended( 0.0);
 DegPerCycle','Extended').setExtended( 360.0);
```

```
DegPerGrad','Extended').setExtended( 0.9);
DegPerRad','Extended').setExtended( 57.295779513082320876798154814105);
GradPerCycle','Extended').setExtended( 400.0);
GradPerDeg','Extended').setExtended( 1.1111111111111111111111111111111);
GradPerRad','Extended').setExtended( 63.661977236758134307553505349006);
RadPerCycle','Extended').setExtended( 6.283185307179586476925286766559);
RadPerDeg','Extended').setExtended( 0.017453292519943295769236907684886);
RadPerGrad','Extended').setExtended( 0.015707963267948966192313216916398);
CyclePerDeg','Extended').setExtended( 0.0027777777777777777777777777777778);
CyclePerGrad','Extended').setExtended( 0.0025);
CyclePerRad','Extended').setExtended( 0.15915494309189533576888376337251);
ArcMinutesPerDeg','Extended').setExtended( 60.0);
ArcSecondsPerArcMinute','Extended').setExtended( 60.0);
Function HowAOneLinerCanBiteYou( const Step, Max : Longint) : Longint
Function MakePercentage( const Step, Max : Longint) : Longint
Function CelsiusToKelvin( const T : double) : double
Function CelsiusToFahrenheit( const T : double) : double
Function KelvinToCelsius( const T : double) : double
Function KelvinToFahrenheit( const T : double) : double
Function FahrenheitToCelsius( const T : double) : double
Function FahrenheitToKelvin( const T : double) : double
Function CycleToDeg( const Cycles : double) : double
Function CycleToGrad( const Cycles : double) : double
Function CycleToRad( const Cycles : double) : double
Function DegToCycle( const Degrees : double) : double
Function DegToGrad( const Degrees : double) : double
Function DegToRad( const Degrees : double) : double
Function GradToCycle( const Grads : double) : double
Function GradToDeg( const Grads : double) : double
Function GradToRad( const Grads : double) : double
Function RadToCycle( const Radians : double) : double
Function RadToDeg( const Radians : double) : double
Function RadToGrad( const Radians : double) : double
Function DmsToDeg( const D, M : Integer; const S : double) : double
Function DmsToRad( const D, M : Integer; const S : double) : double
Procedure DegToDms( const Degrees : double; out D, M : Integer; out S : double)
Function DegToDmsStr( const Degrees : double; const SecondPrecision : Cardinal) : string
Procedure CartesianToPolar( const X, Y : double; out R, Phi : double)
Procedure PolarToCartesian( const R, Phi : double; out X, Y : double)
Procedure CartesianToCylinder( const X, Y, Z : double; out R, Phi, Zeta : double)
Procedure CartesianToSpheric( const X, Y, Z : double; out Rho, Phi, Theta : double)
Procedure CylinderToCartesian( const R, Phi, Zeta : double; out X, Y, Z : double)
Procedure SphericToCartesian( const Rho, Theta, Phi : double; out X, Y, Z : double)
Function CmToInch( const Cm : double) : double
Function InchToCm( const Inch : double) : double
Function FeetToMetre( const Feet : double) : double
Function MetreToFeet( const Metre : double) : double
Function YardToMetre( const Yard : double) : double
Function MetreToYard( const Metre : double) : double
Function NmToKm( const Nm : double) : double
Function KmToNm( const Km : double) : double
Function KmToSm( const Km : double) : double
Function SmToKm( const Sm : double) : double
Function LitreToGalUs( const Litre : double) : double
Function GalUsToLitre( const GalUs : double) : double
Function GalUsToGalCan( const GalUs : double) : double
Function GalCanToGalUs( const GalCan : double) : double
Function GalUsToGalUk( const GalUs : double) : double
Function GalUkToGalUs( const GalUk : double) : double
Function LitreToGalCan( const Litre : double) : double
Function GalCanToLitre( const GalCan : double) : double
Function LitreToGalUk( const Litre : double) : double
Function GalUkToLitre( const GalUk : double) : double
Function KgToLb( const Kg : double) : double
Function LbToKg( const Lb : double) : double
Function KgToOz( const Kg : double) : double
Function OzToKg( const Oz : double) : double
Function CwtUsToKg( const Cwt : double) : double
Function CwtUkToKg( const Cwt : double) : double
Function KaratToKg( const Karat : double) : double
Function KgToCwtUs( const Kg : double) : double
Function KgToCwtUk( const Kg : double) : double
Function KgToKarat( const Kg : double) : double
Function KgToSton( const Kg : double) : double
Function KgToLton( const Kg : double) : double
Function StonToKg( const STon : double) : double
Function LtonToKg( const Lton : double) : double
Function QrUsToKg( const Qr : double) : double
Function QrUkToKg( const Qr : double) : double
Function KgToQrUs( const Kg : double) : double
Function KgToQrUk( const Kg : double) : double
Function PascalToBar( const Pa : double) : double
Function PascalToAt( const Pa : double) : double
Function PascalToTorr( const Pa : double) : double
Function BarToPascal( const Bar : double) : double
Function AtToPascal( const At : double) : double
Function TorrToPascal( const Torr : double) : double
Function KnotToMs( const Knot : double) : double
Function HpElectricToWatt( const HpE : double) : double
```

```
 Function HpMetricToWatt( const HpM : double) : double
 Function MsToKnot( const ms : double) : double
 Function WattToHpElectric( const W : double) : double
 Function WattToHpMetric( const W : double) : double
 function getBigPI: string;    //PI of 1000 numbers

procedure SIRegister_devcutils(CL: TPSPascalCompiler);
begin
 Function CDExecuteFile( const FileName, Params, DefaultDir : string; ShowCmd : Integer) : THandle
 Procedure CDCopyFile( const FileName, DestName : string)
 Procedure CDMoveFile( const FileName, DestName : string)
 Function MakeCommaTextToColor( Text : string; Index : Integer; DefaultColor : TColor) : TColor
 Procedure CDDeleteFiles( Sender : TObject; s : string)
 Function CDGetTempDir : string
 Function CDGetFileSize( FileName : string) : longint
 Function GetFileTime( FileName : string) : longint
 Function GetShortName( FileName : string) : string
 Function GetFullName( FileName : string) : string
 Function WinReboot : boolean
 Function WinDir : String
 Function RunFile( FileToRun : string; Params : string; Dir : string; Wait : boolean) : cardinal
 Function RunFile_( Cmd, WorkDir : string; Wait : boolean) : Boolean
 Function devExecutor : TdevExecutor
end;

procedure SIRegister_FileAssocs(CL: TPSPascalCompiler);
begin
 Procedure CheckAssociations // AssociationsCount','LongInt'( 7);
 Procedure Associate( Index : integer)
 Procedure UnAssociate( Index : integer)
 Function IsAssociated( Index : integer) : boolean
 Function CheckFiletype( const extension, filetype, description, verb, serverapp : string) : boolean
 Procedure RegisterFiletype( const extension, filetype, description, verb, serverapp,IcoNum: string)
 Procedure RegisterDDEServer( const filetype, verb, topic, servername, macro : string)
 procedure RefreshIcons;
function GetShadeColor(ACanvas: TCanvas; clr: TColor; Value: integer): TColor;
function MergColor(Colors: Array of TColor): TColor;
function NewColor(ACanvas: TCanvas; clr: TColor; Value: integer): TColor;
procedure DimBitmap(ABitmap: TBitmap; Value: integer);
function GrayColor(ACanvas: TCanvas; clr: TColor; Value: integer): TColor;
function GetInverseColor(AColor: TColor): TColor;
procedure GrayBitmap(ABitmap: TBitmap; Value: integer);
procedure DrawBitmapShadow(B: TBitmap; ACanvas: TCanvas;X,Y: integer;ShadowColor: TColor);
procedure DrawCheckMark(ACanvas: TCanvas; X, Y: integer);
Procedure GetSystemMenuFont(Font: TFont);
end;

//**************************unit uPSI_JvHLParser;******************************
function IsStringConstant(const St: string): Boolean;
function IsIntConstant(const St: string): Boolean;
function IsRealConstant(const St: string): Boolean;
function IsIdentifier(const ID: string): Boolean;
function GetStringValue(const St: string): string;
procedure ParseString(const S: string; Ss: TStrings);
function IsStringConstantW(const St: WideString): Boolean;
function IsIntConstantW(const St: WideString): Boolean;
function IsRealConstantW(const St: WideString): Boolean;
function IsIdentifierW(const ID: WideString): Boolean;
function GetStringValueW(const St: WideString): WideString;
procedure ParseStringW(const S: WideString; Ss: TStrings);


//**************************unit uPSI_JclMapi;******************************

Function JclSimpleSendMail( const ARecipient,AName,ASubject, ABody : string; const AAttachment :
TFileName; ShowDialog : Boolean; AParentWND : HWND) : Boolean
Function JclSimpleSendFax( const ARecipient, AName,ASubject, ABody : string; const AAttachment :
TFileName; ShowDialog : Boolean; AParentWND : HWND) : Boolean
Function JclSimpleBringUpSendMailDialog(const ASubject,ABody:string;const
AAttach:TFileName;AParentWND:HWND):Bool
Function MapiCheck( const Res : DWORD; IgnoreUserAbort : Boolean) : DWORD
Function MapiErrorMessage( const ErrorCode : DWORD) : string

procedure SIRegister_IdNTLM(CL: TPSPascalCompiler);
begin
  //'Pdes_key_schedule', '^des_key_schedule // will not work
 Function BuildType1Message( ADomain, AHost : String) : String
 Function BuildType3Message(ADomain,AHost,AUsername:WideString;APassword,ANonce:String):String
 Procedure RegisterAuthenticationMethod( MethodName : String; AuthClass : TIdAuthenticationClass)
 Function FindAuthClass( AuthName : String) : TIdAuthenticationClass
GBase64CodeTable','string'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/
GXXECodeTable','string'+-0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz
GUUECodeTable','string'`!"#$%&''()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_
end;

procedure SIRegister_WDosSocketUtils(CL: TPSPascalCompiler);
begin
('IpAny','LongWord').SetUInt( $00000000);
 IpLoopBack','LongWord').SetUInt( $7F000001);
 IpBroadcast','LongWord').SetUInt( $FFFFFFFF);
 IpNone','LongWord').SetUInt( $FFFFFFFF);
 PortAny','LongWord( $0000);
 SocketMaxConnections','LongInt'( 5);
 TIpAddr', 'LongWord
 TIpRec', 'record IpB1 : byte; IpB2 : byte; IpB3 : byte; IpB4 : Byte; end
 Function HostToNetLong( HostLong : LongWord) : LongWord
 Function HostToNetShort( HostShort : Word) : Word
 Function NetToHostLong( NetLong : LongWord) : LongWord
 Function NetToHostShort( NetShort : Word) : Word
 Function StrToIp( Ip : string) : TIpAddr
 Function IpToStr( Ip : TIpAddr) : string
end;

(*----------------------------------------------------------------------------*)
procedure SIRegister_ALSMTPClient(CL: TPSPascalCompiler);
begin
  TAlSmtpClientAuthType', '( AlsmtpClientAuthNone, alsmtpClientAut'
   +'hPlain, AlsmtpClientAuthLogin, AlsmtpClientAuthCramMD5, AlsmtpClientAuthCr'
   +'amSha1, AlsmtpClientAuthAutoSelect )
  TAlSmtpClientAuthTypeSet', 'set of TAlSmtpClientAuthType
  SIRegister_TAlSmtpClient(CL);
end;

procedure SIRegister_WDosPlcUtils(CL: TPSPascalCompiler);
begin
'TBitNo', 'Integer
  TStByteNo', 'Integer
TStationNo', 'Integer
TInOutNo', 'Integer
TIo', '( EE, AA, NE, NA )
TBitSet', 'set of TBitNo
TAddrKind', 'set of ( akBit0, akBit1, akBit2, akOut, akNot, akBus )
TBitAddrRec', 'record Kind : TAddrKind; InOutNo : TInOutNo; ByteNo : Byte; end
TBitAddr', 'LongInt
TByteAddrRec', 'record Kind : TAddrKind; ByteNo : Byte; end
TByteAddr', 'SmallInt
TInOutState', '( iosInit, iosHalt, iosRun, iosError )
 Function BitAddr(aIo: TIo; aInOutNo : TInOutNo; aByteNo : Byte; aBitNo : TBitNo) : TBitAddr
 Function BusBitAddr(aIo:TIo;aInOutNo:TInOutNo;aStat:TStatNo;aStByteNo:TStByteNo;aBitNo:TBitNo):TBitAddr;
 Procedure BitAddrToValues(aBitAdr:TBitAdr;var aIo:TIo;var aInOutNo:TInOutNo;var aByteNo:Byte;var
aBitNo:TBitNo);
 Function BitAddrToStr( Value : TBitAddr) : string
 Function StrToBitAddr( const Value : string) : TBitAddr
 Function ByteAddr( aIo : TIo; aInOutNo : TInOutNo; aByteNo : Byte) : TByteAddr
 Function BusByteAddr(aIo:TIo;aInOutNo:TInOutNo;aStation:TStationNo;aStByteNo: TStByteNo):TByteAddr
 Procedure ByteAddrToValues(aByteAddr:TByteAddr;var aIo:TIo;var aInOutNo:TInOutNo;var aByteNo:Byte)
 Function ByteAddrToStr( Value : TByteAddr) : string
 Function StrToByteAddr( const Value : string) : TByteAddr
 Procedure IncByteAddr( var ByteAddr : TByteAddr; Increment : Integer)
 Procedure DecByteAddr( var ByteAddr : TByteAddr; Decrement : Integer)
 Function InOutStateToStr( State : TInOutState) : string
 Function MasterErrorToStr( ErrorCode : TErrorCode) : string
 Function SlaveErrorToStr( ErrorCode : TErrorCode) : string
end;

procedure SIRegister_WDosTimers(CL: TPSPascalCompiler);
begin
TIntFreq', '( ifNone, if32768, if16384, if8192, if4096, if2048, '
   +'if1024, if512, if256, if128, if64, if32, if16, if8, if4, if2 )
DpmiPmVector', 'Int64
 'DInterval','LongInt'( 1000);
 //'DEnabled','Boolean')BoolToStr( True);
 'DIntFreq','string' if64
 //'DMessages','Boolean if64);
  SIRegister_TwdxCustomTimer(CL);
  SIRegister_TwdxTimer(CL);
  SIRegister_TwdxRtcTimer(CL);
  SIRegister_TCustomIntTimer(CL);
  SIRegister_TIntTimer(CL);
  SIRegister_TRtcIntTimer(CL);
 Function RealNow : TDateTime
 Function MsToDateTime( MilliSecond : LongInt) : TDateTime
 Function DateTimeToMs( Time : TDateTime) : LongInt
end;

procedure SIRegister_IdSysLogMessage(CL: TPSPascalCompiler);
begin
TIdSyslogPRI', 'Integer
TIdSyslogFacility', '( sfKernel, sfUserLevel, sfMailSystem, sfSy'
   +'stemDaemon, sfSecurityOne, sfSysLogInternal, sfLPR, sfNNTP, sfUUCP, sfCloc'
   +'kDaemonOne, sfSecurityTwo, sfFTPDaemon, sfNTP, sfLogAudit, sfLogAlert, sfC'
   +'lockDaemonTwo, sfLocalUseZero, sfLocalUseOne, sfLocalUseTwo, sfLocalUseThr'
   +'ee, sfLocalUseFour, sfLocalUseFive, sfLocalUseSix, sfLocalUseSeven )
TIdSyslogSeverity','(slEmergency,slAlert,slCritical,slError,slWarning,slNotice,slInformational,slDebug)
  SIRegister_TIdSysLogMsgPart(CL);
  SIRegister_TIdSysLogMessage(CL);
 Function FacilityToString( AFac : TIdSyslogFacility) : string
 Function SeverityToString( ASec : TIdsyslogSeverity) : string
 Function NoToSeverity( ASev : Word) : TIdSyslogSeverity
 Function logSeverityToNo( ASev : TIdSyslogSeverity) : Word
 Function NoToFacility( AFac : Word) : TIdSyslogFacility
 Function logFacilityToNo( AFac : TIdSyslogFacility) : Word
end;

procedure SIRegister_TextUtils(CL: TPSPascalCompiler);
begin
 'UWhitespace','String '(?:\s*)
 Function StripSpaces( const AText : string) : string
 Function CharCount( const AText : string; Ch : Char) : Integer
 Function BalancedText( const AText : string; const Ch1, Ch2 : Char; const Count : Integer) : string
 Function BalancedTextReg( const AText:string; const Ch1, Ch2 : Char; const Count : Integer) : string
end;


procedure SIRegister_ExtPascalUtils(CL: TPSPascalCompiler);
begin
 ExtPascalVersion','String '0.9.8
  AddTypeS('TBrowser', '( brUnknown, brIE, brFirefox, brChrome, brSafari, br'
   +'Opera, brKonqueror, brMobileSafari )
  AddTypeS('TCSSUnit', '( cssPX, cssPerc, cssEM, cssEX, cssIN, cssCM, cssMM, cssPT, cssPC, cssnone )
  AddTypeS('TExtProcedure', 'Procedure
 Function DetermineBrowser( const UserAgentStr : string) : TBrowser
 Function ExtExtract(const Delims:array of string;var S:string;var Matches:TStringList;Remove:bool):bool;
 Function ExtExplode( Delim : char; const S : string; Separator : char) : TStringList
 Function FirstDelimiter( const Delimiters, S : string; Offset : integer) : integer
 Function RPosEx( const Substr, Str : string; Offset : integer) : integer
 Function CountStr( const Substr, Str : string; UntilStr : string) : integer
 Function StrToJS( const S : string; UseBR : boolean) : string
 Function CaseOf( const S : string; const Cases : array of string) : integer
 Function RCaseOf( const S : string; const Cases : array of string) : integer
 Function EnumToJSString( TypeInfo : PTypeInfo; Value : integer) : string
 Function SetPaddings(Top:integer;Right:int;Bottom:intr;Left:integer;CSSUnit:TCSSUnit;Header:bool):string;
 Function SetMargins(Top:integer;Right:int;Bottom:int;Left:integer;CSSUnit:TCSSUnit;Header:bool): string;
 Function ExtBefore( const BeforeS, AfterS, S : string) : boolean
 Function IsUpperCase( S : string) : boolean
 Function BeautifyJS(const AScript:string;const StartingLevel:integer;SplitHTMLNewLine: boolean):string;
 Function BeautifyCSS( const AStyle : string) : string
 Function LengthRegExp( Rex : string; CountAll : Boolean) : integer
 Function JSDateToDateTime( JSDate : string) : TDateTime
end;

procedure SIRegister_JclShell(CL: TPSPascalCompiler);
begin
  TSHDeleteOption', '( doSilent, doAllowUndo, doFilesOnly )
  TSHDeleteOptions', 'set of TSHDeleteOption
  TSHRenameOption', '( roSilent, roRenameOnCollision )
  TSHRenameOptions', 'set of TSHRenameOption
 Function SHDeleteFiles( Parent : HWND; const Files : string; Options : TSHDeleteOptions) : Boolean
 Function SHDeleteFolder( Parent : HWND; const Folder : string; Options : TSHDeleteOptions) : Boolean
 Function SHRenameFile( const Src, Dest : string; Options : TSHRenameOptions) : Boolean
  TEnumFolderFlag', '( efFolders, efNonFolders, efIncludeHidden )
  TEnumFolderFlags', 'set of TEnumFolderFlag
  TEnumFolderRec', 'record DisplayName : string; Attributes : DWOR'
   +'D; IconLarge : HICON; IconSmall : HICON; Item : PItemIdList; EnumIdList : '
   +'IEnumIdList; Folder : IShellFolder; end
 Function SHEnumFolderFirst(const Folder:string;Flags:TEnumFolderFlags;var F:TEnumFolderRec):Boolean;
 Function SHEnumSpecialFolderFirst(SpecialFolder:DWORD;Flags:TEnumFolderFlags;var F:TEnumFolderRec):Bool;
 Procedure SHEnumFolderClose( var F : TEnumFolderRec)
 Function SHEnumFolderNext( var F : TEnumFolderRec) : Boolean
 Function GetSpecialFolderLocation( const Folder : Integer) : string
 Function DisplayPropDialog( const Handle : HWND; const FileName : string) : Boolean;
 Function DisplayPropDialog1( const Handle : HWND; const Item : PItemIdList) : Boolean;
 Function DisplayContextMenu( const Handle : HWND; const FileName : string; Pos : TPoint) : Boolean
 Function OpenFolder( const Path : string; Parent : HWND) : Boolean
 Function OpenSpecialFolder( FolderID : Integer; Parent : HWND) : Boolean
 Function SHReallocMem( var P : Pointer; Count : Integer) : Boolean
 Function SHAllocMem( out P : Pointer; Count : Integer) : Boolean
 Function SHGetMem( var P : Pointer; Count : Integer) : Boolean
 Function SHFreeMem( var P : Pointer) : Boolean
 Function DriveToPidlBind( const DriveName : string; out Folder : IShellFolder) : PItemIdList
 Function PathToPidl( const Path : string; Folder : IShellFolder) : PItemIdList
 Function PathToPidlBind( const FileName : string; out Folder : IShellFolder) : PItemIdList
 Function PidlBindToParent(const IdList:PItemIdList;out Folder:IShellFolder;out Last:PItemIdList):Bool;
 Function PidlCompare( const Pidl1, Pidl2 : PItemIdList) : Boolean
 Function PidlCopy( const Source : PItemIdList; out Dest : PItemIdList) : Boolean
 Function PidlFree( var IdList : PItemIdList) : Boolean
 Function PidlGetDepth( const Pidl : PItemIdList) : Integer
 Function PidlGetLength( const Pidl : PItemIdList) : Integer
 Function PidlGetNext( const Pidl : PItemIdList) : PItemIdList
 Function PidlToPath( IdList : PItemIdList) : string
 Function StrRetFreeMem( StrRet : TStrRet) : Boolean
 Function StrRetToString( IdList : PItemIdList; StrRet : TStrRet; Free : Boolean) : string
  PShellLink', '^TShellLink // will not work
  TShellLink', 'record Arguments : string; ShowCmd : Integer; Work'
   +'ingDirectory : string; IdList : PItemIDList; Target : string; Description '
   +': string; IconLocation : string; IconIndex : Integer; HotKey : Word; end
 Procedure ShellLinkFree( var Link : TShellLink)
 Function ShellLinkResolve( const FileName : string; var Link : TShellLink) : HRESULT
 Function ShellLinkCreate( const Link : TShellLink; const FileName : string) : HRESULT
 Function ShellLinkCreateSystem(const Link:TShellLink;const Folder:Integer; const FileName:string):HRESULT;
 Function ShellLinkGetIcon( const Link : TShellLink; const Icon : TIcon) : Boolean
 Function SHDllGetVersion( const FileName : string; var Version : TDllVersionInfo) : Boolean
 Function GetSystemIcon( IconIndex : Integer; Flags : Cardinal) : HICON
 Function OverlayIcon( var Icon : HICON; Overlay : HICON; Large : Boolean) : Boolean
 Function OverlayIconShortCut( var Large, Small : HICON) : Boolean
 Function OverlayIconShared( var Large, Small : HICON) : Boolean
 Function SHGetItemInfoTip( const Folder : IShellFolder; Item : PItemIdList) : string
 Function ShellExecEx(const FileName:string;const Parameters:string;const Verb:string; CmdShow:Int):Bool;
 Function ShellExec(Wnd: Integer;const Operation,FileName,Parameters,Directy:string;ShowCommand:Int):Bool;
 Function ShellExecAndWait(const FileName:string;const Paramets:string;const Verb:string;CmdShow:Int):Bool;
 Function ShellOpenAs( const FileName : string) : Boolean
 Function ShellRasDial( const EntryName : string) : Boolean
 Function ShellRunControlPanel( const NameOrFileName:string; AppletNumber:Integer):Boolean
 Function GetFileNameIcon( const FileName : string; Flags : Cardinal) : HICON
  TJclFileExeType', '( etError, etMsDos, etWin16, etWin32Gui, etWin32Con )
 Function GetFileExeType( const FileName : TFileName) : TJclFileExeType
 Function ShellFindExecutable( const FileName, DefaultDir : string) : string
 Procedure keybd_event( bVk : Byte; bScan : Byte; dwFlags, dwExtraInfo : DWORD)
 Function OemKeyScan( wOemChar : Word) : DWORD
 Procedure mouse_event( dwFlags, dx, dy, dwData, dwExtraInfo : DWORD)
end;

procedure SIRegister_cXMLFunctions(CL: TPSPascalCompiler);
begin
 xmlVersion','String '1.0 FindClass('TOBJECT'),'Exml
 //Function xmlValidChar( const Ch : AnsiChar) : Boolean;
 Function xmlValidChar1( const Ch : UCS4Char) : Boolean;
 Function xmlValidChar2( const Ch : WideChar) : Boolean;
 Function xmlIsSpaceChar( const Ch : WideChar) : Boolean
 Function xmlIsLetter( const Ch : WideChar) : Boolean
 Function xmlIsDigit( const Ch : WideChar) : Boolean
 Function xmlIsNameStartChar( const Ch : WideChar) : Boolean
 Function xmlIsNameChar( const Ch : WideChar) : Boolean
 Function xmlIsPubidChar( const Ch : WideChar) : Boolean
 Function xmlValidName( const Text : UnicodeString) : Boolean
 //xmlSpace','Char #$20 or  #$9 or  #$D or  #$A);
 //Function xmlSkipSpace( var P : PWideChar) : Boolean
 //Function xmlSkipEq( var P : PWideChar) : Boolean
 //Function xmlExtractQuotedText( var P : PWideChar; var S : UnicodeString) : Boolean
 //Function xmlGetEntityEncoding( const Buf : Pointer; const BufSize : Integer; out HeaderSize : Integer)
: TUnicodeCodecClass
 Function xmlResolveEntityReference( const RefName : UnicodeString) : WideChar
 Function xmlTag( const Tag : UnicodeString) : UnicodeString
 Function xmlEndTag( const Tag : UnicodeString) : UnicodeString
 Function xmlAttrTag( const Tag : UnicodeString; const Attr : UnicodeString) : UnicodeString
 Function xmlEmptyTag( const Tag, Attr : UnicodeString) : UnicodeString
 Procedure xmlSafeTextInPlace( var Txt : UnicodeString)
 Function xmlSafeText( const Txt : UnicodeString) : UnicodeString
 Function xmlSpaceIndent( const IndentLength : Integer; const IndentLevel : Integer):UnicodeString
 Function xmlTabIndent( const IndentLevel : Integer) : UnicodeString
 Function xmlComment( const Comment : UnicodeString) : UnicodeString
 Procedure SelfTestcXMLFunctions
end;

(*----------------------------------------------------------------------------*)
procedure SIRegister_DepWalkUtils(CL: TPSPascalCompiler);
begin
 Function AWaitCursor : IUnknown
 Function ChangeCursor( NewCursor : TCursor) : IUnknown
 Procedure SuspendRedraw( AControl : TWinControl; Suspend : boolean)
 Function YesNo( const ACaption, AMsg : string) : boolean
 Procedure strTokenize( const S : string; Delims : TSysCharSet; Results : TStrings)
 Function GetBorlandLibPath( Version : integer; ForDelphi : boolean) : string
 Function GetExpandedLibRoot( Version : integer; ForDelphi : boolean) : string
 Procedure GetPathList( Version : integer; ForDelphi : boolean; Strings : TStrings)
 Procedure GetSystemPaths( Strings : TStrings)
 Procedure MakeEditNumeric( EditHandle : integer)
end;

procedure SIRegister_yuvconverts(CL: TPSPascalCompiler);
begin
  AddTypeS('TVideoCodec','(vcUnknown,vcRGB,vcYUY2,vcUYVY,vcBTYUV,vcYV,U9,vcYUV12,vcY8,vcY211)
 'BI_YUY2','LongWord( $32595559);
 'BI_UYVY','LongWord').SetUInt( $59565955);
 'BI_BTYUV','LongWord').SetUInt( $50313459);
 'BI_YVU9','LongWord').SetUInt( $39555659);
 'BI_YUV12','LongWord( $30323449);
 'BI_Y8','LongWord').SetUInt( $20203859);
 'BI_Y211','LongWord').SetUInt( $31313259);
 Function BICompressionToVideoCodec( Value : DWord) : TVideoCodec
 Function ConvertCodecToRGB(Codec:TVideoCodec;Src,Dst:Pointer;AWidth,AHeight:Integer):Boolean;
end;

(*----------------------------------------------------------------------------*)
procedure SIRegister_AviCap(CL: TPSPascalCompiler);
begin
```

```
 'WM_USER','LongWord').SetUInt( $0400);
 'WM_CAP_START','LongWord').SetUint($0400);
 'WM_CAP_END','longword').SetUint($0400+85);
 //WM_CAP_START+  85
 //    WM_CAP_SET_CALLBACK_CAPCONTROL  = (WM_CAP_START+  85);
 Function capSetCallbackOnError( hwnd : THandle; fpProc : LongInt) : LongInt
 Function capSetCallbackOnStatus( hwnd : THandle; fpProc : LongInt) : LongInt
 Function capSetCallbackOnYield( hwnd : THandle; fpProc : LongInt) : LongInt
 Function capSetCallbackOnFrame( hwnd : THandle; fpProc : LongInt) : LongInt
 Function capSetCallbackOnVideoStream( hwnd : THandle; fpProc : LongInt) : LongInt
 Function capSetCallbackOnWaveStream( hwnd : THandle; fpProc : LongInt) : LongInt
 Function capSetCallbackOnCapControl( hwnd : THandle; fpProc : LongInt) : LongInt
 Function capSetUserData( hwnd : THandle; lUser : LongInt) : LongInt
 Function capGetUserData( hwnd : THandle) : LongInt
 Function capDriverConnect( hwnd : THandle; I : Word) : LongInt
 Function capDriverDisconnect( hwnd : THandle) : LongInt
 Function capDriverGetName( hwnd : THandle; szName : LongInt; wSize : Word) : LongInt
 Function capDriverGetVersion( hwnd : THandle; szVer : LongInt; wSize : Word) : LongInt
 Function capDriverGetCaps( hwnd : THandle; s : LongInt; wSize : Word) : LongInt
 Function capFileSetCaptureFile( hwnd : THandle; szName : LongInt) : LongInt
 Function capFileGetCaptureFile( hwnd : THandle; szName : LongInt; wSize : Word):LongInt
 Function capFileAlloc( hwnd : THandle; dwSize : LongInt) : LongInt
 Function capFileSaveAs( hwnd : THandle; szName : LongInt) : LongInt
 Function capFileSetInfoChunk( hwnd : THandle; lpInfoChunk : LongInt) : LongInt
 Function capFileSaveDIB( hwnd : THandle; szName : LongInt) : LongInt
 Function capEditCopy( hwnd : THandle) : LongInt
 Function capSetAudioFormat( hwnd : THandle; s : LongInt; wSize : Word) : LongInt
 Function capGetAudioFormat( hwnd : THandle; s : LongInt; wSize : Word) : LongInt
 Function capGetAudioFormatSize( hwnd : THandle) : LongInt
 Function capDlgVideoFormat( hwnd : THandle) : LongInt
 Function capDlgVideoSource( hwnd : THandle) : LongInt
 Function capDlgVideoDisplay( hwnd : THandle) : LongInt
 Function capDlgVideoCompression( hwnd : THandle) : LongInt
 Function capGetVideoFormat( hwnd : THandle; s : LongInt; wSize : Word) : LongInt
 Function capGetVideoFormatSize( hwnd : THandle) : LongInt
 Function capSetVideoFormat( hwnd : THandle; s : LongInt; wSize : Word) : LongInt
 Function capPreview( hwnd : THandle; f : Word) : LongInt
 Function capPreviewRate( hwnd : THandle; wMS : Word) : LongInt
 Function capOverlay( hwnd : THandle; f : Word) : LongInt
 Function capPreviewScale( hwnd : THandle; f : Word) : LongInt
 Function capGetStatus( hwnd : THandle; s : LongInt; wSize : Word) : LongInt
 Function capSetScrollPos( hwnd : THandle; lpP : LongInt) : LongInt
 Function capGrabFrame( hwnd : THandle) : LongInt
 Function capGrabFrameNoStop( hwnd : THandle) : LongInt
 Function capCaptureSequence( hwnd : THandle) : LongInt
 Function capCaptureSequenceNoFile( hwnd : THandle) : LongInt
 Function capCaptureStop( hwnd : THandle) : LongInt
 Function capCaptureAbort( hwnd : THandle) : LongInt
 Function capCaptureSingleFrameOpen( hwnd : THandle) : LongInt
 Function capCaptureSingleFrameClose( hwnd : THandle) : LongInt
 Function capCaptureSingleFrame( hwnd : THandle) : LongInt
 Function capCaptureGetSetup( hwnd : THandle; s : LongInt; wSize : Word) : LongInt
 Function capCaptureSetSetup( hwnd : THandle; s : LongInt; wSize : Word) : LongInt
 Function capSetMCIDeviceName( hwnd : THandle; szName : LongInt) : LongInt
 Function capGetMCIDeviceName( hwnd : THandle; szName : LongInt; wSize : Word) : LongInt
 Function capPaletteOpen( hwnd : THandle; szName : LongInt) : LongInt
 Function capPaletteSave( hwnd : THandle; szName : LongInt) : LongInt
 Function capPalettePaste( hwnd : THandle) : LongInt
 Function capPaletteAuto( hwnd : THandle; iFrames : Word; iColors : LongInt) : LongInt
 Function capPaletteManual( hwnd : THandle; fGrab : Word; iColors : LongInt) : LongInt
  //PCapDriverCaps', '^TCapDriverCaps // will not work
  TCapDriverCaps', 'record wDeviceIndex : WORD; fHasOverlay : BOOL'
   +'; fHasDlgVideoSource : BOOL; fHasDlgVideoFormat : BOOL; fHasDlgVideoDispla'
   +'y : BOOL; fCaptureInitialized : BOOL; fDriverSuppliesPalettes : BOOL; hVid'
   +'eoIn : THANDLE; hVideoOut : THANDLE; hVideoExtIn:THANDLE; hVideoExtOut:THANDLE; end
  //PCapStatus', '^TCapStatus // will not work
  TCapStatus', 'record uiImageWidth : UINT; uiImageHeight : UINT; '
   +'fLiveWindow : BOOL; fOverlayWindow : BOOL; fScale : BOOL; ptScroll : TPOIN'
   +'T; fUsingDefaultPalette : BOOL; fAudioHardware : BOOL; fCapFileExists : BO'
   +'OL; dwCurrentVideoFrame : DWORD; dwCurrentVideoFramesDropped : DWORD; dwCu'
   +'rrentWaveSamples : DWORD; dwCurrentTimeElapsedMS : DWORD; hPalCurrent : HP'
   +'ALETTE; fCapturingNow : BOOL; dwReturn : DWORD; wNumVideoAllocated : WORD;'
   +' wNumAudioAllocated : WORD; end
  //PCaptureParms', '^TCaptureParms // will not work
  TCaptureParms', 'record dwRequestMicroSecPerFrame : DWORD; fMake'
   +'UserHitOKToCapture : BOOL; wPercentDropForError : WORD; fYield : BOOL; dwI'
   +'ndexSize : DWORD; wChunkGranularity : WORD; fUsingDOSMemory : BOOL; wNumVi'
   +'deoRequested : WORD; fCaptureAudio : BOOL; wNumAudioRequested : WORD; vKey'
   +'Abort : WORD; fAbortLeftMouse : BOOL; fAbortRightMouse : BOOL; fLimitEnabl'
   +'ed : BOOL; wTimeLimit : WORD; fMCIControl : BOOL; fStepMCIDevice : BOOL; d'
   +'wMCIStartTime : DWORD; dwMCIStopTime : DWORD; fStepCaptureAt2x : BOOL; wSt'
   +'epCaptureAverageFrames : WORD; dwAudioBufferSize : DWORD; fDisableWriteCac'
   +'he : BOOL; AVStreamMaster : WORD; end
 // PCapInfoChunk', '^TCapInfoChunk // will not work
  //TCapInfoChunk', 'record fccInfoID : FOURCC; lpData : LongInt; cbData : LongInt; end
 'CONTROLCALLBACK_PREROLL','LongInt'( 1);
 'CONTROLCALLBACK_CAPTURING','LongInt'( 2);
 Function capCreateCaptureWindow( lpszWindowName: PChar; dwStyle : DWord; x, y : Integer; nWidth, nHeight
: Integer; hwndParent : THandle; nID : Integer) : THandle
```

```
 Function
capGetDriverDescription(wDriverIndex:DWord;lpszName:PChar;cbName:Integer;lpszVer:PChar;cbVer:Int):Bool;
 'IDS_CAP_BEGIN','LongInt'( 300);
 'IDS_CAP_END','LongInt'( 301);
 'IDS_CAP_INFO','LongInt'( 401);
 'IDS_CAP_OUTOFMEM','LongInt'( 402);
 'IDS_CAP_FILEEXISTS','LongInt'( 403);
 'IDS_CAP_ERRORPALOPEN','LongInt'( 404);
 'IDS_CAP_ERRORPALSAVE','LongInt'( 405);
 'IDS_CAP_ERRORDIBSAVE','LongInt'( 406);
 'IDS_CAP_DEFAVIEXT','LongInt'( 407);
 'IDS_CAP_DEFPALEXT','LongInt'( 408);
 'IDS_CAP_CANTOPEN','LongInt'( 409);
 'IDS_CAP_SEQ_MSGSTART','LongInt'( 410);
 'IDS_CAP_SEQ_MSGSTOP','LongInt'( 411);
 'IDS_CAP_VIDEDITERR','LongInt'( 412);
 'IDS_CAP_READONLYFILE','LongInt'( 413);
 'IDS_CAP_WRITEERROR','LongInt'( 414);
 'IDS_CAP_NODISKSPACE','LongInt'( 415);
 'IDS_CAP_SETFILESIZE','LongInt'( 416);
 'IDS_CAP_SAVEASPERCENT','LongInt'( 417);
 'IDS_CAP_DRIVER_ERROR','LongInt'( 418);
 'IDS_CAP_WAVE_OPEN_ERROR','LongInt'( 419);
 'IDS_CAP_WAVE_ALLOC_ERROR','LongInt'( 420);
 'IDS_CAP_WAVE_PREPARE_ERROR','LongInt'( 421);
 'IDS_CAP_WAVE_ADD_ERROR','LongInt'( 422);
 'IDS_CAP_WAVE_SIZE_ERROR','LongInt'( 423);
 'IDS_CAP_VIDEO_OPEN_ERROR','LongInt'( 424);
 'IDS_CAP_VIDEO_ALLOC_ERROR','LongInt'( 425);
 'IDS_CAP_VIDEO_PREPARE_ERROR','LongInt'( 426);
 'IDS_CAP_VIDEO_ADD_ERROR','LongInt'( 427);
 'IDS_CAP_VIDEO_SIZE_ERROR','LongInt'( 428);
 'IDS_CAP_FILE_OPEN_ERROR','LongInt'( 429);
 'IDS_CAP_FILE_WRITE_ERROR','LongInt'( 430);
 'IDS_CAP_RECORDING_ERROR','LongInt'( 431);
 'IDS_CAP_RECORDING_ERROR2','LongInt'( 432);
 'IDS_CAP_AVI_INIT_ERROR','LongInt'( 433);
 'IDS_CAP_NO_FRAME_CAP_ERROR','LongInt'( 434);
 'IDS_CAP_NO_PALETTE_WARN','LongInt'( 435);
 'IDS_CAP_MCI_CONTROL_ERROR','LongInt'( 436);
 'IDS_CAP_MCI_CANT_STEP_ERROR','LongInt'( 437);
 'IDS_CAP_NO_AUDIO_CAP_ERROR','LongInt'( 438);
 'IDS_CAP_AVI_DRAWDIB_ERROR','LongInt'( 439);
 'IDS_CAP_COMPRESSOR_ERROR','LongInt'( 440);
 'IDS_CAP_AUDIO_DROP_ERROR','LongInt'( 441);
 'IDS_CAP_STAT_LIVE_MODE','LongInt'( 500);
 'IDS_CAP_STAT_OVERLAY_MODE','LongInt'( 501);
 'IDS_CAP_STAT_CAP_INIT','LongInt'( 502);
 'IDS_CAP_STAT_CAP_FINI','LongInt'( 503);
 'IDS_CAP_STAT_PALETTE_BUILD','LongInt'( 504);
 'IDS_CAP_STAT_OPTPAL_BUILD','LongInt'( 505);
 'IDS_CAP_STAT_I_FRAMES','LongInt'( 506);
 'IDS_CAP_STAT_L_FRAMES','LongInt'( 507);
 'IDS_CAP_STAT_CAP_L_FRAMES','LongInt'( 508);
 'IDS_CAP_STAT_CAP_AUDIO','LongInt'( 509);
 'IDS_CAP_STAT_VIDEOCURRENT','LongInt'( 510);
 'IDS_CAP_STAT_VIDEOAUDIO','LongInt'( 511);
 'IDS_CAP_STAT_VIDEOONLY','LongInt'( 512);
 'IDS_CAP_STAT_FRAMESDROPPED','LongInt'( 513);
 'AVICAP32','String 'AVICAP32.dll
end;

procedure SIRegister_ALFcnMisc(CL: TPSPascalCompiler);
begin
 Function AlBoolToInt( Value : Boolean) : Integer
 Function ALMediumPos( LTotal, LBorder, LObject : integer) : Integer
 Function AlIsValidEmail( const Value : AnsiString) : boolean
 Function AlLocalDateTimeToGMTDateTime( const aLocalDateTime : TDateTime) : TdateTime
 Function ALInc( var x : integer; Count : integer) : Integer
 function ALCopyStr(const aSourceString: AnsiString; aStart, aLength: Integer): AnsiString
 function ALGetStringFromFile(filename: AnsiString; const ShareMode: Word = fmShareDenyWrite):AnsiString;
 procedure ALSaveStringtoFile(Str: AnsiString; filename: AnsiString);
 Function ALIsInteger(const S: AnsiString): Boolean;
 function ALIsDecimal(const S: AnsiString): boolean;
 Function ALStringToWideString(const S: AnsiString; const aCodePage: Word): WideString;
 function AlWideStringToString(const WS: WideString; const aCodePage: Word): AnsiString;
 function  ALQuotedStr(const S: AnsiString; const Quote: AnsiChar = ''''): AnsiString;
 function  ALDequotedStr(const S: AnsiString; AQuote: AnsiChar): AnsiString;
 function  AlUTF8removeBOM(const S: AnsiString): AnsiString;
 Function ALRandomStr1(const aLength: Longint; const aCharset: Array of Char): AnsiString;
 Function ALRandomStr(const aLength: Longint): AnsiString;
 Function ALRandomStrU1(const aLength: Longint; const aCharset: Array of Char): String;
 Function ALRandomStrU(const aLength: Longint): String;
end;

procedure SIRegister_ALJSONDoc(CL: TPSPascalCompiler);
begin
 Procedure ALJSONToTStrings(const AJsonStr:AnsiString;aLst:TALStrings; const aNullStr:AnsiString;const
aTrueStr: AnsiString; const aFalseStr : AnsiString)
```

```
end;

procedure SIRegister_ALWindows(CL: TPSPascalCompiler);
begin
  _ALMEMORYSTATUSEX', 'record dwLength : DWORD; dwMemoryLoad : DWO'
   +'RD; ullTotalPhys : Int64; ullAvailPhys : Int64; ullTotalPageFile : Int64; '
   +'ullAvailPageFile : Int64; ullTotalVirtual : Int64; ullAvailVirtual : Int64'
   +'; ullAvailExtendedVirtual : Int64; end
  TALMemoryStatusEx', '_ALMEMORYSTATUSEX
 Function ALGlobalMemoryStatusEx( var lpBuffer : TALMEMORYSTATUSEX) : BOOL
 Function ALInterlockedExchange64( var Target : LONGlONG; Value : LONGLONG) : LONGLONG
 'INVALID_SET_FILE_POINTER','LongInt'( DWORD ( - 1 ));
 'QUOTA_LIMITS_HARDWS_MIN_DISABLE','LongWord').SetUInt( $2);
 'QUOTA_LIMITS_HARDWS_MIN_ENABLE','LongWord').SetUInt( $1);
 'QUOTA_LIMITS_HARDWS_MAX_DISABLE','LongWord').SetUInt( $8);
 'QUOTA_LIMITS_HARDWS_MAX_ENABLE','LongWord').SetUInt( $4);
end;

procedure SIRegister_IPCThrd(CL: TPSPascalCompiler);
begin
  SIRegister_THandledObject(CL);
  SIRegister_TEvent(CL);
  SIRegister_TMutex(CL);
  SIRegister_TSharedMem(CL);
 'TRACE_BUF_SIZE','LongInt'( 200 * 1024);
 'TRACE_BUFFER','String 'TRACE_BUFFER
 'TRACE_MUTEX','String 'TRACE_MUTEX
  //PTraceEntry', '^TTraceEntry // will not work
  SIRegister_TIPCTracer(CL);
 'MAX_CLIENTS','LongInt'( 6);
 'IPCTIMEOUT','LongInt'( 2000);
 'IPCBUFFER_NAME','String 'BUFFER_NAME
 'BUFFER_MUTEX_NAME','String 'BUFFER_MUTEX
 'MONITOR_EVENT_NAME','String 'MONITOR_EVENT
 'CLIENT_EVENT_NAME','String 'CLIENT_EVENT
 'CONNECT_EVENT_NAME','String 'CONNECT_EVENT
 'CLIENT_DIR_NAME','String 'CLIENT_DIRECTORY
 'CLIENT_DIR_MUTEX','String 'DIRECTORY_MUTEX
  FindClass('TOBJECT'),'EMonitorActive
  FindClass('TOBJECT'),'TIPCThread
  TEventKind', '( evMonitorAttach, evMonitorDetach, evMonitorSigna'
   +'l, evMonitorExit, evClientStart, evClientStop, evClientAttach, evClientDet'
   +'ach, evClientSwitch, evClientSignal, evClientExit )
  TClientFlag', '( cfError, cfMouseMove, cfMouseDown, cfResize, cfAttach )
  TClientFlags', 'set of TClientFlag
  //PEventData', '^TEventData // will not work
  TEventData', 'record X : SmallInt; Y : SmallInt; Flag : TClientF'
   +'lag; Flags : TClientFlags; end
  TConnectEvent', 'Procedure ( Sender : TIPCThread; Connecting : Boolean)
  TDirUpdateEvent', 'Procedure ( Sender : TIPCThread)
  TIPCNotifyEvent', 'Procedure ( Sender : TIPCThread; Data : TEventData)
  //PIPCEventInfo', '^TIPCEventInfo // will not work
  TIPCEventInfo','record FID:Integer;FKind:TEventKind;FData:TEventData;end
  SIRegister_TIPCEvent(CL);
  //PClientDirRecords', '^TClientDirRecords // will not work
  SIRegister_TClientDirectory(CL);
  TIPCState', '( stInActive, stDisconnected, stConnected )
  SIRegister_TIPCThread(CL);
  SIRegister_TIPCMonitor(CL);
  SIRegister_TIPCClient(CL);
 Function IsMonitorRunning( var Hndl : THandle) : Boolean
end;

(*----------------------------------------------------------------------------*)
procedure SIRegister_ALGSMComm(CL: TPSPascalCompiler);
begin
  SIRegister_TAlGSMComm(CL);
 Function AlGSMComm_BuildPDUMessage( aSMSCenter, aSMSAddress, aMessage : AnsiString) : AnsiString
 Procedure AlGSMComm_DecodePDUMessage(aPDUMessage:AnsiString;var aSMSCenter,aSMSAddress,
AMessage:AnsiString);
 Function AlGSMComm_UnicodeToGSM7BitDefaultAlphabet( aMessage : WideString) : AnsiString
 Function AlGSMComm_GSM7BitDefaultAlphabetToUnicode(aMess:AnsiString;const
UseGreekAlphabet:Bool):Widestring;
 function ALMatchesMask(const Filename, Mask: AnsiString): Boolean;
 end;

procedure SIRegister_ALHttpCommon(CL: TPSPascalCompiler);
begin
  TALHTTPPropertyChangeEvent','Procedure(sender:Tobject;const PropertyIndex:Integer;
  TALHTTPProtocolVersion', '( HTTPpv_1_0, HTTPpv_1_1 )
  TALHTTPMethod','(HTTPmt_Get,HTTPmt_Post,HTTPmt_Head,HTTPmt_Trace,HTTPmt_Put,HTTPmt_Delete);
  TInternetScheme', 'integer
  TALIPv6Binary', 'array[1..16] of Char;
 // TALIPv6Binary = array[1..16] of ansiChar;
 //   TInternetScheme = Integer;
  SIRegister_TALHTTPRequestHeader(CL);
  SIRegister_TALHTTPCookie(CL);
  SIRegister_TALHTTPCookieCollection(CL);
  SIRegister_TALHTTPResponseHeader(CL);
 Function ALHTTPDecode( const AStr : AnsiString) : AnsiString
 Procedure ALHTTPEncodeParamNameValues( ParamValues : TALStrings)
// Procedure ALExtractHTTPFields(Separators, WhiteSpace, Quotes:TSysCharSet; Content:PAnsiChar;Strings:TALStrings;StripQuotes:Boolean;
// Procedure ALExtractHeaderFields( Separators,WhiteSpace, Quotes : TSysCharSet; Content : PAnsiChar; Strings : TALStrings; Decode : Boolean; StripQuotes : Boolean)
// Procedure ALExtractHeaderFieldsWithQuoteEscaped(Separators,WhiteSpace, Quotes:TSysCharSet;Content:PAnsiChar;Strings : TALStrings; Decode : Boolean; StripQuotes : Boolean)
 Function AlRemoveShemeFromUrl( aUrl : AnsiString) : ansiString
 Function AlExtractShemeFromUrl( aUrl : AnsiString) : TInternetScheme
 Function AlExtractHostNameFromUrl( aUrl : AnsiString) : AnsiString
 Function AlExtractDomainNameFromUrl( aUrl : AnsiString) : AnsiString
 Function AlExtractUrlPathFromUrl( aUrl : AnsiString) : AnsiString
 Function AlInternetCrackUrl( aUrl : AnsiString; var SchemeName,HostName,UserName,Password,UrlPath, ExtraInfo : AnsiString; var PortNumber : integer) : Boolean;
 Function AlInternetCrackUrl1( aUrl : AnsiString; var SchemeName, HostName, UserName, Password,UrlPath, Anchor : AnsiString; Query : TALStrings; var PortNumber : integer) : Boolean;
 Function AlInternetCrackUrl2(var Url:AnsiString;var Anchor:AnsiString;Query:TALStrings):Boolean;
 Function AlRemoveAnchorFromUrl( aUrl : AnsiString; var aAnchor : AnsiString) : AnsiString;
 Function AlRemoveAnchorFromUrl1( aUrl : AnsiString) : AnsiString;
 Function AlCombineUrl( RelativeUrl, BaseUrl : AnsiString) : AnsiString;
 Function AlCombineUrl1(RelativeUrl, BaseUrl, Anchor : AnsiString; Query:TALStrings) : AnsiString;
 Function ALGmtDateTimeToRfc822Str( const aValue : TDateTime) : AnsiString
 Function ALDateTimeToRfc822Str( const aValue : TDateTime) : AnsiString
 Function ALTryRfc822StrToGMTDateTime( const S : AnsiString; out Value : TDateTime) : Boolean
 Function ALRfc822StrToGMTDateTime( const s : AnsiString) : TDateTime
 Function ALTryIPV4StrToNumeric( aIPv4Str : ansiString; var aIPv4Num : Cardinal) : Boolean
 Function ALIPV4StrToNumeric( aIPv4 : ansiString) : Cardinal
 Function ALNumericToIPv4Str( aIPv4 : Cardinal) : ansiString
 Function ALZeroIpV6 : TALIPv6Binary
 Function ALTryIPV6StrToBinary( aIPv6Str : ansiString; var aIPv6Bin : TALIPv6Binary) : Boolean
 Function ALIPV6StrTobinary( aIPv6 : ansiString) : TALIPv6Binary
 Function ALBinaryToIPv6Str( aIPv6 : TALIPv6Binary) : ansiString
 Function ALBinaryStrToIPv6Binary( aIPV6BinaryStr : ansiString) : TALIPv6Binary
end;

procedure SIRegister_ALFcnHTML(CL: TPSPascalCompiler);
begin
 Procedure ALUTF8ExtractHTMLText(HtmlCont:AnsiStr;LstExtractedResourceText:TALStrings;const DecodeHTMLText:Bool;
 Function ALUTF8ExtractHTMLText1(HtmlContent:AnsiString;const DecodeHTMLText:Boolean): AnsiString;
 Function ALXMLCDataElementEncode( Src : AnsiString) : AnsiString
 Function ALXMLTextElementEncode(Src : AnsiString; const useNumericReference : boolean) : AnsiString
 Function ALUTF8XMLTextElementDecode( const Src : AnsiString) : AnsiString
 Function ALUTF8HTMLEncode(const Src:AnsiStr;const EncodeASCIIHtmlEntities:Bool;const useNumRef:bool):AnsiString);
 Function ALUTF8HTMLDecode( const Src : AnsiString) : AnsiString
 Function ALJavascriptEncode( const Src : AnsiString; const useNumericReference : boolean) : AnsiString
 Function ALUTF8JavascriptDecode( const Src : AnsiString) : AnsiString
 Procedure ALHideHtmlUnwantedTagForHTMLHandleTagfunct(var HtmlContent:AnsiString; const DeleteBodyOfUnwantedTag : Boolean; const ReplaceUnwantedTagCharBy : AnsiChar)
 Procedure ALCompactHtmlTagParams( TagParams : TALStrings)
end;

procedure SIRegister_ALInternetMessageCommon(CL: TPSPascalCompiler);
begin
  SIRegister_TALEMailHeader(CL);
  SIRegister_TALNewsArticleHeader(CL);
 Function AlParseEmailAddress(FriendlyEmail:AnsiString;var RealName:AString;const decodeRealName:Bool):AnsiString;
 Function AlExtractEmailAddress( FriendlyEmail : AnsiString) : AnsiString
 Function ALMakeFriendlyEmailAddress( aRealName, aEmail : AnsiString) : AnsiString
 Function ALEncodeRealName4FriendlyEmailAddress( aRealName : AnsiString) : AnsiString
 Function AlGenerateInternetMessageID : AnsiString;
 Function AlGenerateInternetMessageID1( ahostname : AnsiString) : AnsiString;
 Function ALDecodeQuotedPrintableString( src : AnsiString) : AnsiString
 Function AlDecodeInternetMessageHeaderInUTF8(aHeaderStr:AnsiString;aDefaultCodePage:Integer):AnsiString;
end;

(*----------------------------------------------------------------------------*)
procedure SIRegister_ALFcnWinSock(CL: TPSPascalCompiler);
begin
 Function ALHostToIP( HostName : AnsiString; var Ip : AnsiString):Boolean
 Function ALIPAddrToName( IPAddr : AnsiString) : AnsiString
 Function ALgetLocalIPs : TALStrings
 Function ALgetLocalHostName : AnsiString
end;

procedure SIRegister_ALFcnCGI(CL: TPSPascalCompiler);
begin
 Procedure AlCGIInitDefaultServerVariablesFromWebRequest(WebRequest : TALWebRequest;ServerVariables:TALStrings);
 Procedure AlCGIInitDefaultServerVariablesFromWebRequest1(WebRequest: TALWebRequest; ServerVariables : TALStrings; ScriptName, ScriptFileName : AnsiString; Url : AnsiString);
 Procedure ALCGIInitDefaultServerVariables( ServerVariables : TALStrings);
 Procedure AlCGIInitDefaultServerVariables1(ServerVars:TALStrings;ScriptName, ScriptFileName:AnsiString;Url:AnsiStr;
 Procedure AlCGIInitServerVariablesFromWebRequest( WebRequest:TALWebRequest; ServerVariables : TALStrings; ScriptName, ScriptFileName : AnsiString; Url : AnsiString);
 Procedure AlCGIExec( InterpreterFilename : AnsiString; ServerVariables : TALStrings; RequestContentStream : Tstream; ResponseContentStream : Tstream; ResponseHeader : TALHTTPResponseHeader);
 Procedure AlCGIExec1(ScriptName,ScriptFileName, Url, X_REWRITE_URL, InterpreterFilename:AnsiString; WebRequest : TALIsapiRequest; overloadedCookies:AnsiString;overloadedQueryString:AnsiString;overloadedReferer: AnsiString;'
 +'overloadedRequestContentStream:Tstream;var ResponseContentStr:AnsiString;ResponseHeader:TALHTTPResponseHeader;
 Procedure AlCGIExec2(ScriptName,ScriptFileName,Url,X_REWRITE_URL, InterpreterFilename:AnsiString;WebRequest: TALIsapiRequest; var ResponseContentString : AnsiString; ResponseHeader : TALHTTPResponseHeader);
end;

procedure SIRegister_ALFcnExecute(CL: TPSPascalCompiler);
begin
  TStartupInfoA', 'TStartupInfo
 'SE_CREATE_TOKEN_NAME','String'( 'SeCreateTokenPrivilege
 SE_ASSIGNPRIMARYTOKEN_NAME','String 'SeAssignPrimaryTokenPrivilege
 SE_LOCK_MEMORY_NAME','String)( 'SeLockMemoryPrivilege
 SE_INCREASE_QUOTA_NAME','String 'SeIncreaseQuotaPrivilege
 SE_UNSOLICITED_INPUT_NAME','String 'SeUnsolicitedInputPrivilege
 SE_MACHINE_ACCOUNT_NAME','String 'SeMachineAccountPrivilege
 SE_TCB_NAME','String 'SeTcbPrivilege
 SE_SECURITY_NAME','String 'SeSecurityPrivilege
 SE_TAKE_OWNERSHIP_NAME','String 'SeTakeOwnershipPrivilege
 SE_LOAD_DRIVER_NAME','String 'SeLoadDriverPrivilege
 SE_SYSTEM_PROFILE_NAME','String 'SeSystemProfilePrivilege
 SE_SYSTEMTIME_NAME','String 'SeSystemtimePrivilege
 SE_PROF_SINGLE_PROCESS_NAME','String 'SeProfileSingleProcessPrivilege
 SE_INC_BASE_PRIORITY_NAME','String 'SeIncreaseBasePriorityPrivilege
 SE_CREATE_PAGEFILE_NAME','String 'SeCreatePagefilePrivilege
 SE_CREATE_PERMANENT_NAME','String 'SeCreatePermanentPrivilege
 SE_BACKUP_NAME','String 'SeBackupPrivilege
 SE_RESTORE_NAME','String 'SeRestorePrivilege
 SE_SHUTDOWN_NAME','String 'SeShutdownPrivilege
 SE_DEBUG_NAME','String 'SeDebugPrivilege
 SE_AUDIT_NAME','String 'SeAuditPrivilege
 SE_SYSTEM_ENVIRONMENT_NAME','String 'SeSystemEnvironmentPrivilege
 SE_CHANGE_NOTIFY_NAME','String 'SeChangeNotifyPrivilege
 SE_REMOTE_SHUTDOWN_NAME','String 'SeRemoteShutdownPrivilege
 SE_UNDOCK_NAME','String 'SeUndockPrivilege
 SE_SYNC_AGENT_NAME','String 'SeSyncAgentPrivilege
 SE_ENABLE_DELEGATION_NAME','String 'SeEnableDelegationPrivilege
 SE_MANAGE_VOLUME_NAME','String 'SeManageVolumePrivilege
 Function AlGetEnvironmentString : AnsiString
 Function ALWinExec32(const FileName,CurrentDir, Environment:AnsiString;InStream:Tstream;OutStream:TStream):Dword;
 Function ALWinExec321(const FileName:AnsiString; InputStream:Tstream;OutputStream:TStream):Dword;
 Function ALWinExecAndWait32( FileName : AnsiString; Visibility : integer) : DWORD
 Function ALWinExecAndWait32V2( FileName : AnsiString; Visibility : integer) : DWORD
 Function ALNTSetPrivilege( sPrivilege : AnsiString; bEnabled : Boolean) : Boolean
end;

procedure SIRegister_ALFcnFile(CL: TPSPascalCompiler);
begin
 Function AlEmptyDirectory(Directory:ansiString;SubDirectory:Bool;IgnoreFiles:array of AnsiString; const RemoveEmptySubDirectory : Boolean; const FileNameMask : ansiString; const MinFileAge : TdateTime):Boolean;
 Function AlEmptyDirectory1( Directory : ansiString; SubDirectory : Boolean; const RemoveEmptySubDirectory:Bool; const FileNameMask : ansiString; const MinFileAge : TdateTime) : Boolean;
 Function AlCopyDirectory( SrcDirectory, DestDirectory : ansiString; SubDirectory : Boolean; const FileNameMask : ansiString; const FailIfExists : Boolean) : Boolean
 Function ALGetModuleName : ansistring
 Function ALGetModuleFileNameWithoutExtension : ansistring
 Function ALGetModulePath : ansiString
 Function AlGetFileSize( const AFileName : ansistring) : int64
 Function AlGetFileVersion( const AFileName : ansistring) : ansiString
 Function ALGetFileCreationDateTime( const aFileName : Ansistring) : TDateTime
 Function ALGetFileLastWriteDateTime( const aFileName : Ansistring) : TDateTime
 Function ALGetFileLastAccessDateTime( const aFileName : Ansistring) : TDateTime
 Procedure ALSetFileCreationDateTime( const aFileName : Ansistring; const aCreationDateTime : TDateTime)
 Function ALIsDirectoryEmpty( const directory : ansiString) : boolean
 Function ALFileExists( const Path : ansiString) : boolean
 Function ALDirectoryExists( const Directory : Ansistring) : Boolean
 Function ALCreateDir( const Dir : Ansistring) : Boolean
 Function ALRemoveDir( const Dir : Ansistring) : Boolean
 Function ALDeleteFile( const FileName : Ansistring) : Boolean
 Function ALRenameFile( const OldName, NewName : ansistring) : Boolean
end;

procedure SIRegister_ALFcnMime(CL: TPSPascalCompiler);
begin
  NativeInt', 'Integer
  NativeUInt', 'Cardinal
 Function ALMimeBase64EncodeString( const S : AnsiString) : AnsiString
 Function ALMimeBase64EncodeStringNoCRLF( const S : AnsiString) : AnsiString
 Function ALMimeBase64DecodeString( const S : AnsiString) : AnsiString
 Function ALMimeBase64EncodedSize( const InputSize : NativeInt) : NativeInt
 Function ALMimeBase64EncodedSizeNoCRLF( const InputSize : NativeInt) : NativeInt
 Function ALMimeBase64DecodedSize( const InputSize : NativeInt) : NativeInt
 Procedure ALMimeBase64Encode( const InputBuffer : TByteDynArray; InputOffset : NativeInt; const InputByteCount : NativeInt; out OutputBuffer : TByteDynArray; OutputOffset : NativeInt)
 Procedure ALMimeBase64EncodeNoCRLF( const InputBuffer : TByteDynArray; InputOffset : NativeInt; const
InputByteCount : NativeInt; out OutputBuffer : TByteDynArray; OutputOffset : NativeInt)
 Procedure ALMimeBase64EncodeFullLines( const InputBuffer : TByteDynArray; InputOffset : NativeInt; const
InputByteCount : NativeInt; out OutputBuffer : TByteDynArray; OutputOffset : NativeInt)
 Function ALMimeBase64Decode( const InputBuffer : TByteDynArray; InputOffset : NativeInt; const
InputByteCount : NativeInt; out OutputBuffer : TByteDynArray; OutputOffset : NativeInt) : NativeInt;
 Function ALMimeBase64DecodePartial( const InputBuffer : TByteDynArray; InputOffset : NativeInt; const
InputByteCount : NativeInt; out OutputBuffer : TByteDynArray; OutputOffset : NativeInt;'
 + 'var ByteBuffer : Cardinal; var ByteBufferSpace : Cardinal) : NativeInt;
 Function ALMimeBase64DecodePartialEnd( out OutputBuffer : TByteDynArray; OutputOffset : NativeInt; const
ByteBuffer : Cardinal; const ByteBufferSpace : Cardinal) : NativeInt;
 Procedure ALMimeBase64Encode(const InputBuf:TByteDynArray;const InputByteCnt:NatInt;out
OutputBuf:TByteDynArray);
 Procedure ALMimeBase64EncodeNoCRLF(const InputBuffer:TByteDynArray; const InputByteCount:NativeInt;out
OutputBuffer:TByteDynArray);
 Procedure ALMimeBase64EncodeFullLines(const InputBuffer:TByteDynArray;const InputByteCount:NativeInt;out
OutputBuffer:TByteDynArray);
 Function ALMimeBase64Decode1(const InputBuffer:TByteDynArray;const InputByteCount:NativeInt;out
OutputBuffer:TByteDynArray):NativeInt;
 Function ALMimeBase64DecodePartial1(const InputBuffer:TByteDynArray;const InputByteCount:NativeInt;out
OutputBuffer: TByteDynArray; var ByteBuffer : Cardinal; var ByteBufferSpace : Cardinal) : NativeInt;
 Function ALMimeBase64DecodePartialEnd1(out OutputBuffer:TByteDynArray;const ByteBuffer:Cardinal;const
ByteBufferSpace:Cardinal):NativeInt;
 Procedure ALMimeBase64EncodeFile( const InputFileName, OutputFileName : TFileName)
 Procedure ALMimeBase64EncodeFileNoCRLF( const InputFileName, OutputFileName : TFileName)
 Procedure ALMimeBase64DecodeFile( const InputFileName, OutputFileName : TFileName)
 Procedure ALMimeBase64EncodeStream( const InputStream : TStream; const OutputStream : TStream)
 Procedure ALMimeBase64EncodeStreamNoCRLF( const InputStream : TStream; const OutputStream : TStream)
 Procedure ALMimeBase64DecodeStream( const InputStream : TStream; const OutputStream : TStream)
 'cALMimeBase64_ENCODED_LINE_BREAK','LongInt'( 76);
 'cALMimeBase64_DECODED_LINE_BREAK','LongInt'( cALMimeBase64_ENCODED_LINE_BREAK div 4 * 3);
 'cALMimeBase64_BUFFER_SIZE','LongInt'( cALMimeBase64_DECODED_LINE_BREAK * 3 * 4 * 4);
 Procedure ALFillMimeContentTypeByExtList( AMIMEList : TALStrings)
 Procedure ALFillExtByMimeContentTypeList( AMIMEList : TALStrings)
 Function ALGetDefaultFileExtFromMimeContentType( aContentType : AnsiString) : AnsiString
 Function ALGetDefaultMIMEContentTypeFromExt( aExt : AnsiString) : AnsiString
end;

procedure SIRegister_ALXmlDoc(CL: TPSPascalCompiler);
begin
 'cALXMLNodeMaxListSize','LongInt'( Maxint div 16);
  FindClass('TOBJECT'),'TALXMLNode
  FindClass('TOBJECT'),'TALXMLNodeList
  FindClass('TOBJECT'),'TALXMLDocument
  TAlXMLParseProcessingInstructionEvent','Procedure (Sender:TObject; const Target,Data:AnsiString)
  TAlXMLParseTextEvent', 'Procedure ( Sender : TObject; const str: AnsiString)
  TAlXMLParseStartElementEvent', 'Procedure ( Sender : TObject; co'
   +'nst Name : AnsiString; const Attributes : TALStrings)
  TAlXMLParseEndElementEvent', 'Procedure ( Sender : TObject; const Name : AnsiString)
  TALXmlNodeType', '( ntReserved, ntElement, ntAttribute, ntText, '
   +'ntCData, ntEntityRef, ntEntity, ntProcessingInstr, ntComment, ntDocument, '
   +'ntDocType, ntDocFragment, ntNotation )
  TALXMLDocOption', '( doNodeAutoCreate, doNodeAutoIndent )
  TALXMLDocOptions', 'set of TALXMLDocOption
  TALXMLParseOption', '( poPreserveWhiteSpace, poIgnoreXMLReferences )
  TALXMLParseOptions', 'set of TALXMLParseOption
  TALXMLPrologItem', '( xpVersion, xpEncoding, xpStandalone )
  PALPointerXMLNodeList', '^TALPointerXMLNodeList // will not work
  SIRegister_EALXMLDocError(CL);
  SIRegister_TALXMLNodeList(CL);
  SIRegister_TALXMLNode(CL);
  SIRegister_TALXmlElementNode(CL);
  SIRegister_TALXmlAttributeNode(CL);
  SIRegister_TALXmlTextNode(CL);
  SIRegister_TALXmlDocumentNode(CL);
  SIRegister_TALXmlCommentNode(CL);
  SIRegister_TALXmlProcessingInstrNode(CL);
  SIRegister_TALXmlCDataNode(CL);
  SIRegister_TALXmlEntityRefNode(CL);
  SIRegister_TALXmlEntityNode(CL);
  SIRegister_TALXmlDocTypeNode(CL);
  SIRegister_TALXmlDocFragmentNode(CL);
  SIRegister_TALXmlNotationNode(CL);
  SIRegister_TALXMLDocument(CL);
 cAlXMLUTF8EncodingStr','String 'UTF-8
 cALXmlUTF8HeaderStr','String'<?xmlversion="1.0"encoding="UTF-8"standalone="yes"?>'+#13#10);
 CALNSDelim','String ':
 CALXML','String 'xml
 CALVersion','String 'version
 CALEncoding','String 'encoding
 CALStandalone','String 'standalone
 CALDefaultNodeIndent','String '
 CALXmlDocument','String 'DOCUMENT
 Function ALCreateEmptyXMLDocument( const Rootname : AnsiString) : TalXMLDocument
 Procedure ALClearXMLDocument(const rootname:AnsiString;xmldoc:TalXMLDocument;const
EncodingStr:AnsiString);
 Function ALFindXmlNodeByChildNodeValue(xmlrec:TalxmlNode;const ChildNodeName,
ChildNodeValue:AnsiString;const Recurse: Boolean) : TalxmlNode
 Function ALFindXmlNodeByNameAndChildNodeValue( xmlrec : TalxmlNode; const NodeName : ansiString; const
ChildNodeName, ChildNodeValue : AnsiString; const Recurse : Boolean) : TalxmlNode
```

```
 Function ALFindXmlNodeByAttribute(xmlrec:TalxmlNode;const AttributeName,AttributeValue:AnsiString;const
Recurse: Boolean):TalxmlNode
 Function ALFindXmlNodeByNameAndAttribute( xmlrec : TalxmlNode; const NodeName : ansiString; const
AttributeName, AttributeValue : AnsiString; const Recurse : Boolean) : TalxmlNode
 Function ALExtractAttrValue( const AttrName, AttrLine : AnsiString; const Default : AnsiString) :
AnsiString
end;

procedure SIRegister_TeCanvas(CL: TPSPascalCompiler);
//based on TEEProc, TeCanvas, TEEngine, TChart
begin
 'TeePiStep','Double').setExtended( Pi / 180.0);
 'TeeDefaultPerspective','LongInt'( 100);
 'TeeMinAngle','LongInt'( 270);
 'teeclMoneyGreen','LongWord').SetUInt( TColor ( $C0DCC0 ));
 'teeclSkyBlue','LongWord').SetUInt( TColor ( $F0CAA6 ));
 'teeclCream','LongWord( TColor ( $F0FBFF ));
 'teeclMedGray','LongWord').SetUInt( TColor ( $A4A0A0 ));
 'teeclMoneyGreen','LongWord').SetUInt( TColor ( $C0DCC0 ));
 'teeclSkyBlue','LongWord').SetUInt( TColor ( $F0CAA6 ));
 'teeclCream','LongWord').SetUInt( TColor ( $F0FBFF ));
 'teeclMedGray','LongWord').SetUInt( TColor ( $A4A0A0 ));
 'TA_LEFT','LongInt'( 0);
 'TA_RIGHT','LongInt'( 2);
 'TA_CENTER','LongInt'( 6);
 'TA_TOP','LongInt'( 0);
 'TA_BOTTOM','LongInt'( 8);
 'teePATCOPY','LongInt'( 0);
 'NumCirclePoints','LongInt'( 64);
 'teeDEFAULT_CHARSET','LongInt'( 1);
 'teeANTIALIASED_QUALITY','LongInt'( 4);
 'TA_LEFT','LongInt'( 0);
 'bs_Solid','LongInt'( 0);
 'teepf24Bit','LongInt'( 0);
 'teepfDevice','LongInt'( 1);
 'CM_MOUSELEAVE','LongInt'( 10000);
 'CM_SYSCOLORCHANGE','LongInt'( 10001);
 'DC_BRUSH','LongInt'( 18);
 'DC_PEN','LongInt'( 19);
  teeCOLORREF', 'LongWord
  TLogBrush', 'record lbStyle : Integer; lbColor : TColor; lbHatch: Integer; end
  //TNotifyEvent', 'Procedure ( Sender : TObject)
  SIRegister_TFilterRegion(CL);
  SIRegister_IFormCreator(CL);
  SIRegister_TTeeFilter(CL);
  //TFilterClass', 'class of TTeeFilter
  SIRegister_TFilterItems(CL);
  SIRegister_TConvolveFilter(CL);
  SIRegister_TBlurFilter(CL);
  SIRegister_TTeePicture(CL);
  TPenEndStyle', '( esRound, esSquare, esFlat )
  SIRegister_TChartPen(CL);
  SIRegister_TChartHiddenPen(CL);
  SIRegister_TDottedGrayPen(CL);
  SIRegister_TDarkGrayPen(CL);
  SIRegister_TWhitePen(CL);
  SIRegister_TChartBrush(CL);
  TTeeView3DScrolled', 'Procedure ( IsHoriz : Boolean)
  TTeeView3DChangedZoom', 'Procedure ( NewZoom : Integer)
  SIRegister_TView3DOptions(CL);
  FindClass('TOBJECT'),'TTeeCanvas
  TTeeTransparency', 'Integer
  SIRegister_TTeeBlend(CL);
  FindClass('TOBJECT'),'TCanvas3D
  SIRegister_TTeeShadow(CL);
  teeTGradientDirection', '( gdTopBottom, gdBottomTop, gdLeftRight, g'
   +'dRightLeft, gdFromCenter, gdFromTopLeft, gdFromBottomLeft, gdRadial,gdDiagonalUp,gdDiagonalDown )
  FindClass('TOBJECT'),'TSubGradient
  SIRegister_TCustomTeeGradient(CL);
  SIRegister_TSubGradient(CL);
  SIRegister_TTeeGradient(CL);
  SIRegister_TTeeFontGradient(CL);
  SIRegister_TTeeFont(CL);
  TCanvasBackMode', '( cbmNone, cbmTransparent, cbmOpaque )
  TCanvasTextAlign', 'Integer
  TTeeCanvasHandle', 'HDC
  SIRegister_TTeeCanvas(CL);
  TPoint3DFloat', 'record X : Double; Y : Double; Z : Double; end
  SIRegister_TFloatXYZ(CL);
  TPoint3D', 'record x : integer; y : integer; z : Integer; end
  TRGB', 'record blue: byte; green: byte; red: byte; end
  {TRGB=packed record
    Blue  : Byte;
    Green : Byte;
    Red   : Byte;
    //$IFDEF CLX  //Alpha : Byte; // Linux  end;}

TTeeCanvasCalcPoints', 'Function ( x, z : Integer; var P0, P1 : '
   +'TPoint3D; var Color0, Color1 : TColor) : Boolean
```

```
TTeeCanvasSurfaceStyle', '( tcsSolid, tcsWire, tcsDot )
TCanvas3DPlane', '( cpX, cpY, cpZ )
  //IInterface', 'IUnknown
  SIRegister_TCanvas3D(CL);
  SIRegister_TTeeCanvas3D(CL);
  TTrianglePoints', 'Array[0..2] of TPoint;
  TFourPoints', 'Array[0..3] of TPoint;
 Function ApplyDark( Color : TColor; HowMuch : Byte) : TColor
 Function ApplyBright( Color : TColor; HowMuch : Byte) : TColor
 Function Point3D( const x, y, z : Integer) : TPoint3D
 Procedure SwapDouble( var a, b : Double)
 Procedure SwapInteger( var a, b : Integer)
 Procedure RectSize( const R : TRect; var RectWidth, RectHeight : Integer)
 Procedure teeRectCenter( const R : TRect; var X, Y : Integer)
 Function RectFromPolygon( const Points : array of TPoint; NumPoints : Integer): TRect
 Function RectFromTriangle( const Points : TTrianglePoints) : TRect
 Function RectangleInRectangle( const Small, Big : TRect) : Boolean
 Procedure ClipCanvas( ACanvas : TCanvas; const Rect : TRect)
 Procedure UnClipCanvas( ACanvas : TCanvas)
 Procedure ClipEllipse( ACanvas : TTeeCanvas; const Rect : TRect)
 Procedure ClipRoundRectangle(ACanvas:TTeeCanvas;const Rect : TRect; RoundSize : Integer)
 Procedure ClipPolygon(ACanvas:TTeeCanvas;const Points:array of TPoint;NumPoints:Integer)
 'TeeCharForHeight','String 'W
 'DarkerColorQuantity','Byte').SetUInt( 128);
 'DarkColorQuantity','Byte').SetUInt( 64);
  TButtonGetColorProc', 'Function  : TColor
  SIRegister_TTeeButton(CL);
  SIRegister_TButtonColor(CL);
  SIRegister_TComboFlat(CL);
 Procedure TeeSetTeePen(FPen:TPen; APen : TChartPen; AColor : TColor; Handle:TTeeCanvasHandle)
 Function TeePoint( const aX, aY : Integer) : TPoint
 Function TEEPointInRect( const Rect : TRect; const P : TPoint) : Boolean;
 Function PointInRect1( const Rect : TRect; x, y : Integer) : Boolean;
 Function TeeRect( Left, Top, Right, Bottom : Integer) : TRect
 Function OrientRectangle( const R : TRect) : TRect
 Procedure TeeSetBitmapSize( Bitmap : TBitmap; Width, Height : Integer)
 Function PolygonBounds( const P : array of TPoint) : TRect
 Function PolygonInPolygon( const A, B : array of TPoint) : Boolean
 Function RGBValue( const Color : TColor) : TRGB
 Function EditColor( AOwner : TComponent; AColor : TColor) : TColor
 Function EditColorDialog( AOwner : TComponent; var AColor : TColor) : Boolean
 Function PointAtDistance( AFrom, ATo : TPoint; ADist : Integer) : TPoint
 Function TeeCull( const P : TFourPoints) : Boolean;
 Function TeeCull1( const P0, P1, P2 : TPoint) : Boolean;
  TSmoothStretchOption', '( ssBestQuality, ssBestPerformance )
 Procedure SmoothStretch( Src, Dst : TBitmap);
 Procedure SmoothStretch1( Src, Dst : TBitmap; Option : TSmoothStretchOption);
 Function TeeDistance( const x, y : Double) : Double
 Function TeeLoadLibrary( const FileName : String) : HInst
 Procedure TeeFreeLibrary( hLibModule : HMODULE)
 Procedure TeeBlendBitmaps( const Percent : Double; ABitmap, BBitmap : TBitmap; BOrigin : TPoint)
 //Procedure TeeCalcLines( var Lines : TRGBArray; Bitmap : TBitmap)
 Procedure TeeShadowSmooth(Bitmap, Back : TBitmap; Left, Top, Width, Height, horz, vert : Integer;
Smoothness : Double; FullDraw : Boolean; ACanvas : TCanvas3D; Clip : Boolean)
  SIRegister_ICanvasHyperlinks(CL);
  SIRegister_ICanvasToolTips(CL);
 Function Supports( const Instance : IInterface; const IID : TGUID) : Boolean
end;

procedure SIRegister_ovcmisc(CL: TPSPascalCompiler);
begin
  TOvcHdc', 'Integer
  TOvcHWND', 'Cardinal
  TOvcHdc', 'HDC
  TOvcHWND', 'HWND
 Function LoadBaseBitmap( lpBitmapName : PChar) : HBITMAP
 Function LoadBaseCursor( lpCursorName : PChar) : HCURSOR
 Function ovCompStruct( const S1, S2, Size : Cardinal) : Integer
 Function DefaultEpoch : Integer
 Function DrawButtonFrame(Canvas:TCanvas;const Client:TRect;IsDown,IsFlat:Bool;Style:TButtonStyle):TRect;
 Procedure FixRealPrim( P : PChar; DC : Char)
 Function GetDisplayString( Canvas : TCanvas; const S : string; MinChars, MaxWidth : Integer) : string
 Function GetLeftButton : Byte
 Function GetNextDlgItem( Ctrl : TOvcHWnd) : hWnd
 Procedure GetRGB( Clr : TColor; var IR, IG, IB : Byte)
 Function GetShiftFlags : Byte
 Function ovCreateRotatedFont( F : TFont; Angle : Integer) : hFont
 Function GetTopTextMargin(Font:TFont;BorderStyle:TBorderStyle; Height:Integer;Ctl3D:Boolean): Integer
 Function ovExtractWord( N : Integer; const S : string; WordDelims : TCharSet) : string
 Function ovIsForegroundTask : Boolean
 Function ovTrimLeft( const S : string) : string
 Function ovTrimRight( const S : string) : string
 Function ovQuotedStr( const S : string) : string
 Function ovWordCount( const S : string; const WordDelims : TCharSet) : Integer
 Function ovWordPosition(const N:Integer;const S: string;const WordDelims : TCharSet) : Integer
 Function PtrDiff( const P1, P2 : PChar) : Word
 Procedure PtrInc( var P, Delta : Word)
 Procedure PtrDec( var P, Delta : Word)
 Procedure FixTextBuffer( InBuf, OutBuf : PChar; OutSize : Integer)
 Procedure TransStretchBlt( DstDC : TOvcHdc; DstX, DstY, DstW, DstH : Integer; SrcDC : TOvcHdc; SrcX, SrcY,
 SrcW, SrcH : Integer; MaskDC : TOvcHdc; MaskX, MaskY : Integer)
 Function ovMinI( X, Y : Integer) : Integer
 Function ovMaxI( X, Y : Integer) : Integer
 Function ovMinL( X, Y : LongInt) : LongInt
 Function ovMaxL( X, Y : LongInt) : LongInt
 Function GenerateComponentName( PF : TWinControl; const Root : string) : string
 Function PartialCompare( const S1, S2 : string) : Boolean
 Function PathEllipsis( const S : string; MaxWidth : Integer) : string
 Function ovCreateDisabledBitmap( FOriginal : TBitmap; OutlineColor : TColor) : TBitmap
 Procedure ovCopyParentImage( Control : TControl; Dest : TCanvas)
 Procedure ovDrawTransparentBitmap( Dest : TCanvas; X, Y, W, H : Integer; Rect : TRect; Bitmap : TBitmap;
TransparentColor : TColor)
 Procedure DrawTransparentBitmapPrim(DC:TOvcHdc;Bitmap:HBitmap;xStart,yStart,Width,Height:Int;Rect:TRect;
TransparentColor : TColorRef)
 Function ovWidthOf( const R : TRect) : Integer
 Function ovHeightOf( const R : TRect) : Integer
 Procedure ovDebugOutput( const S : string)
 Function GetArrowWidth( Width, Height : Integer) : Integer
 Procedure StripCharSeq( CharSeq : string; var Str : string)
 Procedure StripCharFromEnd( aChr : Char; var Str : string)
 Procedure StripCharFromFront( aChr : Char; var Str : string)
 Function SystemParametersInfo( uiAction, uiParam : UINT; pvParam : UINT; fWinIni : UINT) : BOOL
 Function SystemParametersInfoNCM(uiAction,uiParam:UINT;pvParam:TNonClientMetrics;fWinIni:UINT):BOOL;
 Function SystemParametersInfoA( uiAction, uiParam : UINT; pvParam : UINT; fWinIni : UINT) : BOOL
 Function CreateEllipticRgn( p1, p2, p3, p4 : Integer) : HRGN
 Function CreateEllipticRgnIndirect( const p1 : TRect) : HRGN
 Function CreateFontIndirect( const p1 : TLogFont) : HFONT
 Function CreateMetaFile( p1 : PChar) : HDC
 Function DescribePixelFormat(DC: HDC;p2:Int;p3:UINT;var p4:TPixelFormatDescriptor): BOOL
 Function DrawText(hDC:HDC;lpString:PChar;nCount:Integer; var lpRect : TRect;uFormat:UINT):Integer
 Function DrawTextS(hDC:HDC;lpString:string;nCount:Integer; var lpRect:TRect;uFormat:UINT):Integer
 Function SetMapperFlags( DC : HDC; Flag : DWORD) : DWORD
 Function SetGraphicsMode( hdc : HDC; iMode : Integer) : Integer
 Function SetMapMode( DC : HDC; p2 : Integer) : Integer
 Function SetMetaFileBitsEx( Size : UINT; const Data : PChar) : HMETAFILE
 //Function SetPaletteEntries(Palette:HPALETTE;StartIndex,NumEntries:UINT;var PaletteEntries):UINT
 Function SetPixel( DC : HDC; X, Y : Integer; Color : COLORREF) : COLORREF
 Function SetPixelV( DC : HDC; X, Y : Integer; Color : COLORREF) : BOOL
 //Function SetPixelFormat( DC : HDC; PixelFormat : Integer; FormatDef : PPixelFormatDescriptor) : BOOL
 Function SetPolyFillMode( DC : HDC; PolyFillMode : Integer) : Integer
 Function StretchBlt(DestDC:HDC;X,Y,Width,Height:Int;SrcDC:HDC;XSrc,YSrc,SrcWidth,
SrcHeight:Int;Rop:DWORD):BOOL
 Function SetRectRgn( Rgn : HRgn; X1, Y1, X2, Y2 : Integer) : BOOL
 Function StretchDIBits(DC : HDC; DestX,DestY,DestWidth,DestHeight,SrcX,SrcY,SrcWidth,
SrcHeight:Int;Bits:int; var BitsInfo : TBitmapInfo; Usage : UINT; Rop : DWORD) : Integer
 Function SetROP2( DC : HDC; p2 : Integer) : Integer
 Function SetStretchBltMode( DC : HDC; StretchMode : Integer) : Integer
 Function SetSystemPaletteUse( DC : HDC; p2 : UINT) : UINT
 Function SetTextCharacterExtra( DC : HDC; CharExtra : Integer) : Integer
 Function SetTextColor( DC : HDC; Color : COLORREF) : COLORREF
 Function SetTextAlign( DC : HDC; Flags : UINT) : UINT
 Function SetTextJustification( DC : HDC; BreakExtra, BreakCount : Integer) : Integer
 Function UpdateColors( DC : HDC) : BOOL
 Function GetViewportExtEx( DC : HDC; var Size : TSize) : BOOL
 Function GetViewportOrgEx( DC : HDC; var Point : TPoint) : BOOL
 Function GetWindowExtEx( DC : HDC; var Size : TSize) : BOOL
 Function GetWindowOrgEx( DC : HDC; var Point : TPoint) : BOOL
 Function IntersectClipRect( DC : HDC; X1, Y1, X2, Y2 : Integer) : Integer
 Function InvertRgn( DC : HDC; p2 : HRGN) : BOOL
 Function MaskBlt( DestDC : HDC; XDest, YDest, Width, Height : Integer; SrcDC : HDC; XScr, YScr : Integer;
Mask : HBITMAP; xMask, yMask : Integer; Rop : DWORD) : BOOL
 Function PlgBlt(DestDC:HDC;const PtsArray,SrcDC:HDC;XSrc,YSrc,Widt,Heigh:Int;Mask:HBITMAP;xMask,
yMask:Int):BOOL;
 Function OffsetClipRgn( DC : HDC; XOffset, YOffset : Integer) : Integer
 Function OffsetRgn( RGN : HRGN; XOffset, YOffset : Integer) : Integer
 Function PatBlt( DC : HDC; X, Y, Width, Height : Integer; Rop : DWORD) : BOOL
 Function Pie( DC : HDC; X1, Y1, X2, Y2, X3, Y3, X4, Y4 : Integer) : BOOL
 Function PlayMetaFile( DC : HDC; MF : HMETAFILE) : BOOL
 Function PaintRgn( DC : HDC; RGN : HRGN) : BOOL
 Function PtInRegion( RGN : HRGN; X, Y : Integer) : BOOL
 Function PtVisible( DC : HDC; X, Y : Integer) : BOOL
 Function RectInRegion( RGN : HRGN; const Rect : TRect) : BOOL
 Function RectVisible( DC : HDC; const Rect : TRect) : BOOL
 Function Rectangle( DC : HDC; X1, Y1, X2, Y2 : Integer) : BOOL
 Function RestoreDC( DC : HDC; SavedDC : Integer) : BOOL
end;

procedure SIRegister_ovcfiler(CL: TPSPascalCompiler);
begin
  SIRegister_TOvcAbstractStore(CL);
  //PExPropInfo', '^TExPropInfo // will not work
//  TExPropInfo', 'record PI : TPropInfo; AObject : TObject; end
  SIRegister_TOvcPropertyList(CL);
  SIRegister_TOvcDataFiler(CL);
 Procedure UpdateStoredList( AForm : TWinControl; AStoredList : TStrings; FromForm : Boolean)
 Procedure UpdateStoredList1( AForm : TCustomForm; AStoredList : TStrings; FromForm : Boolean)
 Function CreateStoredItem( const CompName, PropName : string) : string
 Function ParseStoredItem( const Item : string; var CompName, PropName : string) : Boolean
 //Function GetPropType( PropInfo : PExPropInfo) : PTypeInfo
end;

procedure SIRegister_ovccoco(CL: TPSPascalCompiler);
begin
 'ovsetsize','LongInt'( 16);
 'etSyntax','LongInt'( 0);
 'etSymantic','LongInt'( 1);
 'chCR','Char #13);
 'chLF','Char #10);
 'chLineSeparator',' chCR);
  SIRegister_TCocoError(CL);
  SIRegister_TCommentItem(CL);
  SIRegister_TCommentList(CL);
  SIRegister_TSymbolPosition(CL);
TGenListType', '( glNever, glAlways, glOnError )
TovBitSet', 'set of Integer
  //PStartTable', '^TStartTable // will not work
'TovCharSet', 'set of AnsiChar
TAfterGenListEvent', 'Procedure ( Sender : TObject; var PrintErrorCount : boolean)
TCommentEvent', 'Procedure ( Sender : TObject; CommentList : TCommentList)
TCustomErrorEvent', 'Function(Sender:TObject;const ErrorCode: longint;const Data:string): string
TovErrorEvent', 'Procedure ( Sender : TObject; Error : TCocoError)
TovErrorProc', 'Procedure ( ErrorCode : integer; Symbol : TSymbolP'
   +'osition; const Data : string; ErrorType : integer)
TFailureEvent', 'Procedure ( Sender : TObject; NumErrors : integer)
TGetCH', 'Function ( pos : longint) : char
TStatusUpdateProc', 'Procedure ( Sender : TObject; Status: string; LineNum:integer)
  SIRegister_TCocoRScanner(CL);
  SIRegister_TCocoRGrammar(CL);
 '_EF','Char #0);
 '_TAB','Char').SetString( #09);
 '_CR','Char').SetString( #13);
 '_LF','Char').SetString( #10);
 '_EL','').SetString( _CR);
 '_EOF','Char').SetString( #26);
 'LineEnds','TCharSet'(ord(_CR) or ord(_LF) or ord(_EF));
 'minErrDist','LongInt'( 2);
 Function ovPadL( S : string; ch : char; L : integer) : string
end;

  TFormatSettings = record
    CurrencyFormat: Byte;
    NegCurrFormat: Byte;
    ThousandSeparator: Char;
    DecimalSeparator: Char;
    CurrencyDecimals: Byte;
    DateSeparator: Char;
    TimeSeparator: Char;
    ListSeparator: Char;
    CurrencyString: string;
    ShortDateFormat: string;
    LongDateFormat: string;
    TimeAMString: string;
    TimePMString: string;
    ShortTimeFormat: string;
    LongTimeFormat: string;
    ShortMonthNames: array[1..12] of string;
    LongMonthNames: array[1..12] of string;
    ShortDayNames: array[1..7] of string;
    LongDayNames: array[1..7] of string;
    TwoDigitYearCenturyWindow: Word;
  end;

procedure SIRegister_OvcFormatSettings(CL: TPSPascalCompiler);
begin
 Function ovFormatSettings : TFormatSettings
end;

procedure SIRegister_ovcstr(CL: TPSPascalCompiler);
begin
  TOvcCharSet', 'set of Char
  ovBTable', 'array[0..255] of Byte
  //BTable = array[0..{$IFDEF UNICODE}{$IFDEF
HUGE_UNICODE_BMTABLE}$FFFF{$ELSE}$FF{$ENDIF}{$ELSE}$FF{$ENDIF}] of Byte;
 Function BinaryBPChar( Dest : PChar; B : Byte) : PChar
 Function BinaryLPChar( Dest : PChar; L : LongInt) : PChar
 Function BinaryWPChar( Dest : PChar; W : Word) : PChar
 Procedure BMMakeTable( MatchString : PChar; var BT : ovBTable)
 Function BMSearch(var Buffer,BufLength:Cardinal;var BT:ovBTable; MatchString:PChar;var Pos:Cardinal):Bool;
 Function BMSearchUC(var Buffer,BufLength:Card; var BT:ovBTable;MatchString:PChar; var Pos: Card):Boolean
 Function CharStrPChar( Dest : PChar; C : Char; Len : Cardinal) : PChar
 Function DetabPChar( Dest : PChar; Src : PChar; TabSize : Byte) : PChar
 Function HexBPChar( Dest : PChar; B : Byte) : PChar
 Function HexLPChar( Dest : PChar; L : LongInt) : PChar
 Function HexPtrPChar( Dest : PChar; P : TObject) : PChar
 Function HexWPChar( Dest : PChar; W : Word) : PChar
 Function LoCaseChar( C : Char) : Char
 Function OctalLPChar( Dest : PChar; L : LongInt) : PChar
 Function StrChDeletePrim( P : PChar; Pos : Cardinal) : PChar
 Function StrChInsertPrim( Dest : PChar; C : Char; Pos: Cardinal) : PChar
 Function StrChPos( P : PChar; C : Char; var Pos : Cardinal) : Boolean
 Procedure StrInsertChars( Dest : PChar; Ch : Char; Pos, Count : Word)
 Function StrStCopy( Dest, S : PChar; Pos, Count : Cardinal) : PChar
 Function StrStDeletePrim( P : PChar; Pos, Count : Cardinal) : PChar
 Function StrStInsert( Dest, S1, S2 : PChar; Pos : Cardinal) : PChar
 Function StrStInsertPrim( Dest, S : PChar; Pos : Cardinal) : PChar
 Function StrStPos( P, S : PChar; var Pos : Cardinal) : Boolean
 Function StrToLongPChar( S : PChar; var I : LongInt) : Boolean
 Procedure TrimAllSpacesPChar( P : PChar)
 Function TrimEmbeddedZeros( const S : string) : string
 Procedure TrimEmbeddedZerosPChar( P : PChar)
 Function TrimTrailPrimPChar( S : PChar) : PChar
 Function TrimTrailPChar( Dest, S : PChar) : PChar
 Function TrimTrailingZeros( const S : string) : string
 Procedure TrimTrailingZerosPChar( P : PChar)
 Function UpCaseChar( C : Char) : Char
 Function ovcCharInSet( C : Char; const CharSet : TOvcCharSet) : Boolean
 Function ovc32StringIsCurrentCodePage( const S : WideString) : Boolean;
 //Function ovc32StringIsCurrentCodePage1( const S:PWideChar; CP : Cardinal) : Boolean;
end;

procedure SIRegister_AfUtils(CL: TPSPascalCompiler);
begin
  //PRaiseFrame', '^TRaiseFrame // will not work
  TRaiseFrame', 'record NextRaise : PRaiseFrame; ExceptAddr : ___Poin'
   +'ter; ExceptObject : TObject; ExceptionRecord : PExceptionRecord; end
 Procedure SafeCloseHandle( var Handle : THandle)
 Procedure ExchangeInteger( X1, X2 : Integer)
 Procedure FillInteger( const Buffer, Size, Value : Integer)
 Function LongMulDiv( Mult1, Mult2, Div1 : Longint) : Longint
 Function afCompareMem( P1, P2 : TObject; Length : Integer) : Boolean

FILENAME_ADVAPI32     = 'ADVAPI32.DLL';
    function AbortSystemShutdown; external advapi32 name 'AbortSystemShutdownW';
function AbortSystemShutdown(lpMachineName: PKOLChar): BOOL; stdcall;
    function AccessCheckAndAuditAlarm(SubsystemName: PKOLChar;
      HandleId: Pointer; ObjectTypeName, ObjectName: PKOLChar;
      SecurityDescriptor: PSecurityDescriptor; DesiredAccess: DWORD;
      const GenericMapping: TGenericMapping;  ObjectCreation: BOOL;
      var GrantedAccess: DWORD; var AccessStatus, pfGenerateOnClose: BOOL): BOOL; stdcall;
    function AccessCheckByTypeAndAuditAlarm(SubsystemName: PKOLChar;
      HandleId: Pointer; ObjectTypeName, ObjectName: PKOLChar;
      SecurityDescriptor: PSecurityDescriptor; PrincipalSelfSid: PSID; DesiredAccess: DWORD;
      AuditType: AUDIT_EVENT_TYPE; Flags: DWORD; ObjectTypeList: PObjectTypeList;
      ObjectTypeListLength: DWORD; const GenericMapping: TGenericMapping;  ObjectCreation: BOOL;
      var GrantedAccess: DWORD; var AccessStatus, pfGenerateOnClose: BOOL): BOOL; stdcall;
    function AccessCheckByTypeResultListAndAuditAlarm(SubsystemName: PKOLChar;
      HandleId: Pointer; ObjectTypeName, ObjectName: PKOLChar;
      SecurityDescriptor: PSecurityDescriptor; PrincipalSelfSid: PSID; DesiredAccess: DWORD;
      AuditType: AUDIT_EVENT_TYPE; Flags: DWORD; ObjectTypeList: PObjectTypeList;
      ObjectTypeListLength: DWORD; const GenericMapping: TGenericMapping;  ObjectCreation: BOOL;
      var GrantedAccess: DWORD; var AccessStatusList:DWORD; var pfGenerateOnClose: BOOL): BOOL; stdcall;
    function BackupEventLog(hEventLog: THandle; lpBackupFileName: PKOLChar): BOOL; stdcall;
    function ClearEventLog(hEventLog: THandle; lpBackupFileName: PKOLChar): BOOL; stdcall;
    function CreateProcessAsUser(hToken: THandle; lpApplicationName: PKOLChar;
      lpCommandLine: PKOLChar; lpProcessAttributes: PSecurityAttributes;
      lpThreadAttributes: PSecurityAttributes; bInheritHandles: BOOL;
      dwCreationFlags: DWORD; lpEnvironment: Pointer; lpCurrentDirectory: PKOLChar;
      const lpStartupInfo: TStartupInfo; var lpProcessInformation: TProcessInformation): BOOL; stdcall;
    function GetCurrentHwProfile(var lpHwProfileInfo: THWProfileInfo): BOOL; stdcall;
    function GetFileSecurity(lpFileName: PKOLChar; RequestedInformation: SECURITY_INFORMATION;
      pSecurityDescriptor: PSecurityDescriptor;nLength:DWORD;var lpnLengthNeeded: DWORD):BOOL; stdcall;
    function GetUserName(lpBuffer: PKOLChar; var nSize: DWORD): BOOL; stdcall;
    function InitiateSystemShutdown(lpMachineName, lpMessage: PKOLChar;
      dwTimeout: DWORD; bForceAppsClosed, bRebootAfterShutdown: BOOL): BOOL; stdcall;
    function LogonUser(lpszUsername, lpszDomain, lpszPassword: PKOLChar;
      dwLogonType, dwLogonProvider: DWORD; var phToken: THandle): BOOL; stdcall;
    function LookupAccountName(lpSystemName, lpAccountName: PKOLChar;
      Sid: PSID; var cbSid: DWORD; ReferencedDomainName: PKOLChar;
      var cbReferencedDomainName: DWORD; var peUse: SID_NAME_USE): BOOL; stdcall;
    function LookupAccountSid(lpSystemName: PKOLChar; Sid: PSID;
      Name: PKOLChar; var cbName: DWORD; ReferencedDomainName: PKOLChar;
      var cbReferencedDomainName: DWORD; var peUse: SID_NAME_USE): BOOL; stdcall;
    function LookupPrivilegeDisplayName(lpSystemName, lpName: PKOLChar;
      lpDisplayName: PKOLChar; var cbDisplayName, lpLanguageId: DWORD): BOOL; stdcall;
    function LookupPrivilegeName(lpSystemName: PKOLChar;
      var lpLuid: TLargeInteger; lpName: PKOLChar; var cbName: DWORD): BOOL; stdcall;
    function LookupPrivilegeValue(lpSystemName, lpName: PKOLChar;
      var lpLuid: TLargeInteger): BOOL; stdcall;
    function ObjectCloseAuditAlarm(SubsystemName: PKOLChar;
      HandleId: Pointer; GenerateOnClose: BOOL): BOOL; stdcall;
    function ObjectDeleteAuditAlarm(SubsystemName: PKOLChar;
      HandleId: Pointer; GenerateOnClose: BOOL): BOOL; stdcall;
    function ObjectOpenAuditAlarm(SubsystemName: PKOLChar; HandleId: Pointer;
      ObjectTypeName: PKOLChar; ObjectName: PKOLChar; pSecurityDescriptor: PSecurityDescriptor;
      ClientToken: THandle; DesiredAccess, GrantedAccess: DWORD;
      var Privileges: TPrivilegeSet; ObjectCreation, AccessGranted: BOOL;
      var GenerateOnClose: BOOL): BOOL; stdcall;
    function ObjectPrivilegeAuditAlarm(SubsystemName: PKOLChar;
      HandleId: Pointer; ClientToken: THandle; DesiredAccess: DWORD;
      var Privileges: TPrivilegeSet; AccessGranted: BOOL): BOOL; stdcall;
    function OpenBackupEventLog(lpUNCServerName, lpFileName: PKOLChar): THandle; stdcall;
    function OpenEventLog(lpUNCServerName, lpSourceName: PKOLChar): THandle; stdcall;
    function PrivilegedServiceAuditAlarm(SubsystemName, ServiceName: PKOLChar;
      ClientToken: THandle; var Privileges: TPrivilegeSet; AccessGranted: BOOL): BOOL; stdcall;
    function ReadEventLog(hEventLog: THandle; dwReadFlags, dwRecordOffset: DWORD;
      lpBuffer: Pointer; nNumberOfBytesToRead: DWORD;
      var pnBytesRead, pnMinNumberOfBytesNeeded: DWORD): BOOL; stdcall;
    function RegConnectRegistry(lpMachineName: PKOLChar; hKey: HKEY;
      var phkResult: HKEY): Longint; stdcall;
    function RegCreateKey(hKey: HKEY; lpSubKey: PKOLChar;
      var phkResult: HKEY): Longint; stdcall;
    function RegCreateKeyEx(hKey: HKEY; lpSubKey: PKOLChar;
      Reserved: DWORD; lpClass: PKOLChar; dwOptions: DWORD; samDesired: REGSAM;
      lpSecurityAttributes: PSecurityAttributes; var phkResult: HKEY;
      lpdwDisposition: PDWORD): Longint; stdcall;
    function RegDeleteKey(hKey: HKEY; lpSubKey: PKOLChar): Longint; stdcall;
    function RegDeleteValue(hKey: HKEY; lpValueName: PKOLChar): Longint; stdcall;
    function RegEnumKeyEx(hKey: HKEY; dwIndex: DWORD; lpName: PKOLChar;
      var lpcbName: DWORD; lpReserved: Pointer; lpClass: PKOLChar;
      lpcbClass: PDWORD; lpftLastWriteTime: PFileTime): Longint; stdcall;
    function RegEnumKey(hKey:HKEY; dwIndex:DWORD; lpName:PKOLChar; cbName:DWORD):Longint;stdcall;
    function RegEnumValue(hKey: HKEY; dwIndex: DWORD; lpValueName: PKOLChar;
      var lpcbValueName: DWORD; lpReserved: Pointer; lpType: PDWORD;
      lpData: PByte; lpcbData: PDWORD): Longint; stdcall;
    function RegLoadKey(hKey: HKEY; lpSubKey, lpFile: PKOLChar): Longint; stdcall;
    function RegOpenKey(hKey: HKEY; lpSubKey: PKOLChar;var phkResult: HKEY):Longint; stdcall;
    function RegOpenKeyEx(hKey: HKEY; lpSubKey: PKOLChar;
      ulOptions: DWORD; samDesired: REGSAM; var phkResult: HKEY): Longint; stdcall;
    function RegQueryInfoKey(hKey: HKEY; lpClass: PKOLChar;
      lpcbClass: PDWORD; lpReserved: Pointer;
      lpcSubKeys, lpcbMaxSubKeyLen, lpcbMaxClassLen, lpcValues,
      lpcbMaxValueNameLen, lpcbMaxValueLen, lpcbSecurityDescriptor: PDWORD;
      lpftLastWriteTime: PFileTime): Longint; stdcall;
    function RegQueryMultipleValues(hKey: HKEY; var ValList;
      NumVals: DWORD; lpValueBuf: PKOLChar; var ldwTotsize: DWORD): Longint; stdcall;
    function RegQueryValue(hKey: HKEY; lpSubKey: PKOLChar;
      lpValue: PKOLChar; var lpcbValue: Longint): Longint; stdcall;
    function RegQueryValueEx(hKey: HKEY; lpValueName: PKOLChar;
      lpReserved: Pointer; lpType: PDWORD; lpData: PByte; lpcbData: PDWORD): Longint; stdcall;
    function RegReplaceKey(hKey: HKEY; lpSubKey: PKOLChar;
       lpNewFile: PKOLChar; lpOldFile: PKOLChar): Longint; stdcall;
    function RegRestoreKey(hKey: HKEY; lpFile: PKOLChar; dwFlags: DWORD): Longint; stdcall;
    function RegSaveKey(hKey: HKEY; lpFile: PKOLChar;
      lpSecurityAttributes: PSecurityAttributes): Longint; stdcall;
    function RegSetValue(hKey: HKEY; lpSubKey: PKOLChar;
      dwType: DWORD; lpData: PKOLChar; cbData: DWORD): Longint; stdcall;
    function RegSetValueEx(hKey: HKEY; lpValueName: PKOLChar;
      Reserved: DWORD; dwType: DWORD; lpData: Pointer; cbData: DWORD): Longint; stdcall;
    function RegUnLoadKey(hKey: HKEY; lpSubKey: PKOLChar): Longint; stdcall;
    function RegisterEventSource(lpUNCServerName, lpSourceName: PKOLChar): THandle; stdcall;
    function ReportEvent(hEventLog: THandle; wType, wCategory: Word;
      dwEventID: DWORD; lpUserSid: Pointer; wNumStrings: Word;
      dwDataSize: DWORD; lpStrings, lpRawData: Pointer): BOOL; stdcall;
    function SetFileSecurity(lpFileName: PKOLChar; SecurityInformation: SECURITY_INFORMATION;
      pSecurityDescriptor: PSecurityDescriptor): BOOL; stdcall;

 Function wAddAtom( lpString : PKOLChar) : ATOM
 Function wBeginUpdateResource( pFileName : PKOLChar; bDeleteExistingResources : BOOL) : THandle
 //Function wCallNamedPipe( lpNamedPipeName : PKOLChar; lpInBuffer : Pointer; nInBufferSize : DWORD;
lpOutBuffer : Pointer; nOutBufferSize : DWORD; var lpBytesRead : DWORD; nTimeOut : DWORD) : BOOL
 //Function wCommConfigDialog( lpszName : PKOLChar; hWnd : HWND; var lpCC : TCommConfig) : BOOL
 Function wCompareString( Locale : LCID; dwCmpFlags : DWORD; lpString1 : PKOLChar; cchCount1 : Integer;
lpString2 : PKOLChar; cchCount2 : Integer) : Integer
 Function wCopyFile( lpExistingFileName, lpNewFileName : PKOLChar; bFailIfExists : BOOL) : BOOL
 //Function wCopyFileEx( lpExistingFileName, lpNewFileName : PKOLChar; lpProgressRoutine :
TFNProgressRoutine; lpData : Pointer; pbCancel : PBool; dwCopyFlags : DWORD) : BOOL
 Function wCreateDirectory( lpPathName : PKOLChar; lpSecurityAttributes : PSecurityAttributes) : BOOL
 Function wCreateDirectoryEx(lpTemplateDirectory,
lpNewDirectory:PKOLChar;lpSecAttrib:PSecurityAttribts):BOOL;
 Function wCreateEvent(lpEventAttribes:PSecurityAttrib;bManualReset,
bInitialState:BOOL;lpName:PKOLChar):THandle;
 Function wCreateFile( lpFileName : PKOLChar; dwDesiredAccess, dwShareMode : DWORD; lpSecurityAttributes :
PSecurityAttributes; dwCreationDisposition, dwFlagsAndAttributes:DWORD;hTemplateFile:THandle):THandle
 Function wCreateFileMapping( hFile : THandle; lpFileMappingAttributes : PSecurityAttributes; flProtect,
dwMaximumSizeHigh, dwMaximumSizeLow : DWORD; lpName : PKOLChar) : THandle
 Function wCreateHardLink(lpFileName,
lpExistingFileName:PKOLChar;lpSecurityAttributes:PSecurityAttributes):BOOL
 Function
CreateMailslot(lpName:PKOLChar;MaxMessSize:DWORD;lReadTimeout:DWORD;lpSecurityAttrib:PSecurityAttributes):THandle);
 Function wCreateNamedPipe( lpName : PKOLChar; dwOpenMode, dwPipeMode, nMaxInstances, nOutBufferSize,
nInBufferSize, nDefaultTimeOut : DWORD; lpSecurityAttributes : PSecurityAttributes) : THandle
 //Function CreateProcess( lpApplicationName : PKOLChar; lpCommandLine : PKOLChar; lpProcessAttributes,
lpThreadAttributes : PSecurityAttributes; bInheritHandles : BOOL; dwCreationFlags : DWORD; lpEnvironment :
Pointer; lpCurrentDirectory:PKOLChar;const lpStartupInfo:TStartupInfo;var
lpProcessInfo:TProcessInformation): BOOL
```

```
 Function wCreateSemaphore(lpSemaphoreAttributes: PSecurityAttributes; lInitialCount, lMaximumCount : Longint; lpName : PKOLChar) : THandle
 Function wCreateWaitableTimer(lpTimerAttributes:PSecurityAttribs;bManualReset:BOOL;lpTimerName:PKOLChar):THandle);
 Function wDefineDosDevice( dwFlags : DWORD; lpDeviceName, lpTargetPath : PKOLChar) : BOOL
 Function wDeleteFile( lpFileName : PKOLChar) : BOOL
 Function wEndUpdateResource( hUpdate : THandle; fDiscard : BOOL) : BOOL
 //Function wEnumCalendarInfo(lpCalInfEnumProc:TFNCalInfEnumProc;Locale:LCID;Calendar:CALID;CalType:CALTYPE):BOOL;
 //Function wEnumDateFormats(lpDateFmtEnumProc: TFNDateFmtEnumProc; Locale : LCID; dwFlags : DWORD) : BOOL
 //Function wEnumResourceNames(hModule:HMODULE;lpType:PKOLChar;lpEnumFunc:ENUMRESNAMEPROC;lParam:Longint):BOOL;
 //Function wEnumResourceTypes( hModule:HMODULE; lpEnumFunc:ENUMRESTYPEPROC;lParam:Longint):BOOL;
 //Function wEnumSystemCodePages( lpCodePageEnumProc : TFNCodepageEnumProc; dwFlags : DWORD) : BOOL
 //Function wEnumSystemLocales( lpLocaleEnumProc : TFNLocaleEnumProc; dwFlags : DWORD) : BOOL
 //Function wEnumTimeFormats(lpTimeFmtEnumProc:TFNTimeFmtEnumProc;Locale:LCID;dwFlags:DWORD):BOOL;
 Function wExpandEnvironmentStrings( lpSrc : PKOLChar; lpDst : PKOLChar; nSize : DWORD) : DWORD
 Procedure wFatalAppExit( uAction : UINT; lpMessageText : PKOLChar)
 //Function wFillConsoleOutputCharacter( hConsoleOutput : THandle; cCharacter : KOLChar; nLength : DWORD; dwWriteCoord : TCoord; var lpNumberOfCharsWritten : DWORD) : BOOL
 Function wFindAtom( lpString : PKOLChar) : ATOM
 Function wFindFirstChangeNotification(lpPathName:PKOLChar;bWatchSubtree:BOOL;dwNotifyFilter:DWORD):THandle;
 Function wFindFirstFile( lpFileName : PKOLChar; var lpFindFileData : TWIN32FindData) : THandle
 //Function wFindFirstFileEx( lpFileName : PKOLChar; fInfoLevelId : TFindexInfoLevels; lpFindFileData : Pointer; fSearchOp : TFindexSearchOps; lpSearchFilter : Pointer; dwAdditionalFlags : DWORD) : BOOL
 Function wFindNextFile( hFindFile : THandle; var lpFindFileData : TWIN32FindData) : BOOL
 Function wFindResource( hModule : HMODULE; lpName, lpType : PKOLChar) : HRSRC
 Function wFindResourceEx( hModule : HMODULE; lpType, lpName : PKOLChar; wLanguage : Word) : HRSRC
 Function wFoldString(dwMapFlags:DWORD;lpSrcStr:PKOLChar;cchSrc:Int;lpDestStr:PKOLChar;cchDest:Integer):Integer);
 //Function wFormatMessage( dwFlags : DWORD; lpSource : Pointer; dwMessageId : DWORD; dwLanguageId : DWORD; lpBuffer : PKOLChar; nSize : DWORD; Arguments : Pointer) : DWORD
 Function wFreeEnvironmentStrings( EnvBlock : PKOLChar) : BOOL
 Function wGetAtomName( nAtom : ATOM; lpBuffer : PKOLChar; nSize : Integer) : UINT
 Function wGetBinaryType( lpApplicationName : PKOLChar; var lpBinaryType : DWORD) : BOOL
 Function wGetCommandLine : PKOLChar
 //Function wGetCompressedFileSize( lpFileName : PKOLChar; lpFileSizeHigh : PDWORD) : DWORD
 Function wGetComputerName( lpBuffer : PKOLChar; var nSize : DWORD) : BOOL
 Function wGetConsoleTitle( lpConsoleTitle : PKOLChar; nSize : DWORD) : DWORD
 //Function wGetCurrencyFormat( Locale : LCID; dwFlags : DWORD; lpValue : PKOLChar; lpFormat : PCurrencyFmt; lpCurrencyStr : PKOLChar; cchCurrency : Integer) : Integer
 Function wGetCurrentDirectory( nBufferLength : DWORD; lpBuffer : PKOLChar) : DWORD
 //Function wGetDateFormat( Locale : LCID; dwFlags : DWORD; lpDate : PSystemTime; lpFormat : PKOLChar; lpDateStr : PKOLChar; cchDate : Integer) : Integer
 //Function wGetDefaultCommConfig( lpszName:PKOLChar;var lpCC : TCommConfig;var lpdwSize:DWORD):BOOL
 Function wGetDiskFreeSpace( lpRootPathName : PKOLChar; var lpSectorsPerCluster, lpBytesPerSector, lpNumberOfFreeClusters, lpTotalNumberOfClusters : DWORD) : BOOL
 //Function wGetDiskFreeSpaceEx( lpDirectoryName : PKOLChar; var lpFreeBytesAvailableToCaller, lpTotalNumberOfBytes, lpTotalNumberOfFreeBytes : PLargeInteger) : BOOL
 Function wGetDriveType( lpRootPathName : PKOLChar) : UINT
 Function wGetEnvironmentStrings : PKOLChar
 Function wGetEnvironmentVariable( lpName : PKOLChar; lpBuffer : PKOLChar; nSize : DWORD) : DWORD;
 Function wGetFileAttributes( lpFileName : PKOLChar) : DWORD
 //Function wGetFileAttributesEx(lpFileName:PKOLChar;fInfoLevelId:TGetFileExInfoLevs;lpFileInform:Pointer):BOOL;
 Function wGetFullPathName(lpFileName:PKOLChar;nBufferLength:WORD;lpBuffer:PKOLChar;var lpFilePart:PKOLChar):DWORD;
 //Function wGetLocaleInfo(Locale:LCID; LCType:LCTYPE;lpLCData:PKOLChar;cchData:Integer): Integer
 Function wGetLogicalDriveStrings( nBufferLength : DWORD; lpBuffer : PKOLChar) : DWORD
 Function wGetModuleFileName( hModule : HINST; lpFilename : PKOLChar; nSize : DWORD) : DWORD
 Function wGetModuleHandle( lpModuleName : PKOLChar) : HMODULE
 //Function wGetNamedPipeHandleState( hNamedPipe : THandle; lpState, lpCurInstances, lpMaxCollectionCount, lpCollectDataTimeout : PDWORD; lpUserName : PKOLChar; nMaxUserNameSize : DWORD) : BOOL
 //Function wGetNumberFormat( Locale : LCID; dwFlags:DWORD; lpValue:PKOLChar; lpFormat:PNumberFmt; lpNumberStr : PKOLChar; cchNumber : Integer) : Integer
 Function wGetPrivateProfileInt(lpAppName,lpKeyName:PKOLChar;nDefault:Integer;lpFileName:PKOLChar):UINT;
 Function wGetPrivateProfileSection(lpAppName:PKOLChar;lpRetrStr:PKOLChar;nSize:DWORD;pFileName:PKOLChar):DWORD;
 Function wGetPrivateProfileSectionNames(lpszReturnBuffer:PKOLChar;nSize:DWORD;lpFileName:PKOLChar):DWORD;
 Function wGetPrivateProfileString( lpAppName, lpKeyName, lpDefault : PKOLChar;lpReturnedStr: PKOLChar; nSize:DWORD; lpFileName : PKOLChar) : DWORD
 Function wGetProfileInt( lpAppName, lpKeyName : PKOLChar; nDefault : Integer) : UINT
 Function wGetProfileSection( lpAppName : PKOLChar; lpReturnedString : PKOLChar; nSize : DWORD) : DWORD
 Function wGetProfileString(lpAppName,lpKeyName, lpDefault:PKOLChar;lpReturnedStr:PKOLChar;nSize:DWORD):DWORD;
 Function wGetShortPathName( lpszLongPath:PKOLChar;lpszShortPath: PKOLChar; cchBuffer : DWORD) : DWORD
 //Procedure wGetStartupInfo( var lpStartupInfo : TStartupInfo)
// Function wGetStringTypeEx(Locale:LCID; dwInfoType:DWORD;lpSrcStr:PKOLChar;cchSrc:Integer;var lpCharType):BOOL
 Function wGetSystemDirectory( lpBuffer : PKOLChar; uSize : UINT) : UINT
 Function wGetTempFileName( lpPathName, lpPrefixString: PKOLChar;uUnique:UINT;lpTempFileName:PKOLChar):UINT
 Function wGetTempPath( nBufferLength : DWORD; lpBuffer : PKOLChar) : DWORD
//Function wGetTimeFormat(Loc:LCID;dwFlgs:DWORD;lpTime:PSystemTime;lpFrm:PKOLChar;lpTimeStr:PKOLChar;cTime:Int):Int
//Function wGetVersionEx( var lpVersionInformation : TOSVersionInfo) : BOOL
//Function GetVolumeInformation( lpRootPathName : PKOLChar; lpVolumeNameBuffer : PKOLChar; nVolumeNameSize : DWORD; lpVolumeSerialNumber : PDWORD; var lpMaximumComponentLength, lpFileSystemFlags : DWORD; lpFileSystemNameBuffer : PKOLChar; nFileSystemNameSize : DWORD) : BOOL
```

```
 Function wGetWindowsDirectory( lpBuffer : PKOLChar; uSize : UINT) : UINT
 Function wGlobalAddAtom( lpString : PKOLChar) : ATOM
 Function wGlobalFindAtom( lpString : PKOLChar) : ATOM
 Function wGlobalGetAtomName( nAtom : ATOM; lpBuffer : PKOLChar; nSize : Integer) : UINT
 Function wIsBadStringPtr( lpsz : PKOLChar; ucchMax : UINT) : BOOL
 Function wLCMapString(Loc:LCID;dwMapFlgs:DWORD;lpSrcStr:PKOLChar;cchSrc:Int;lpDestStr:PKOLChar;cchDest:Int):Int;
 Function wLoadLibrary( lpLibFileName : PKOLChar) : HMODULE
 Function wLoadLibraryEx( lpLibFileName : PKOLChar; hFile : THandle; dwFlags : DWORD) : HMODULE
 Function wMoveFile( lpExistingFileName, lpNewFileName : PKOLChar) : BOOL
 Function wMoveFileEx( lpExistingFileName, lpNewFileName : PKOLChar; dwFlags : DWORD) : BOOL
 //Function wMoveFileWithProgress( lpExistingFileName, lpNewFileName : PKOLChar; lpProgressRoutine : TFNProgressRoutine; lpData : Pointer; dwFlags : DWORD) : BOOL
 Function wOpenEvent( dwDesiredAccess : DWORD; bInheritHandle : BOOL;lpName:PKOLChar) : THandle
 Function wOpenFileMapping( dwDesiredAccess : DWORD; bInheritHandle:BOOL;lpName: PKOLChar):THandle
 Function wOpenMutex( dwDesiredAccess : DWORD; bInheritHandle : BOOL; lpName : PKOLChar) : THandle
 Function wOpenSemaphore( dwDesiredAccess : DWORD; bInheritHandle : BOOL; lpName : PKOLChar):THandle
 Function wOpenWaitableTimer(dwDesiredAccess:DWORD;bInheritHandle:BOOL;lpTimerName:PKOLChar):THandle
 Procedure wOutputDebugString( lpOutputString : PKOLChar)
 //Function wPeekConsoleInput(hConsoleInput:THandle;var lpBuffer:TInputRecord;nLength:DWORD;var lpNumberOfEventsRead:DWORD):BOOL;
 Function wQueryDosDevice( lpDeviceName : PKOLChar; lpTargetPath : PKOLChar; ucchMax : DWORD) : DWORD
 //Function wQueryRecoveryAgents(p1:PKOLChar;var p2:Pointer;var p3:TRecoveryAgentInformation):DWORD
 //Function wReadConsole( hConsoleInput : THandle; lpBuffer : Pointer; nNumberOfCharsToRead : DWORD; var lpNumberOfCharsRead : DWORD; lpReserved : Pointer) : BOOL
 //Function wReadConsoleInput(hConsInp:THandle;var lpBuf:TInpRec;nLength:DWORD;var lpNumbOfEventsRead:DWORD):BOOL;
 //Function wReadConsoleOutput( hConsoleOutput : THandle; lpBuffer : Pointer; dwBufferSize, dwBufferCoord : TCoord; var lpReadRegion : TSmallRect) : BOOL
 //Function wReadConsoleOutputCharacter( hConsoleOutput : THandle; lpCharacter : PKOLChar; nLength : DWORD; dwReadCoord : TCoord; var lpNumberOfCharsRead : DWORD) : BOOL
 Function wRemoveDirectory( lpPathName : PKOLChar) : BOOL
 //Function wScrollConsoleScreenBuffer( hConsoleOutput : THandle; const lpScrollRectangle : TSmallRect; lpClipRectangle : PSmallRect; dwDestinationOrigin : TCoord; var lpFill : TCharInfo) : BOOL
 Function wSearchPath( lpPath,lpFileName,lpExtension:PKOLChar;nBufferLength:DWORD;lpBuffer:PKOLChar;var lpFilePart:PKOLChar):DWORD;
 Function wSetComputerName( lpComputerName : PKOLChar) : BOOL
 Function wSetConsoleTitle( lpConsoleTitle : PKOLChar) : BOOL
 Function wSetCurrentDirectory( lpPathName : PKOLChar) : BOOL
 //Function wSetDefaultCommConfig( lpszName : PKOLChar; lpCC : PCommConfig; dwSize : DWORD) : BOOL
 Function wSetEnvironmentVariable( lpName, lpValue : PKOLChar) : BOOL
 Function wSetFileAttributes( lpFileName : PKOLChar; dwFileAttributes : DWORD) : BOOL
 //Function wSetLocaleInfo( Locale : LCID; LCType : LCTYPE; lpLCData : PKOLChar) : BOOL
 Function wSetVolumeLabel( lpRootPathName : PKOLChar; lpVolumeName : PKOLChar) : BOOL
 //Function wUpdateResource(hUpdate:THandle;lpType, lpName:PKOLChar;wLanguage:Word;lpData:Ptr;cbData:DWORD):BOOL
 Function wVerLanguageName( wLang : DWORD; szLang : PKOLChar; nSize : DWORD) : DWORD
 Function wWaitNamedPipe( lpNamedPipeName : PKOLChar; nTimeOut : DWORD) : BOOL
 //Function wWriteConsole( hConsoleOutput : THandle; const lpBuffer : Pointer; nNumberOfCharsToWrite : DWORD; var lpNumberOfCharsWritten : DWORD; lpReserved : Pointer) : BOOL
 //Function wWriteConsoleInput( hConsoleInput : THandle; const lpBuffer : TInputRecord; nLength : DWORD; var lpNumberOfEventsWritten : DWORD) : BOOL
 //Function wWriteConsoleOutput(hConsoleOutput:THandle; lpBuffer:Pointer; dwBufferSize,dwBufferCoord : TCoord; var lpWriteRegion : TSmallRect) : BOOL
 //Function wWriteConsoleOutputCharacter( hConsoleOutput : THandle; lpCharacter : PKOLChar; nLength : DWORD; dwWriteCoord : TCoord; var lpNumberOfCharsWritten : DWORD) : BOOL
 Function wWritePrivateProfileSection( lpAppName, lpString, lpFileName : PKOLChar) : BOOL
 Function wWritePrivateProfileString( lpAppName, lpKeyName, lpString, lpFileName : PKOLChar) : BOOL
 Function wWriteProfileSection( lpAppName, lpString : PKOLChar) : BOOL
 Function wWriteProfileString( lpAppName, lpKeyName, lpString : PKOLChar) : BOOL
 Function wlstrcat( lpString1, lpString2 : PKOLChar) : PKOLChar
 Function wlstrcmp( lpString1, lpString2 : PKOLChar) : Integer
 Function wlstrcmpi( lpString1, lpString2 : PKOLChar) : Integer
 Function wlstrcpy( lpString1, lpString2 : PKOLChar) : PKOLChar
 Function wlstrcpyn( lpString1, lpString2 : PKOLChar; iMaxLength : Integer) : PKOLChar
 Function wlstrlen( lpString : PKOLChar) : Integer
 //Function wMultinetGetConnectionPerformance( lpNetResource : PNetResource; lpNetConnectInfoStruc : PNetConnectInfoStruct) : DWORD
 //Function wWNetAddConnection2(var lpNetResource:TNetResource;lpPassword, lpUserName:PKOLChar;dwFlags:DWORD):DWORD;
 //Function wWNetAddConnection3( hwndOwner : HWND; var lpNetResource:TNetResource;lpPassword, lpUserName:PKOLChar; dwFlags : DWORD) : DWORD
 Function wWNetAddConnection( lpRemoteName, lpPassword, lpLocalName : PKOLChar) : DWORD
 Function wWNetCancelConnection2( lpName : PKOLChar; dwFlags : DWORD; fForce : BOOL) : DWORD
 Function wWNetCancelConnection( lpName : PKOLChar; fForce : BOOL) : DWORD
 //Function wWNetConnectionDialog1( var lpConnDlgStruct : TConnectDlgStruct) : DWORD
 //Function wWNetDisconnectDialog1( var lpConnDlgStruct : TDiscDlgStruct) : DWORD
 //Function wWNetEnumResource(hEnum:THandle;var lpcCount:DWORD;lpBuffer:Ptr;var lpBufferSize:DWORD):DWORD;
 Function wWNetGetConnection(lpLocalName:PKOLChar;lpRemoteName:PKOLChar; var lpnLength:DWORD):DWORD;
 Function wWNetGetLastError( var lpError : DWORD; lpErrorBuf : PKOLChar; nErrorBufSize : DWORD; lpNameBuf : PKOLChar; nNameBufSize : DWORD) : DWORD
 //Function wWNetGetNetworkInformation(lpProvider:PKOLChar;var lpNetInfoStruct:TNetInfoStruct):DWORD;
 Function wWNetGetProviderName(dwNetType:DWORD;lpProviderName:PKOLChar;var lpBufferSize:DWORD):DWORD;
 //Function wWNetGetResourceParent(lpNetResource:PNetResource;lpBuffer:Pointer;var cbBuffer:DWORD):DWORD;
//Function wWNetGetUniversalName(lpLocalPath:PKOLChar;dwInfoLevel:DWORD;lpBuffer:Ptr;var lpBufferSize:DWORD):DWORD;
 Function wWNetGetUser( lpName : PKOLChar; lpUserName : PKOLChar; var lpnLength : DWORD) : DWORD
// Function wWNetOpenEnum(dwScope,dwType,dwUsage:DWORD;lpNetResource:PNetRes;var lphEnum:THandle):DWORD;
// Function wWNetSetConnection( lpName : PKOLChar; dwProperties : DWORD; pvValues : Pointer) : DWORD
```

```
 //Function wWNetUseConnection(hwndOwner:HWND;var lpNetResource:TNetResource;lpUserID:PKOLChar;lpPassword:PKOLChar; dwFlags:DWORD;lpAccessName:PKOLChar;var lpBufferSize:DWORD;var lpResult:DWORD):DWORD
 Function wGetFileVersionInfo(lptstrFilename:PKOLChar;dwHandle,dwLen:DWORD;lpData:Pointer):BOOL;
 Function wGetFileVersionInfoSize( lptstrFilename : PKOLChar; var lpdwHandle : DWORD) : DWORD
 Function wVerFindFile( uFlags : DWORD; szFileName, szWinDir, szAppDir, szCurDir : PKOLChar; var lpuCurDirLen : UINT; szDestDir : PKOLChar; var lpuDestDirLen : UINT) : DWORD
 Function wVerInstallFile( uFlags : DWORD; szSrcFileName, szDestFileName, szSrcDir, szDestDir, szCurDir, szTmpFile : PKOLChar; var lpuTmpFileLen : UINT) : DWORD
//Function wVerQueryValue(pBlock:Pointer;lpSubBlock:PKOLChar;var lplpBuffer:Ptr;var puLen:UINT):BOOL;
//Function wGetPrivateProfileStruct(lpszSection, lpszKey:PKOLChar;lpStruct:Ptr;uSizeStruct:UINT;szFile:PKOLChar):BOOL;
//Function wWritePrivateProfileStruct(lpszSection, lpszKey:PKOLChar;lpStruct:Ptr;uSizeStruct:UINT;szFile:PKOLChar):BOOL;
 Function wAddFontResource( FileName : PKOLChar) : Integer
 //Function wAddFontResourceEx( p1 : PKOLChar; p2 : DWORD; p3 : PDesignVector) : Integer
 Function wCopyEnhMetaFile( p1 : HENHMETAFILE; p2 : PKOLChar) : HENHMETAFILE
 Function wCopyMetaFile( p1 : HMETAFILE; p2 : PKOLChar) : HMETAFILE
 //Function wCreateColorSpace( var ColorSpace : TLogColorSpace) : HCOLORSPACE
 //Function wCreateDC( lpszDriver, lpszDevice, lpszOutput : PKOLChar; lpdvmInit : PDeviceMode) : HDC
// Function wCreateEnhMetaFile( DC : HDC; FileName : PKOLChar; Rect : PRect; Desc : PKOLChar) : HDC
 Function wCreateFont( nHeight, nWidth, nEscapement, nOrientaion, fnWeight : Integer; fdwItalic, fdwUnderline, fdwStrikeOut,fdwCharSet,fdwOutputPrec,fdwClipPrecision,fdwQualy, fdwPitchAndFamily:DWORD;lpszFace:PKOLChar):HFONT;
 Function wCreateFontIndirect( const p1 : TLogFont) : HFONT
 //Function wCreateFontIndirectEx( const p1 : PEnumLogFontExDV) : HFONT
// Function wCreateIC( lpszDriver, lpszDevice, lpszOutput : PKOLChar; lpdvmInit : PDeviceMode) : HDC
 Function wCreateMetaFile( p1 : PKOLChar) : HDC
 Function wCreateScalableFontResource( p1 : DWORD; p2, p3, p4 : PKOLChar) : BOOL
//Function wDeviceCapabilities(pDriverNa,pDeviceNam, pPort:PKOLChar;iIdx:Int;pOut:PKOLChar;DevMod:PDeviceMode):Int;
// Function wEnumFontFamilies( DC : HDC; p2 : PKOLChar; p3 : TFNFontEnumProc; p4 : LPARAM) : BOOL
 //Function wEnumFontFamiliesEx(DC:HDC;var p2:TLogFont;p3:TFNFontEnumProc;p4:LPARAM;p5:DWORD):BOOL);
 //Function wEnumFonts(DC:HDC;lpszFace:PKOLChar;fntenmprc:TFNFontEnumProc;lpszData:PKOLChar):Integer;
 //Function wEnumICMProfiles( DC : HDC; ICMProc : TFNICMEnumProc; p3 : LPARAM) : Integer
 //Function wExtTextOut(DC:HDC;X, Y:Int;Options:Longint;Rect:PRect;Str:PKOLChar;Count:Longint;Dx:PInteger:BOOL
 //Function wGetCharABCWidths( DC : HDC; FirstChar, LastChar : UINT; const ABCStructs) : BOOL
 //Function wGetCharABCWidthsFloat( DC : HDC; FirstChar, LastChar : UINT; const ABCFloatSturcts) : BOOL
 //Function wGetCharWidth32( DC : HDC; FirstChar, LastChar : UINT; const Widths) : BOOL
 //Function wGetCharWidth( DC : HDC; FirstChar, LastChar : UINT; const Widths) : BOOL
// Function wGetCharWidthFloat( DC : HDC; FirstChar, LastChar : UINT; const Widths) : BOOL
// Function wGetCharacterPlacement(DC:HDC;p2:PKOLChar;p3,p4:BOOL;var p5:TGCPResults;p6:DWORD):DWORD
 Function wGetEnhMetaFile( p1 : PKOLChar) : HENHMETAFILE
 Function wGetEnhMetaFileDescription( p1 : HENHMETAFILE; p2 : UINT; p3 : PKOLChar) : UINT
// Function wGetGlyphIndices( DC : HDC; p2 : PKOLChar; p3 : Integer; p4 : PWORD; p5 : DWORD) : DWORD
// Function wGetGlyphOutline( DC : HDC; uChar,uFormat:UINT;const lpgm:TGlyphMetrics; cbBuffer : DWORD; lpvBuffer : Pointer; const lpmat2 : TMat2) : DWORD
 Function wGetICMProfile( DC : HDC; var Size : DWORD; Name : PKOLChar) : BOOL
// Function wGetLogColorSpace( p1 : HCOLORSPACE; var ColorSpace : TLogColorSpace; Size : DWORD) : BOOL
 Function wGetMetaFile( p1 : PKOLChar) : HMETAFILE
// Function wGetObject( p1 : HGDIOBJ; p2 : Integer; p3 : Pointer) : Integer
 //Function wGetOutlineTextMetrics( DC : HDC; p2 : UINT; OTMetricStructs : Pointer) : UINT
 //Function wGetTextExtentExPoint(DC:HDC;p2:PKOLChar; p3,p4:Integer;p5,p6:PInteger;var p7:TSize):BOOL
 Function wGetTextExtentPoint32( DC : HDC; Str : PKOLChar; Count : Integer; var Size : TSize) : BOOL
 Function wGetTextExtentPoint( DC : HDC; Str : PKOLChar; Count : Integer; var Size : TSize) : BOOL
 Function wGetTextFace( DC : HDC; Count : Integer; Buffer : PKOLChar) : Integer
 //Function wGetTextMetrics( DC : HDC; var TM : TTextMetric) : BOOL
 Function wPolyTextOut( DC : HDC; const PolyTextArray, Strings : Integer) : BOOL
 Function wRemoveFontResource( FileName : PKOLChar) : BOOL
 //Function wRemoveFontResourceEx( p1 : PKOLChar; p2 : DWORD; p3 : PDesignVector) : BOOL
 //Function wResetDC( DC : HDC; const InitData : TDeviceMode) : HDC
 Function wSetICMProfile( DC : HDC; Name : PKOLChar) : BOOL
 //Function wStartDoc( DC : HDC; const p2 : TDocInfo) : Integer
 Function wTextOut( DC : HDC; X, Y : Integer; Str : PKOLChar; Count : Integer) : BOOL
 Function wUpdateICMRegKey( p1 : DWORD; p2, p3 : PKOLChar; p4 : UINT) : BOOL
 Function wwglUseFontBitmaps( DC : HDC; p2, p3, p4 : DWORD) : BOOL
 //Function wwglUseFontOutlines(p1:HDC; p2,p3,p4:DWORD;p5,p6:Single;p7:Int;p8:PGlyphMetricsFloat):BOOL
 Function wAppendMenu( hMenu : HMENU; uFlags, uIDNewItem : UINT; lpNewItem : PKOLChar) : BOOL
 Function wCallMsgFilter( var lpMsg : TMsg; nCode : Integer) : BOOL
 //Function wCallWindowProc(lpPrevWndFunc:TFNWndProc;hWnd:HWND;Msg:UINT;wParam:WPARAM;lParam:LPARAM):LRESULT
 //Function wChangeDisplaySettings( var lpDevMode : TDeviceMode; dwFlags : DWORD) : Longint
// Function wChangeDisplaySettingsEx( lpszDeviceName:PKOLChar;var lpDevMode: TDeviceMode; wnd : HWND; dwFlags : DWORD; lParam : Pointer) : Longint
 Function wChangeMenu(hMenu:HMENU;cmd:UINT;lpszNewItem:PKOLChar;cmdInsert:UINT;flags:UINT):BOOL;
 Function wCharLower( lpsz : PKOLChar) : PKOLChar
 Function wCharLowerBuff( lpsz : PKOLChar; cchLength : DWORD) : DWORD
 Function wCharNext( lpsz : PKOLChar) : PKOLChar
 //Function wCharNextEx( CodePage : Word; lpCurrentChar : LPCSTR; dwFlags : DWORD) : LPSTR
 Function wCharPrev( lpszStart : PKOLChar; lpszCurrent : PKOLChar) : PKOLChar
// Function wCharPrevEx( CodePage : Word; lpStart, lpCurrentChar : LPCSTR; dwFlags : DWORD) : LPSTR
 Function wCharToOem( lpszSrc : PKOLChar; lpszDst : PKOLChar) : BOOL
 Function wCharToOemBuff( lpszSrc : PKOLChar; lpszDst : PKOLChar; cchDstLength : DWORD) : BOOL
 Function wCharUpper( lpsz : PKOLChar) : PKOLChar
 Function wCharUpperBuff( lpsz : PKOLChar; cchLength : DWORD) : DWORD
 Function wCopyAcceleratorTable( hAccelSrc : HACCEL; var lpAccelDst, cAccelEntries : Integer) : Integer
 Function wCreateAcceleratorTable( var Accel, Count : Integer) : HACCEL
 //Function wCreateDesktop(lpszDesktop, lpszDevice:PKOLChar;pDevmode:PDeviceMode;dwFlags:DWORD;dwDesiredAccess:DWORD;lpsa:PSecurityAttribs):HDESK
 //Function wCreateDialogIndirectParam( hInstance : HINST; const lpTemplate : TDlgTemplate; hWndParent : HWND; lpDialogFunc : TFNDlgProc; dwInitParam : LPARAM) : HWND
 //Function wCreateDialogParam( hInstance : HINST; lpTemplateName : PKOLChar; hWndParent : HWND; lpDialogFunc : TFNDlgProc; dwInitParam : LPARAM) : HWND
 Function wCreateMDIWindow( lpClassName, lpWindowName:PKOLChar;dwStyle: DWORD; X,Y,nWidth,nHeight:Integer; hWndParent : HWND; hInstance : HINST; lParam : LPARAM) : HWND
//Function wCreateWindowEx(dwExStyle:DWORD;lpClassName:PKOLChar;lpWindowName:PKOLChar;dwStyle DWORD;X,Y, nWidth, nHeight:Int WndParent:HWND;hMenu:HMENU;hInstance:HINST;lpParam:Pointer):HWND
 //Function wCreateWindowStation(lpwinsta:PKOLChar;dwReserv, dwDesiredAccess:DWORD;lpsa:PSecurityAttribs):HWINSTA;
 Function wDefDlgProc( hDlg : HWND; Msg : UINT; wParam : WPARAM; lParam : LPARAM) : LRESULT
 Function wDefFrameProc(hWnd,hWndMDIClient:HWND;uMsg:UINT;wParam:WPARAM;lParam:LPARAM):LRESULT;
 Function wDefMDIChildProc( hWnd : HWND; uMsg : UINT; wParam : WPARAM; lParam: LPARAM):LRESULT;
 Function wDefWindowProc( hWnd : HWND; Msg : UINT; wParam : WPARAM; lParam:LPARAM): LRESULT
 //Function wDialogBoxIndirectParam( hInstance : HINST; const lpDialogTemplate : TDlgTemplate; hWndParent : HWND; lpDialogFunc : TFNDlgProc; dwInitParam : LPARAM) : Integer
 //Function wDialogBoxParam( hInstance : HINST; lpTemplateName : PKOLChar; hWndParent : HWND; lpDialogFunc : TFNDlgProc; dwInitParam : LPARAM) : Integer
 Function wDispatchMessage( const lpMsg : TMsg) : Longint
 Function wDlgDirList(hDlg:HWND;lpPathSpec:PKOLChar;nIDListBox, nIDStaticPath:Integer;uFileType:UINT):Integer;
 Function wDlgDirListComboBox(hDlg:HWND;lpPathSpec:PKOLChar;nIDComboBox, nIDStaticPath:Int;uFiletype:UINT):Int;
 Function wDlgDirSelectComboBoxEx( hDlg : HWND; lpString:PKOLChar;nCount,nIDComboBox:Integer): BOOL
 Function wDlgDirSelectEx( hDlg : HWND; lpString : PKOLChar; nCount, nIDListBox : Integer) : BOOL
//Function wDrawState(DC:HDC;Brush:HBRUSH;CBFunc:TFNDrawStateProc;lData:LPARA;wDat:WPARA;x,y,cx, cy:Int;Flags:UINT):BOOL;
 Function wDrawText(hDC:HDC;lpString:PKOLChar;nCount:Integer;var lpRect:TRect;uFormat:UINT):Integer;
 Function wFindWindow( lpClassName, lpWindowName : PKOLChar) : HWND
 Function wFindWindowEx( Parent, Child : HWND; ClassName, WindowName : PKOLChar) : HWND
 //Function wGetAltTabInfo(hwnd:HWND;iItem:Int;var pati:TAltTabInfo;pszItemText:PKOLChar;cchItemText:UINT):BOOL;
// Function wGetClassInfo( hInstance : HINST; lpClassName : PKOLChar; var lpWndClass : TWndClass) : BOOL
 //Function wGetClassInfoEx( Instance : HINST; Classname : PKOLChar; var WndClass : TWndClassEx) : BOOL
 Function wGetClassLong( hWnd : HWND; nIndex : Integer) : DWORD
 Function wGetClassName( hWnd : HWND; lpClassName : PKOLChar; nMaxCount : Integer) : Integer
 Function wGetClipboardFormatName( format : UINT; lpszFormatName:PKOLChar;cchMaxCount:Integer):Integer;
 Function wGetDlgItemText( hDlg: HWND;nIDDlgItem:Integer;lpString:PKOLChar;nMaxCount:Integer):UINT
 Function wGetKeyNameText( lParam : Longint; lpString : PKOLChar; nSize : Integer) : Integer
 Function wGetKeyboardLayoutName( pwszKLID : PKOLChar) : BOOL
 //Function wGetMenuItemInfo( p1 : HMENU; p2 : UINT; p3 : BOOL; var p4 : TMenuItemInfo) : BOOL
 Function wGetMenuString(hMenu:HMENU;uIDItem:UINT;lpString:PKOLChar;nMaxCount:Integer;uFlag:UINT):Integer;
 Function wGetMessage( var lpMsg : TMsg; hWnd : HWND; wMsgFilterMin, wMsgFilterMax : UINT) : BOOL
 Function wGetProp( hWnd : HWND; lpString : PKOLChar) : THandle
 //Function wGetTabbedTextExtent( hDC : HDC; lpString : PKOLChar; nCount, nTabPositions : Integer; var lpnTabStopPositions) : DWORD
//Function wGetUserObjectInformation(hObj:THandle;nIndex:Int;pvInfo:Ptr;nLength:DWORD;var lpnLengthNeed:DWORD)BOOL;
 Function wGetWindowLong( hWnd : HWND; nIndex : Integer) : Longint
 Function wGetWindowModuleFileName( hwnd : HWND; pszFileName : PKOLChar; cchFileNameMax : UINT) : UINT
 Function wGetWindowText( hWnd : HWND; lpString : PKOLChar; nMaxCount : Integer) : Integer
 Function wGetWindowTextLength( hWnd : HWND) : Integer
//Function wGrayString(hDC:HDC;hBrush:HBRUSH;lpOutFunc:TFNGrayStrProc;lpDat:LPARA;nCnt,X,Y,nWidt, nHeigt:Int):BOOL;
 Function wInsertMenu( hMenu : HMENU; uPosition, uFlags, uIDNewItem : UINT; lpNewItem : PKOLChar) : BOOL
 //Function wInsertMenuItem( p1 : HMENU; p2 : UINT; p3 : BOOL; const p4 : TMenuItemInfo) : BOOL
 Function wIsCharAlpha( ch : KOLChar) : BOOL
 Function wIsCharAlphaNumeric( ch : KOLChar) : BOOL
 Function wIsCharLower( ch : KOLChar) : BOOL
 Function wIsCharUpper( ch : KOLChar) : BOOL
 Function wIsDialogMessage( hDlg : HWND; var lpMsg : TMsg) : BOOL
 Function wLoadAccelerators( hInstance : HINST; lpTableName : PKOLChar) : HACCEL
 Function wLoadBitmap( hInstance : HINST; lpBitmapName : PKOLChar) : HBITMAP
 Function wLoadCursor( hInstance : HINST; lpCursorName : PKOLChar) : HCURSOR
 Function wLoadCursorFromFile( lpFileName : PKOLChar) : HCURSOR
 Function wLoadIcon( hInstance : HINST; lpIconName : PKOLChar) : HICON
 Function wLoadImage(hInst:HINST;ImageName:PKOLChar;ImageType:UINT;X,Y:Integer;Flags:UINT): THandle
 Function wLoadKeyboardLayout( pwszKLID : PKOLChar; Flags : UINT) : HKL
 Function wLoadMenu( hInstance : HINST; lpMenuName : PKOLChar) : HMENU
 //Function wLoadMenuIndirect( lpMenuTemplate : Pointer) : HMENU
 Function wLoadString(hInstance:HINST;uID: UINT; lpBuffer :PKOLChar;nBufferMax:Integer):Integer
 Function wMapVirtualKey( uCode, uMapType : UINT) : UINT
 Function wMapVirtualKeyEx( uCode, uMapType : UINT; dwhkl : HKL) : UINT
 Function wMessageBox( hWnd : HWND; lpText, lpCaption : PKOLChar; uType : UINT) : Integer
 Function wMessageBoxEx( hWnd:HWND; lpText,lpCaption:PKOLChar;uType:UINT;wLanguageId:Word): Integer
 //Function wMessageBoxIndirect( const MsgBoxParams : TMsgBoxParams) : BOOL
 Function wModifyMenu( hMnu : HMENU; uPosition, uFlags, uIDNewItem : UINT; lpNewItem : PKOLChar) : BOOL
 //Function wOemToAnsi( const lpszSrc : LPCSTR; lpszDst : LPSTR) : BOOL
 //7Function wOemToAnsiBuff( lpszSrc : LPCSTR; lpszDst : LPSTR; cchDstLength : DWORD) : BOOL
 //Function wOemToChar( lpszSrc : PKOLChar; lpszDst : PKOLChar) : BOOL
 Function wOemToCharBuff( lpszSrc : PKOLChar; lpszDst : PKOLChar; cchDstLength : DWORD) : BOOL
 Function wOpenDesktop(lpszDesktop:PKOLChar;dwFlags:DWORD;fInherit:BOOL;dwDesiredAccess:DWORD): HDESK
 Function wOpenWindowStation( lpszWinSta : PKOLChar; fInherit : BOOL; dwDesiredAccess : DWORD): HWINSTA
 Function wPeekMessage( var lpMsg : TMsg; hWnd: HWND;wMsgFilterMin,wMsgFilterMax,wRemoveMsg:UINT):BOOL
 Function wPostMessage( hWnd : HWND; Msg : UINT; wParam : WPARAM; lParam : LPARAM) : BOOL
 Function wPostThreadMessage(idThread:DWORD;Msg : UINT; wParam : WPARAM; lParam : LPARAM) : BOOL
 Function wRealGetWindowClass( hwnd : HWND; pszType : PKOLChar; cchType : UINT) : UINT
// Function wRegisterClass( const lpWndClass : TWndClass) : ATOM
// Function wRegisterClassEx( const WndClass : TWndClassEx) : ATOM
```

```
 Function wRegisterClipboardFormat( lpszFormat : PKOLChar) : UINT
// Function wRegisterDeviceNotification(hRecipient:THandle;NotificFilter:Pointer;Flags:DWORD):HDEVNOTIFY
 Function wRegisterWindowMessage( lpString : PKOLChar) : UINT
 Function wRemoveProp( hWnd : HWND; lpString : PKOLChar) : THandle
 Function wSendDlgItemMessage(hDlg:HWND;nIDDlgItem:Integer;Msg:UINT;wParam:WPARAM;lParam:LPARAM):Longint;
 Function wSendMessage( hWnd : HWND; Msg : UINT; wParam : WPARAM; lParam : LPARAM) : LRESULT
 //Function wSendMessageCallback( hWnd : HWND; Msg : UINT; wParam : WPARAM; lParam:LPARAM;
lpResultCallBack : TFNSendAsyncProc; dwData : DWORD) : BOOL
 Function wSendMessageTimeout(hWnd:HWND;Msg:UINT;wParam:WPARAM;lParam:LPARAM; fuFlags,uTimeout:UINT;var
lpdwResult:DWORD): LRESULT
 Function wSendNotifyMessage( hWnd : HWND; Msg : UINT; wParam : WPARAM; lParam : LPARAM) : BOOL
 Function wSetClassLong( hWnd : HWND; nIndex : Integer; dwNewLong : Longint) : DWORD
 Function wSetDlgItemText( hDlg : HWND; nIDDlgItem : Integer; lpString : PKOLChar) : BOOL
 //Function wSetMenuItemInfo( p1 : HMENU; p2 : UINT; p3 : BOOL; const p4 : TMenuItemInfo) : BOOL
 Function wSetProp( hWnd : HWND; lpString : PKOLChar; hData : THandle) : BOOL
// Function wSetUserObjectInformation(hObj:THandle;nIndex:Integer;pvInfo:Pointer;nLength:DWORD):BOOL
 Function wSetWindowLong( hWnd : HWND; nIndex : Integer; dwNewLong : Longint) : Longint
 Function wSetWindowText( hWnd : HWND; lpString : PKOLChar) : BOOL
 //Function wSetWindowsHook( nFilterType : Integer; pfnFilterProc : TFNHookProc) : HHOOK
 //Function wSetWindowsHookEx(idHook:Integer;lpfn:TFNHookProc;hmod:HINST;dwThreadId:DWORD):HHOOK;
// Function wSystemParametersInfo( uiAction, uiParam : UINT; pvParam : Pointer; fWinIni: UINT):BOOL
 Function wTabbedTextOut(hDC:HDC;X,Y:Int;lpString:PKOLChar;nCount,nTabPositions:Int;var
lpnTabStopPositions,nTabOrigin:Int):Longint;
 Function wTranslateAccelerator( hWnd : HWND; hAccTable : HACCEL; var lpMsg : TMsg) : Integer
 Function wUnregisterClass( lpClassName : PKOLChar; hInstance : HINST) : BOOL
 Function wVkKeyScan( ch : KOLChar) : SHORT
 Function wVkKeyScanEx( ch : KOLChar; dwhkl : HKL) : SHORT
 Function wWinHelp( hWndMain : HWND; lpszHelp : PKOLChar; uCommand : UINT; dwData : DWORD) : BOOL
 Function wwsprintf( Output : PKOLChar; Format : PKOLChar) : Integer
 Function wwvsprintf( Output : PKOLChar; Format : PKOLChar; arglist : va_list) : Integer

//TestDrive!
'SID_REVISION','LongInt'(1);'FILENAME_ADVAPI32','String').SetString('ADVAPI32.DLL
 'PROC_CONVERTSIDTOSTRINGSIDA','String').SetString( 'ConvertSidToStringSidA
 Function GetDomainUserSidS(const domainName:String;const userName:String; var foundDomain:String):String;
 Function GetLocalUserSidStr( const UserName : string) : string
 Function getPid4user( const domain : string; const user : string; var pid : dword) : boolean
 Function Impersonate2User( const domain : string; const user : string) : boolean
 Function GetProcessUserBypid( pid : DWORD; var UserName, Domain : AnsiString) : Boolean
 Function KillProcessbyname( const exename : string; var found : integer) : integer
 Function getWinProcessList : TStringList
end;

procedure SIRegister_AfSafeSync(CL: TPSPascalCompiler);
begin
 'AfMaxSyncSlots','LongInt'( 64);
 'AfSynchronizeTimeout','LongInt'( 2000);
  TAfSyncSlotID', 'DWORD
  TAfSyncStatistics','record MessagesCount:Int;TimeoutMessages:Int;DisabledMessages:Int;end;
  TAfSafeSyncEvent', 'Procedure ( ID : TAfSyncSlotID)
  TAfSafeDirectSyncEvent', 'Procedure
 Function AfNewSyncSlot( const AEvent : TAfSafeSyncEvent) : TAfSyncSlotID
 Function AfReleaseSyncSlot( const ID : TAfSyncSlotID) : Boolean
 Function AfEnableSyncSlot( const ID : TAfSyncSlotID; Enable : Boolean) : Boolean
 Function AfValidateSyncSlot( const ID : TAfSyncSlotID) : Boolean
 Function AfSyncEvent( const ID : TAfSyncSlotID; Timeout : DWORD) : Boolean
 Function AfDirectSyncEvent( Event : TAfSafeDirectSyncEvent; Timeout : DWORD) : Boolean
 Function AfIsSyncMethod : Boolean
 Function AfSyncWnd : HWnd
 Function AfSyncStatistics : TAfSyncStatistics
 Procedure AfClearSyncStatistics
end;

procedure SIRegister_AfComPortCore(CL: TPSPascalCompiler);
begin
 'fBinary','LongWord')( $00000001);
 'fParity','LongWord'( $00000002);
 'fOutxCtsFlow','LongWord').SetUInt( $00000004);
 'fOutxDsrFlow','LongWord')( $00000008);
 'fDtrControl','LongWord')( $00000030);
 'fDtrControlDisable','LongWord')( $00000000);
 'fDtrControlEnable','LongWord')( $00000010);
 'fDtrControlHandshake','LongWord')( $00000020);
 'fDsrSensitivity','LongWord')( $00000040);
 'fTXContinueOnXoff','LongWord')( $00000080);
 'fOutX','LongWord')( $00000100);
 'fInX','LongWord')( $00000200);
 'fErrorChar','LongWord')( $00000400);
 'fNull','LongWord')( $00000800);
 'fRtsControl','LongWord')( $00003000);
 'fRtsControlDisable','LongWord')( $00000000);
 'fRtsControlEnable','LongWord')( $00001000);
 'fRtsControlHandshake','LongWord')( $00002000);
 'fRtsControlToggle','LongWord')( $00003000);
 'fAbortOnError','LongWord')( $00004000);
 'fDummy2','LongWord')( $FFFF8000);
  TAfCoreEvent', '( ceOutFree, ceLineEvent, ceNeedReadData, ceException )
  FindClass('TOBJECT'),'EAfComPortCoreError
  FindClass('TOBJECT'),'TAfComPortCore
  TAfComPortCoreEvent', 'Procedure ( Sender : TAfComPortCore; Even'
   +'tKind : TAfCoreEvent; Data : DWORD)
  SIRegister_TAfComPortCoreThread(CL);
  SIRegister_TAfComPortEventThread(CL);
  SIRegister_TAfComPortWriteThread(CL);
  SIRegister_TAfComPortCore(CL);
 Function FormatDeviceName( PortNumber : Integer) : string
end;

procedure SIRegister_ApplicationFileIO(CL: TPSPascalCompiler);
begin
  TAFIOFileStreamEvent', 'Function ( const fileName : String; mode: Word) : TStream
  TAFIOFileStreamExistsEvent', 'Function ( const fileName : String) : Boolean
  SIRegister_TApplicationFileIO(CL);
  TDataFileCapability', '( dfcRead, dfcWrite )
  TDataFileCapabilities', 'set of TDataFileCapability
  SIRegister_TDataFile(CL);
  //TDataFileClass', 'class of TDataFile
 Function ApplicationFileIODefined : Boolean
 Function CreateFileStream(const fileName: String; mode: WordfmShareDenyNone):TStream
 Function FileStreamExists(const fileName: String) : Boolean
 //Procedure Register
end;

procedure SIRegister_ALFBXLib(CL: TPSPascalCompiler);
begin
  TALFBXFieldType', '( uftUnKnown, uftNumeric, uftChar, uftVarchar'
   +', uftCstring, uftSmallint, uftInteger, uftQuad, uftFloat, uftDoublePrecisi'
   +'on, uftTimestamp, uftBlob, uftBlobId, uftDate, uftTime, uftInt64, uftArray, uftNull )
  TALFBXScale', 'Integer
  FindClass('TOBJECT'),'EALFBXConvertError
  SIRegister_EALFBXError(CL);
  SIRegister_EALFBXException(CL);
  FindClass('TOBJECT'),'EALFBXGFixError
  FindClass('TOBJECT'),'EALFBXDSQLError
  FindClass('TOBJECT'),'EALFBXDynError
  FindClass('TOBJECT'),'EALFBXGBakError
  FindClass('TOBJECT'),'EALFBXGSecError
  FindClass('TOBJECT'),'EALFBXLicenseError
  FindClass('TOBJECT'),'EALFBXGStatError
  //EALFBXExceptionClass', 'class of EALFBXError
  TALFBXCharacterSet', '( csNONE, csASCII, csBIG_5, csCYRL, csDOS4'
   +'37, csDOS850, csDOS852, csDOS857, csDOS860, csDOS861, csDOS863, csDOS865, '
   +'csEUCJ_0208, csGB_2312, csISO8859_1, csISO8859_2, csKSC_5601, csNEXT, csOC'
   +'TETS, csSJIS_0208, csUNICODE_FSS, csUTF8, csWIN1250, csWIN1251, csWIN1252,'
   +' csWIN1253, csWIN1254, csDOS737, csDOS775, csDOS858, csDOS862, csDOS864, c'
   +'sDOS866, csDOS869, csWIN1255, csWIN1256, csWIN1257, csISO8859_3, csISO8859'
   +'_4, csISO8859_5, csISO8859_6, csISO8859_7, csISO8859_8, csISO8859_9, csISO'
   +'8859_13, csKOI8R, csKOI8U, csWIN1258, csTIS620, csGBK, csCP943C )
  TALFBXTransParam', '( tpConsistency, tpConcurrency, tpShared, tp'
   +'Protected, tpExclusive, tpWait, tpNowait, tpRead, tpWrite, tpLockRead, tpL'
   +'ockWrite, tpVerbTime, tpCommitTime, tpIgnoreLimbo, tpReadCommitted, tpAuto'
   +'Commit, tpRecVersion, tpNoRecVersion, tpRestartRequests, tpNoAutoUndo, tpLockTimeout )
  TALFBXTransParams', 'set of TALFBXTransParam
 Function ALFBXStrToCharacterSet( const CharacterSet : AnsiString) : TALFBXCharacterSet
 Function ALFBXCreateDBParams( Params : AnsiString; Delimiter : Char) : AnsiString
 Function ALFBXCreateBlobParams( Params : AnsiString; Delimiter : Char) : AnsiString
 'cALFBXMaxParamLength','LongInt'( 125);
  TALFBXParamsFlag', '( pfNotInitialized, pfNotNullable )
  TALFBXParamsFlags', 'set of TALFBXParamsFlag
  //PALFBXSQLVar', '^TALFBXSQLVar // will not work
  //PALFBXSQLDaData', '^TALFBXSQLDaData // will not work
  TALFBXStatementType', '( stSelect, stInsert, stUpdate, stDelete,'
   +' stDDL, stGetSegment, stPutSegment, stExecProcedure, stStartTrans, stCommi'
   +'t, stRollback, stSelectForUpdate, stSetGenerator, stSavePoint )
  SIRegister_TALFBXSQLDA(CL);
  //PALFBXPtrArray', '^TALFBXPtrArray // will not work
  SIRegister_TALFBXPoolStream(CL);
  //PALFBXBlobData', '^TALFBXBlobData // will not work
  TALFBXBlobData', 'record Size : Integer; Buffer : string; end
  //PALFBXArrayDesc', '^TALFBXArrayDesc // will not work
  //TALFBXArrayDesc', 'TISCArrayDesc
  //TALFBXBlobDesc', 'TISCBlobDesc
  //PALFBXArrayInfo', '^TALFBXArrayInfo // will not work
  //TALFBXArrayInfo', 'record index : Integer; size : integer; info: TALFBXArrayDesc; end
  SIRegister_TALFBXSQLResult(CL);
  //TALFBXSQLResultClass', 'class of TALFBXSQLResult
  SIRegister_TALFBXSQLParams(CL);
  //TALFBXSQLParamsClass', 'class of TALFBXSQLParams
  TALFBXDSQLInfoData', 'record InfoCode : byte; InfoLen : Word; St'
   +'atementType : TALFBXStatementType; end
  FindClass('TOBJECT'),'TALFBXLibrary
  //PALFBXStatusVector', '^TALFBXStatusVector // will not work
  TALFBXOnConnectionLost', 'Procedure ( Lib : TALFBXLibrary)
  //TALFBXOnGetDBExceptionClass', 'Procedure ( Number : Integer; out'
   //+' Excep : EALFBXExceptionClass)
  SIRegister_TALFBXLibrary(CL);
 'cAlFBXDateOffset','LongInt'( 15018);
 'cALFBXTimeCoeff','LongInt'( 864000000);
 //Procedure ALFBXDecodeTimeStamp( v : PISCTimeStamp; out DateTime : Double);
 //Procedure ALFBXDecodeTimeStamp1( v : PISCTimeStamp; out TimeStamp : TTimeStamp);
 //Function ALFBXDecodeTimeStamp2( v : PISCTimeStamp) : Double;
 Procedure ALFBXDecodeSQLDate( v : Integer; out Year : SmallInt; out Month, Day : Word)
 Procedure ALFBXDecodeSQLTime(v:Cardinal;out Hour,Minute,Second:Word; out Fractions: LongWord)
 //Procedure ALFBXEncodeTimeStamp( const DateTime : TDateTime; v : PISCTimeStamp);
 //Procedure ALFBXEncodeTimeStamp1( const Date : Integer; v : PISCTimeStamp);
 //Procedure ALFBXEncodeTimeStamp2( const Time : Cardinal; v : PISCTimeStamp);
 Function ALFBXEncodeSQLDate( Year : Integer; Month, Day : Integer) : Integer
 Function ALFBXEncodeSQLTime( Hour, Minute, Second : Word; var Fractions : LongWord): Cardinal
  TALFBXParamType', '( prNone, prByte, prShrt, prCard, prStrg, prIgno )
  TALFBXDPBInfo', 'record Name : AnsiString; ParamType : TALFBXParamType; end
 Function ALFBXSQLQuote( const name : AnsiString) : AnsiString
 Function ALFBXSQLUnQuote( const name : AnsiString) : AnsiString
end;

procedure SIRegister_ALFBXClient(CL: TPSPascalCompiler);
begin
  TALFBXClientSQLParam', 'record Value : AnsiString; IsNull : Boolean; end
  TALFBXClientSQLParams', 'array of TALFBXClientSQLParam
  TALFBXClientSelectDataSQL', 'record SQL : AnsiString; Params : T'
   +'ALFBXClientSQLParams; RowTag : AnsiString; ViewTag : AnsiString; Skip : in'
   +'teger; First : Integer; CacheThreshold : Integer; end
  TALFBXClientSelectDataSQLs', 'array of TALFBXClientSelectDataSQL
  TALFBXClientUpdateDataSQL', 'record SQL : AnsiString; Params: TALFBXClientSQLParams; end
  TALFBXClientUpdateDataSQLs', 'array of TALFBXClientUpdateDataSQL
  TALFBXClientMonitoringIOStats', 'record page_reads : int64; page'
   +'_writes : int64; page_fetches : int64; page_marks : int64; end
  SIRegister_TALFBXClient(CL);
  SIRegister_TALFBXConnectionStatementPoolBinTreeNode(CL);
  SIRegister_TALFBXConnectionStatementPoolBinTree(CL);
  SIRegister_TALFBXConnectionWithStmtPoolContainer(CL);
  SIRegister_TALFBXConnectionWithoutStmtPoolContainer(CL);
  SIRegister_TALFBXReadTransactionPoolContainer(CL);
  SIRegister_TALFBXReadStatementPoolContainer(CL);
  SIRegister_TALFBXStringKeyPoolBinTreeNode(CL);
  SIRegister_TALFBXConnectionPoolClient(CL);
  SIRegister_TALFBXEventThread(CL);
  Function AlMySqlClientSlashedStr( const Str : AnsiString) : AnsiString
end;

procedure SIRegister_ovcBidi(CL: TPSPascalCompiler);
begin
_OSVERSIONINFOA = record
    dwOSVersionInfoSize: DWORD;
    dwMajorVersion: DWORD;
    dwMinorVersion: DWORD;
    dwBuildNumber: DWORD;
    dwPlatformId: DWORD;
    szCSDVersion: array[0..127] of AnsiChar; { Maintenance AnsiString for PSS usage }
  end;
 TOSVersionInfoA', '_OSVERSIONINFOA
  TOSVersionInfo', 'TOSVersionInfoA
 'WS_EX_RIGHT','LongWord')( $00001000);
 'WS_EX_LEFT','LongWord')( $00000000);
 'WS_EX_RTLREADING','LongWord')( $00002000);
 'WS_EX_LTRREADING','LongWord')( $00000000);
 'WS_EX_LEFTSCROLLBAR','LongWord')( $00004000);
 'WS_EX_RIGHTSCROLLBAR','LongWord')( $00000000);
 Function SetProcessDefaultLayout( dwDefaultLayout : DWORD) : BOOL
 'LAYOUT_RTL','LongWord')( $00000001);
 'LAYOUT_BTT','LongWord')( $00000002);
 'LAYOUT_VBH','LongWord')( $00000004);
 'LAYOUT_BITMAPORIENTATIONPRESERVED','LongWord')( $00000008);
 'NOMIRRORBITMAP','LongWord')( DWORD ( $80000000 ));
 Function SetLayout( dc : HDC; dwLayout : DWORD) : DWORD
 Function GetLayout( dc : hdc) : DWORD
 Function IsBidi : Boolean
 Function GetCurrentHwProfile( var lpHwProfileInfo : THWProfileInfo) : BOOL
 Function GetVersionEx( var lpVersionInformation : TOSVersionInfo) : BOOL
 Function SetPriorityClass( hProcess : THandle; dwPriorityClass: DWORD) : BOOL
 Function GetPriorityClass( hProcess : THandle) : DWORD
 Function OpenClipboard( hWndNewOwner : HWND) : BOOL
 Function CloseClipboard : BOOL
 Function GetClipboardSequenceNumber : DWORD
 Function GetClipboardOwner : HWND
 Function SetClipboardViewer( hWndNewViewer : HWND) : HWND
 Function GetClipboardViewer : HWND
 Function ChangeClipboardChain( hWndRemove, hWndNewNext : HWND) : BOOL
 Function SetClipboardData( uFormat : UINT; hMem : THandle) : THandle
 Function GetClipboardData( uFormat : UINT) : THandle
 Function RegisterClipboardFormat( lpszFormat : PChar) : UINT
 Function CountClipboardFormats : Integer
 Function EnumClipboardFormats( format : UINT) : UINT
 Function GetClipboardFormatName(format:UINT;lpszFormatName:PChar;cchMaxCount:Integer):Integer
 Function EmptyClipboard : BOOL
 Function IsClipboardFormatAvailable( format : UINT) : BOOL
  Function GetPriorityClipboardFormat( var paFormatPriorityList, cFormats : Integer) : Integer
 Function GetOpenClipboardWindow : HWND
 Function EndDialog( hDlg : HWND; nResult : Integer) : BOOL
 Function GetDlgItem( hDlg : HWND; nIDDlgItem : Integer) : HWND
 Function SetDlgItemInt( hDlg : HWND; nIDDlgItem : Integer; uValue : UINT; bSigned: BOOL): BOOL
 Function GetDlgItemInt(hDlg:HWND;nIDDlgItem:Integer;var lpTranslated:BOOL;bSigned: BOOL): UINT
 Function SetDlgItemText( hDlg : HWND; nIDDlgItem : Integer; lpString : PChar) : BOOL
 Function CheckDlgButton( hDlg : HWND; nIDButton : Integer; uCheck : UINT) : BOOL
 Function CheckRadioButton( hDlg : HWND; nIDFirstButton, nIDLastButton,nIDCheckButton:Integer) : BOOL
 Function IsDlgButtonChecked( hDlg : HWND; nIDButton : Integer) : UINT
 Function SendDlgItemMessage(hDlg:HWND;nIDDlgItem:Int;Msg:UINT;wParam:WPARAM;lParam:LPARAM):Longint;
end;

procedure SIRegister_DXPUtils(CL: TPSPascalCompiler);
begin
 Function glExecuteAndWait(cmdLine:String;visibility:Word;timeout:Cardinal;killAppOnTimeOut:Bool):Int;
 Function GetTemporaryFilesPath : String
 Function GetTemporaryFileName : String
 Function FindFileInPaths( const fileName, paths : String) : String
 Function PathsToString( const paths : TStrings) : String
 Procedure StringToPaths( const pathsString : String; paths : TStrings)
 //Function MacroExpandPath( const aPath : String) : String
end;

procedure SIRegister_ALMultiPartBaseParser(CL: TPSPascalCompiler);
begin
  SIRegister_TALMultiPartBaseContent(CL);
  SIRegister_TALMultiPartBaseContents(CL);
  SIRegister_TAlMultiPartBaseStream(CL);
  SIRegister_TALMultipartBaseEncoder(CL);
  SIRegister_TALMultipartBaseDecoder(CL);
 Function ALMultipartExtractBoundaryFromContentType( aContentType : AnsiString) : AnsiString
 Function ALMultipartExtractSubValueFromHeaderLine(aHeaderLine:AnsiString;aName:AnsiString):AnsiString;
 Function ALMultipartSetSubValueInHeaderLine(aHeaderLine:AnsiString;aName,AValue:AnsiString):AnsiString;
end;

procedure SIRegister_SmallUtils(CL: TPSPascalCompiler);
begin
  TdriveSize', 'record FreeS : Int64; TotalS : Int64; end
  TWinVerRec', 'record WinPlatform : Integer; WinMajorVersion : In'
   +teger; WinMinorVersion :Integer; WinBuildNumber : Integer; WinCSDVersion: String; end
 Function aAllocPadedMem( Size : Cardinal) : TObject
 Procedure aFreePadedMem( var P : TObject);
 Procedure aFreePadedMem1( var P : PChar);
 Function aCheckPadedMem( P : Pointer) : Byte
 Function aGetPadMemSize( P : Pointer) : Cardinal
 Function aAllocMem( Size : Cardinal) : Pointer
 Function aStrLen( const Str : PChar) : Cardinal
 Function aStrLCopy( Dest : PChar; const Source : PChar; MaxLen : Cardinal) : PChar
 Function aStrECopy( Dest : PChar; const Source : PChar) : PChar
 Function aStrCopy( Dest : PChar; const Source : PChar) : PChar
 Function aStrEnd( const Str : PChar) : PChar
 Function aStrScan( const Str : PChar; aChr : Char) : PChar
 Function aStrMove( Dest : PChar; const Source : PChar; Count : Cardinal) : PChar
 Function aPCharLength( const Str : PChar) : Cardinal
 Function aPCharUpper( Str : PChar) : PChar
 Function aPCharLower( Str : PChar) : PChar
 Function aStrCat( Dest : PChar; const Source : PChar) : PChar
 Function aLastDelimiter( const Delimiters, S : String) : Integer
 Function aCopyTail( const S : String; Len : Integer) : String
 Function aInt2Thos( I : Int64) : String
 Function aUpperCase( const S : String) : String
 Function aLowerCase( const S : string) : String
 Function aCompareText( const S1, S2 : string) : Integer
 Function aSameText( const S1, S2 : string) : Boolean
 Function aInt2Str( Value : Int64) : String
 Function aStr2Int( const Value : String) : Int64
 Function aStr2IntDef( const S : string; Default : Int64) : Int64
 Function aGetFileExt( const FileName : String) : String
 Function aGetFilePath( const FileName : String) : String
 Function aGetFileName( const FileName : String) : String
 Function aChangeExt( const FileName, Extension : String) : String
 Function aAdjustLineBreaks( const S : string) : string
 Function aGetWindowStr( WinHandle : HWND) : String
 Function aDiskSpace( Drive : String) : TdriveSize
 Function aFileExists( FileName : String) : Boolean
 Function aFileSize( FileName : String) : Int64
 Function aDirectoryExists( const Name : string) : Boolean
 Function aSysErrorMessage( ErrorCode : Integer) : string
 Function aShortPathName( const LongName : string) : string
 Function aGetWindowVer : TWinVerRec
 procedure InitDriveSpacePtr;
end;

procedure SIRegister_MakeApp(CL: TPSPascalCompiler);
begin
 aZero','LongInt'( 0);
 'makeappDEF','LongInt'( - 1);
  'CS_VREDRAW','LongInt'( DWORD ( 1 ));
 'CS_HREDRAW','LongInt'( DWORD ( 2 ));
 'CS_KEYCVTWINDOW','LongInt'( 4);
 'CS_DBLCLKS','LongInt'( 8);
 'CS_OWNDC','LongWord')( $20);
 'CS_CLASSDC','LongWord')( $40);
 'CS_PARENTDC','LongWord')( $80);
 'CS_NOKEYCVT','LongWord')( $100);
 'CS_NOCLOSE','LongWord')( $200);
 'CS_SAVEBITS','LongWord')( $800);
 'CS_BYTEALIGNCLIENT','LongWord')( $1000);
 'CS_BYTEALIGNWINDOW','LongWord')( $2000);
 'CS_GLOBALCLASS','LongWord')( $4000);
 'CS_IME','LongWord')( $10000);
 'CS_DROPSHADOW','LongWord')( $20000);
  //PPanelFunc', '^TPanelFunc // will not work
  TPanelStyle', '(psEdge, psTabEdge, psBorder, psTabBorder, psTab, psNone )
  TFontLook', '( flBold, flItalic, flUnderLine, flStrikeOut )
  TFontLooks', 'set of TFontLook
 TMessagefunc','function(hWnd,iMsg,wParam,lParam:Integer):Integer)
 Function SetWinClass(const ClassName:String; pMessFunc: Tmessagefunc; wcStyle : Integer): Word
 Function SetWinClassO( const ClassName : String; pMessFunc : TObject; wcStyle : Integer): Word
 Function SetWinClass( const ClassName : String; pMessFunc : TObject; wcStyle : Integer) : Word
 Function MakeForm(Left,Top,Width,Height:Integer;const Caption:String;WinStyle:Integer):Integer
 Procedure RunMsgLoop( Show : Boolean)
 Function MakeFont(Height,Width:Integer; const FontName:String; Look:TFontLooks; Roman:Boolean): Integer
 Function MakeButton(Left,Top,Width,Height:Integer;pCaption:PChar;hParent,
ID_Number:Cardinal;hFont:Int):Int;
 Function MakeListBox(Left,Top,Width,Height,Parent:Integer;const ListItems:String;WinStyle:Integer):Integer
 Function MakeComboBox(Left,Top,Width,Height,Parent:Integer;const ListItems:String;WinStyle:Integer):Int
 Function MakePanel(Left,Top,Width,Height,
hParent:Int;WndFunc:TPanelFunc;ID_Number:Card;Style:TPanelStyle):Int;
 Function MakeSubMenu(const ItemList : String; ID1, ID2 : Cardinal; hMenu : Integer) : Integer
 Function id4menu( a, b : Byte; c : Byte; d : Byte) : Cardinal
 Procedure DoInitMakeApp   //set first to init formclasscontrol!
end;

procedure SIRegister_ScreenSaver(CL: TPSPascalCompiler);
begin
  TScreenSaverOption', '( ssoAutoAdjustFormProperties, ssoAutoHook'
   +'KeyboardEvents, ssoAutoHookMouseEvents, ssoEnhancedMouseMoveDetection )
  TScreenSaverOptions', 'set of TScreenSaverOption
 'cDefaultScreenSaverOptions','LongInt').Value.ts32:=ord(ssoAutoAdjustFormProperties) or
ord(ssoAutoHookKeyboardEvents) or ord(ssoEnhancedMouseMoveDetection);
  TScreenSaverPreviewEvent', 'Procedure ( Sender : TObject; previewHwnd: HWND)
  SIRegister_TScreenSaver(CL);
 //Procedure Register
 Procedure SetScreenSaverPassword
end;

procedure SIRegister_XCollection(CL: TPSPascalCompiler);
begin
  FindClass('TOBJECT'),'TXCollection
  SIRegister_EFilerException(CL);
  SIRegister_TXCollectionItem(CL);
  //TXCollectionItemClass', 'class of TXCollectionItem
  SIRegister_TXCollection(CL);
 Procedure RegisterXCollectionDestroyEvent( notifyEvent : TNotifyEvent)
 Procedure DeRegisterXCollectionDestroyEvent( notifyEvent : TNotifyEvent)
 Procedure RegisterXCollectionItemClass( aClass : TXCollectionItemClass)
 Procedure UnregisterXCollectionItemClass( aClass : TXCollectionItemClass)
 Function FindXCollectionItemClass( const className : String) : TXCollectionItemClass
 Function GetXCollectionItemClassesList( baseClass : TXCollectionItemClass) : TList
end;

procedure SIRegister_XOpenGL(CL: TPSPascalCompiler);
begin
  TMapTexCoordMode', '(mtcmUndefined, mtcmNull, mtcmMain, mtcmDual, mtcmSecond,mtcmArbitrary);
 Procedure xglMapTexCoordToNull
 Procedure xglMapTexCoordToMain
 Procedure xglMapTexCoordToSecond
 Procedure xglMapTexCoordToDual
 Procedure xglMapTexCoordToArbitrary( const units : array of Cardinal);
 Procedure xglMapTexCoordToArbitrary1( const bitWiseUnits : Cardinal);
 Procedure xglMapTexCoordToArbitraryAdd( const bitWiseUnits : Cardinal)
 Procedure xglBeginUpdate
 Procedure xglEndUpdate
 Procedure xglPushState
 Procedure xglPopState
 Procedure xglForbidSecondTextureUnit
 Procedure xglAllowSecondTextureUnit
 Function xglGetBitWiseMapping : Cardinal
end;

procedure SIRegister_VectorLists(CL: TPSPascalCompiler);
begin
  TBaseListOption', '( bloExternalMemory, bloSetCountResetsMemory)
  TBaseListOptions', 'set of TBaseListOption
  SIRegister_TBaseList(CL);
  SIRegister_TBaseVectorList(CL);
  SIRegister_TAffineVectorList(CL);
  SIRegister_TVectorList(CL);
  SIRegister_TTexPointList(CL);
  SIRegister_TXIntegerList(CL);
  //PSingleArrayList', '^TSingleArrayList // will not work
  SIRegister_TSingleList(CL);
  SIRegister_TByteList(CL);
  SIRegister_TQuaternionList(CL);
 Procedure QuickSortLists( startIndex, endIndex : Integer; refList : TSingleList; objList : TList);
 Procedure QuickSortLists1( startIndex, endIndex : Integer; refList : TSingleList; objList : TBaseList);
 Procedure FastQuickSortLists(startIndex,
endIndex:Integer;refList:TSingleList;objList:TPersistentObjectList);
end;

procedure SIRegister_MeshUtils(CL: TPSPascalCompiler);
begin
 Procedure ConvertStripToList( const strip : TAffineVectorList; list : TAffineVectorList);
 Procedure ConvertStripToList1( const strip : TIntegerList; list : TIntegerList);
 Procedure ConvertStripToList2(const strip:TAffineVectorList;const
indices:TIntegerList;list:TAffineVectorList);
 Procedure ConvertIndexedListToList(const data:TAffineVectlist;const
indices:TIntegerList;list:TAffineVectorList);
 Function BuildVectorCountOptimizedIndices(const vertices:TAffineVectorList; const
normals:TAffineVectorList; const texCoords : TAffineVectorList) : TIntegerList
 Procedure RemapReferences( reference : TAffineVectorList; const indices : TIntegerList);
 Procedure RemapReferences1( reference : TIntegerList; const indices : TIntegerList);
 Procedure RemapAndCleanupReferences( reference : TAffineVectorList; indices : TIntegerList)
 Function RemapIndicesToIndicesMap( remapIndices : TIntegerList) : TIntegerList
 Procedure RemapTrianglesIndices( indices, indicesMap : TIntegerList)
 Procedure RemapIndices( indices, indicesMap : TIntegerList)
 Procedure UnifyTrianglesWinding( indices : TIntegerList)
 Procedure InvertTrianglesWinding( indices : TIntegerList)
 Function BuildNormals( reference : TAffineVectorList; indices : TIntegerList) : TAffineVectorList
 Function BuildNonOrientedEdgesList(triangleIndices:TIntegerList; triangleEdges : TIntegerList;
edgesTriangles : TIntegerList) : TIntegerList
 Procedure WeldVertices( vertices : TAffineVectorList; indicesMap : TIntegerList; weldRadius : Single)
 Function StripifyMesh(indices:TIntegerList;maxVertexIndex:Integer;agglomerateLoneTriangles:Boolean):
TPersistentObjectList;
 Procedure IncreaseCoherency( indices : TIntegerList; cacheSize : Integer)
 Procedure SubdivideTriangles( smoothFactor : Single; vertices : TAffineVectorList; triangleIndices :
TIntegerList; normals : TAffineVectorList; onSubdivideEdge : TSubdivideEdgeEvent)
end;

procedure SIRegister_JclSysUtils(CL: TPSPascalCompiler);
begin
 Procedure GetAndFillMem( var P : TObject; const Size : Integer; const Value : Byte)
 Procedure FreeMemAndNil( var P : TObject)
 Function PCharOrNil( const S : string) : PChar
  SIRegister_TJclReferenceMemoryStream(CL);
  FindClass('TOBJECT'),'EJclVMTError
 {Function GetVirtualMethodCount( AClass : TClass) : Integer
 Function GetVirtualMethod( AClass : TClass; const Index : Integer) : Pointer
 Procedure SetVirtualMethod( AClass : TClass; const Index : Integer; const Method:Pointer)
  PDynamicIndexList', '^TDynamicIndexList // will not work
  PDynamicAddressList', '^TDynamicAddressList // will not work
 Function GetDynamicMethodCount( AClass : TClass) : Integer
 Function GetDynamicIndexList( AClass : TClass) : PDynamicIndexList
 Function GetDynamicAddressList( AClass : TClass) : PDynamicAddressList
 Function HasDynamicMethod( AClass : TClass; Index : Integer) : Boolean
 Function GetDynamicMethod( AClass : TClass; Index : Integer) : Pointer
 Function GetInitTable( AClass : TClass) : PTypeInfo
  PFieldEntry', '^TFieldEntry // will not work}
  TFieldEntry', 'record OffSet : Integer; IDX : Word; Name : Short'
   +'String; end
 Function JIsClass( Address : Pointer) : Boolean
 Function JIsObject( Address : Pointer) : Boolean
 Function GetImplementorOfInterface( const I : IInterface) : TObject
  TDigitCount', 'Integer
  SIRegister_TJclNumericFormat(CL);
 Function JIntToStrZeroPad( Value, Count : Integer) : AnsiString
  TTextHandler', 'Procedure ( const Text : string)
// 'ABORT_EXIT_CODE','LongInt'( ERROR_CANCELLED 1223);
 Function JExecute(const
CommandLine:string;OutputLineCallback:TTextHandler;RawOutpt:Bool;AbortPtr:PBool):Card;
 Function JExecute1(const CommandLine:string;var Output:string; RawOutput:Bool; AbortPtr:PBool):Cardinal;
 Function ReadKey : Char   //to and from the DOS console !
  TModuleHandle', 'HINST
  //TModuleHandle', 'Pointer
 'INVALID_MODULEHANDLE_VALUE','LongInt'( TModuleHandle ( 0 ));
 Function LoadModule( var Module : TModuleHandle; FileName : string) : Boolean
 Function LoadModuleEx( var Module : TModuleHandle; FileName : string; Flags : Cardinal) : Boolean
 Procedure UnloadModule( var Module : TModuleHandle)
 Function GetModuleSymbol( Module : TModuleHandle; SymbolName : string) : Pointer
 Function GetModuleSymbolEx( Module : TModuleHandle; SymbolName : string; var Accu : Boolean) : Pointer
 Function ReadModuleData(Module:TModuleHandle;SymbolName:string;var Buffer,Size: Cardinal):Boolean;
 Function WriteModuleData(Module:TModuleHandle;SymbolName:string;var Buffer,Size:Cardinal):Boolean;
  FindClass('TOBJECT'),'EJclConversionError
 Function JStrToBoolean( const S : string) : Boolean
 Function JBooleanToStr( B : Boolean) : string
 Function JIntToBool( I : Integer) : Boolean
 Function JBoolToInt( B : Boolean) : Integer
 'ListSeparator','String ';
 'ListSeparator1','String ':
 Procedure ListAddItems( var List : string; const Separator, Items : string)
 Procedure ListIncludeItems( var List : string; const Separator, Items : string)
 Procedure ListRemoveItems( var List : string; const Separator, Items : string)
 Procedure ListDelItem( var List : string; const Separator : string; const Index : Integer)
 Function ListItemCount( const List, Separator : string) : Integer
 Function ListGetItem( const List, Separator : string; const Index : Integer) : string
 Procedure ListSetItem(var List:string;const Separator:string;const Index:Integer;const Value:string)
 Function ListItemIndex( const List, Separator, Item : string) : Integer
 Function SystemTObjectInstance : LongWord
 Function IsCompiledWithPackages : Boolean
 Function JJclGUIDToString( const GUID : TGUID) : string
 Function JJclStringToGUID( const S : string) : TGUID
  SIRegister_TJclIntfCriticalSection(CL);
  SIRegister_TJclSimpleLog(CL);
 Procedure InitSimpleLog( const ALogFileName : string)
end;

procedure SIRegister_JclBorlandTools(CL: TPSPascalCompiler);
begin
  FindClass('TOBJECT'),'EJclBorRADException
  TJclBorRADToolKind', '( brDelphi, brCppBuilder, brBorlandDevStudio )
  TJclBorRADToolEdition', '( deOPEN, dePRO, deSVR )
  TJclBorRADToolEdition', '( deSTD, dePRO, deCSS, deARC )
  TJclBorRADToolPath', 'string
 'SupportedDelphiVersions','LongInt'( 5 or  6 or  7 or  8 or  9 or  10 or  11);
 'SupportedBCBVersions','LongInt'( 5 or  6 or  10 or  11);
 'SupportedBDSVersions','LongInt'( 1 or  2 or  3 or  4 or  5);
 BorRADToolRepositoryPagesSection','String 'Repository Pages
 BorRADToolRepositoryDialogsPage','String 'Dialogs
 BorRADToolRepositoryFormsPage','String 'Forms
 BorRADToolRepositoryProjectsPage','String 'Projects
 BorRADToolRepositoryDataModulesPage','String 'Data Modules
 BorRADToolRepositoryObjectType','String 'Type
 BorRADToolRepositoryFormTemplate','String 'FormTemplate
 BorRADToolRepositoryProjectTemplate','String 'ProjectTemplate
 BorRADToolRepositoryObjectName','String 'Name
 BorRADToolRepositoryObjectPage','String 'Page
 BorRADToolRepositoryObjectIcon','String 'Icon
 BorRADToolRepositoryObjectDescr','String 'Description
 BorRADToolRepositoryObjectAuthor','String 'Author
 BorRADToolRepositoryObjectAncestor','String 'Ancestor
 BorRADToolRepositoryObjectDesigner','String 'Designer
 BorRADToolRepositoryDesignerDfm','String 'dfm
 BorRADToolRepositoryDesignerXfm','String 'xfm
 BorRADToolRepositoryObjectNewForm','String 'DefaultNewForm
 BorRADToolRepositoryObjectMainForm','String 'DefaultMainForm
 SourceExtensionDelphiPackage','String '.dpk
 SourceExtensionBCBPackage','String '.bpk
 SourceExtensionDelphiProject','String '.dpr
 SourceExtensionBCBProject','String '.bpr
 SourceExtensionBDSProject','String '.bdsproj
 SourceExtensionDProject','String '.dproj
 BinaryExtensionPackage','String '.bpl
 BinaryExtensionLibrary','String '.dll
 BinaryExtensionExecutable','String '.exe
 CompilerExtensionDCP','String '.dcp
 CompilerExtensionBPI','String '.bpi
 CompilerExtensionLIB','String '.lib
 CompilerExtensionTDS','String '.tds
 CompilerExtensionMAP','String '.map
 CompilerExtensionDRC','String '.drc
 CompilerExtensionDEF','String '.def
 SourceExtensionCPP','String '.cpp
 SourceExtensionH','String '.h
 SourceExtensionPAS','String '.pas
 SourceExtensionDFM','String '.dfm
 SourceExtensionXFM','String '.xfm
 SourceDescriptionPAS','String 'Pascal source file
 SourceDescriptionCPP','String 'C++ source file
 DesignerVCL','String 'VCL
 DesignerCLX','String 'CLX
 ProjectTypePackage','String 'package
 ProjectTypeLibrary','String 'library
 ProjectTypeProgram','String 'program
 Personality32Bit','String '32 bit
 Personality64Bit','String '64 bit
 PersonalityDelphi','String 'Delphi
 PersonalityDelphiDotNet','String 'Delphi.net
 PersonalityBCB','String 'C++Builder
 PersonalityCSB','String 'C#Builder
 PersonalityVB','String 'Visual Basic
 PersonalityDesign','String 'Design
 PersonalityUnknown','String 'Unknown personality
 PersonalityBDS','String 'Borland Developer Studio
 DOFDirectoriesSection','String 'Directories
 DOFUnitOutputDirKey','String 'UnitOutputDir
 DOFSearchPathName','String 'SearchPath
 DOFConditionals','String 'Conditionals
 DOFLinkerSection','String 'Linker
 DOFPackagesKey','String 'Packages
 DOFCompilerSection','String 'Compiler
 DOFPackageNoLinkKey','String 'PackageNoLink
 DOFAdditionalSection','String 'Additional
 DOFOptionsKey','String 'Options
  TJclBorPersonality', '( bpDelphi32, bpDelphi64, bpBCBuilder32, b'
   +'pBCBuilder64, bpDelphiNet32, bpDelphiNet64, bpCSBuilder32, bpCSBuilder64, '
   +'bpVisualBasic32, bpVisualBasic64, bpDesign, bpUnknown )
  TJclBorPersonalities', 'set of TJclBorPersonality
  TJclBorDesigner', '( bdVCL, bdCLX )
  TJclBorDesigners', 'set of TJClBorDesigner
  TJclBorPlatform', '( bp32bit, bp64bit )
  FindClass('TOBJECT'),'TJclBorRADToolInstallation
  SIRegister_TJclBorRADToolInstallationObject(CL);
  SIRegister_TJclBorlandOpenHelp(CL);
  TJclHelp2Object', '( hoRegisterSession, hoRegister, hoPlugin )
  TJclHelp2Objects', 'set of TJclHelp2Object
  SIRegister_TJclHelp2Manager(CL);
  SIRegister_TJclBorRADToolIdeTool(CL);
  SIRegister_TJclBorRADToolIdePackages(CL);
  SIRegister_IJclCommandLineTool(CL);
  FindClass('TOBJECT'),'EJclCommandLineToolError
  SIRegister_TJclCommandLineTool(CL);
  SIRegister_TJclBorlandCommandLineTool(CL);
  SIRegister_TJclBCC32(CL);
  SIRegister_TJclDCC32(CL);
  TJclDCC', 'TJclDCC32
  SIRegister_TJclBpr2Mak(CL);
  SIRegister_TJclBorlandMake(CL);
  SIRegister_TJclBorRADToolPalette(CL);
  SIRegister_TJclBorRADToolRepository(CL);
  TCommandLineTool', '( clAsm, clBcc32, clDcc32, clDccIL, clMake,clProj2Mak )
  TCommandLineTools', 'set of TCommandLineTool
  //TJclBorRADToolInstallationClass', 'class of TJclBorRADToolInstallation
  SIRegister_TJclBorRADToolInstallation(CL);
  SIRegister_TJclBCBInstallation(CL);
  SIRegister_TJclDelphiInstallation(CL);
  SIRegister_TJclDCCIL(CL);
  SIRegister_TJclBDSInstallation(CL);
  TTraverseMethod', 'Function ( Installation : TJclBorRADToolInstallation) : Boolean
  SIRegister_TJclBorRADToolInstallations(CL);
 Function BPLFileName( const BPLPath, PackageFileName : string) : string
 Function BinaryFileName( const OutputPath, ProjectFileName : string) : string
 Function IsDelphiPackage( const FileName : string) : Boolean
 Function IsDelphiProject( const FileName : string) : Boolean
 Function IsBCBPackage( const FileName : string) : Boolean
 Function IsBCBProject( const FileName : string) : Boolean
 Procedure GetDPRFileInfo(const DPRFileName:string;out BinaryExtensio:string;const LibSuffix:PString);
 Procedure GetBPRFileInfo(const BPRFileName:string;out BinaryFileName:string;const Descript:PString);
 Procedure GetDPKFileInfo(const DPKFileName:string;out RunOnly:Bool;const LibSuffix:PString;const Descript:PString;
 Procedure GetBPKFileInfo(const BPKFileName:string;out RunOnly:Bool;const BinaryFName:PString;const Descript:PString
 function SamePath(const Path1, Path2: string): Boolean;
end;

procedure SIRegister_JclFileUtils_max(CL: TPSPascalCompiler);
begin
 'ERROR_NO_MORE_FILES','LongInt'( 18);
 //Function stat64( FileName: PChar;var StatBuffer : TStatBuf64) : Integer
 //Function fstat64( FileDes: Integer;var StatBuffer : TStatBuf64) : Integer
 //Function lstat64( FileName: PChar;var StatBuffer : TStatBuf64) : Integer
 'LPathSeparator','String '/
 'LDirDelimiter','String '/
 'LDirSeparator','String ':
 'JXPathDevicePrefix','String '\\.\
 'JXPathSeparator','String '\
 'JXDirDelimiter','String '\
 'JXDirSeparator','String ';
 'JXPathUncPrefix','String '\\
 'faNormalFile','LongWord')( $00000080);
 //'faUnixSpecific',' faSymLink);
  JXTCompactPath', '( cpCenter, cpEnd )
    _WIN32_FILE_ATTRIBUTE_DATA', 'record dwFileAttributes : DWORD; f'
   +'tCreationTime : TFileTime; ftLastAccessTime : TFileTime; ftLastWriteTime :'
   +' TFileTime; nFileSizeHigh : DWORD; nFileSizeLow : DWORD; end
  TWin32FileAttributeData', '_WIN32_FILE_ATTRIBUTE_DATA
  WIN32_FILE_ATTRIBUTE_DATA', '_WIN32_FILE_ATTRIBUTE_DATA

 Function jxPathAddSeparator( const Path : string) : string
 Function jxPathAddExtension( const Path, Extension : string) : string
 Function jxPathAppend( const Path, Append : string) : string
 Function jxPathBuildRoot( const Drive : Byte) : string
 Function jxPathCanonicalize( const Path : string) : string
 Function jxPathCommonPrefix( const Path1, Path2 : string) : Integer
 Function jxPathCompactPath(const DC:HDC;const Path:string;const Width:Integer;CmpFmt:TCompactPath):string
 Procedure jxPathExtractElements( const Source : string; var Drive, Path, FileName, Ext : string)
 Function jxPathExtractFileDirFixed( const S : string) : string
 Function jxPathExtractFileNameNoExt( const Path : string) : string
 Function jxPathExtractPathDepth( const Path : string; Depth : Integer) : string
 Function jxPathGetDepth( const Path : string) : Integer
 Function jxPathGetLongName( const Path : string) : string
 Function jxPathGetShortName( const Path : string) : string
 Function jxPathGetLongName( const Path : string) : string
 Function jxPathGetShortName( const Path : string) : string
 Function jxPathGetRelativePath( Origin, Destination : string) : string
 Function jxPathGetTempPath : string
 Function jxPathIsAbsolute( const Path : string) : Boolean
 Function jxPathIsChild( const Path, Base : string) : Boolean
 Function jxPathIsDiskDevice( const Path : string) : Boolean
 Function jxPathIsUNC( const Path : string) : Boolean
 Function jxPathRemoveSeparator( const Path : string) : string
 Function jxPathRemoveExtension( const Path : string) : string
 Function jxPathGetPhysicalPath( const LocalizedPath : string) : string
 Function jxPathGetLocalizedPath( const PhysicalPath : string) : string
  JxTFileListOption', '( flFullNames, flRecursive, flMaskedSubfolders)
  JxTFileListOptions', 'set of TFileListOption
  JxTJclAttributeMatch', '( amAny, amExact, amSubSetOf, amSuperSetOf, amCustom )
  TFileHandler', 'Procedure ( const FileName : string)
  TFileHandlerEx', 'Procedure ( const Directory : string; const FileInfo : TSearchRec)
 Function BuildFileList( const Path : string; const Attr : Integer; const List : TStrings) : Boolean
 //Function AdvBuildFileList( const Path : string; const Attr : Integer; const Files : TStrings; const
AttributeMatch:TJclAttributeMatch;const Optis:TFileListOptions;const SubfoldersMask:string;const
FileMatchFunc:TFileMatchFunc):Bool;
 Function jxVerifyFileAttributeMask( var RejectedAttributes, RequiredAttributes : Int) : Boolean
 Function jxIsFileAttributeMatch(FileAttributes,RejectedAttributes,RequiredAttributes:Int):Boolean;
 Function jxFileAttributesStr( const FileInfo : TSearchRec) : string
 Function jxIsFileNameMatch(FileName:string;const Mask:string;const CaseSensitive:Boolean):Boolean;
 Procedure jxEnumFiles(const Path:string; HandleFile:TFileHandlerEx;
RejectedAttributes:Integer;RequiredAttributes : Integer; Abort : TObject)
 Procedure jxEnumDirectories(const Root:string;const HandleDirectory:TFileHandler;const
IncludeHiddenDirects:Boolean;const SubDirectoriesMask:string;Abort:TObject;ResolveSymLinks:Bool)
 Procedure jxCreateEmptyFile( const FileName : string)
 Function jxCloseVolume( var Volume : THandle) : Boolean
 Function jxDeleteDirectory( const DirectoryName : string; MoveToRecycleBin : Boolean) : Boolean
 Function jxCopyDirectory( ExistingDirectoryName, NewDirectoryName : string) : Boolean
 Function jxMoveDirectory( ExistingDirectoryName, NewDirectoryName : string) : Boolean
 Function jxDelTree( const Path : string) : Boolean
 //Function DelTreeEx(const Path:string;AbortOnFailure:Boolean; Progress:TDelTreeProgress):Boolean
 Function jxDiskInDrive( Drive : Char) : Boolean
 Function jxDirectoryExists( const Name : string; ResolveSymLinks : Boolean) : Boolean
 Function jxFileCreateTemp( var Prefix : string) : THandle
 Function jxFileBackup( const FileName : string; Move : Boolean) : Boolean
 Function jxFileCopy( const ExistingFileName, NewFileName : string; ReplaceExisting : Boolean) : Boolean
 Function jxFileDelete( const FileName : string; MoveToRecycleBin : Boolean) : Boolean
 Function jxFileExists( const FileName : string) : Boolean
 Function jxFileMove( const ExistingFileName, NewFileName : string; ReplaceExisting : Boolean) : Boolean
 Function jxFileRestore( const FileName : string) : Boolean
 Function jxGetBackupFileName( const FileName : string) : string
 Function jxIsBackupFileName( const FileName : string) : Boolean
 Function jxFileGetDisplayName( const FileName : string) : string
 Function jxFileGetGroupName( const FileName : string; ResolveSymLinks : Boolean) : string
 Function jxFileGetOwnerName( const FileName : string; ResolveSymLinks : Boolean) : string
 Function jxFileGetSize( const FileName : string) : Int64
 Function jxFileGetTempName( const Prefix : string) : string
 Function jxFileGetTypeName( const FileName : string) : string
 Function jxFindUnusedFileName(FileName:string; const FileExt : string; NumberPrefix : string) : string
 Function jxForceDirectories( Name : string) : Boolean
 Function jxGetDirectorySize( const Path : string) : Int64
 Function jxGetDriveTypeStr( const Drive : Char) : string
 Function jxGetFileAgeCoherence( const FileName : string) : Boolean
 Procedure jxGetFileAttributeList( const Items : TStrings; const Attr : Integer)
 Procedure jxGetFileAttributeListEx( const Items : TStrings; const Attr : Integer)
 Function jxGetFileInformation( const FileName : string; out FileInfo : TSearchRec) : Boolean;
 Function jxGetFileInformation1( const FileName : string) : TSearchRec;
 //Function GetFileStatus(const FileName:string;out StatBuf:TStatBuf64;const
ResolveSymLinks:Boolean):Integer
 Function jxGetFileLastWrite( const FName : string) : TFileTime;
 Function jxGetFileLastWrite1( const FName : string; out LocalTime : TDateTime) : Boolean;
 Function jxGetFileLastAccess( const FName : string) : TFileTime;
 Function jxGetFileLastAccess1( const FName : string; out LocalTime : TDateTime) : Boolean;
 Function jxGetFileCreation( const FName : string) : TFileTime;
 Function jxGetFileCreation1( const FName : string; out LocalTime : TDateTime) : Boolean;
 Function jxGetFileLastWrite( const FName : string;out TimeStamp:Integer;ResolveSymLinks : Bool):Bool;
 Function jxGetFileLastWrite1(const FName:string; out LocalTime:TDateTime;ResolveSymLinks:Bool): Bool;
 Function jxGetFileLastWrite2( const FName : string; ResolveSymLinks : Boolean) : Integer;
 Function jxGetFileLastAccess(const FName:string; out TimeStamp:Integer;ResolveSymLinks: Bool): Bool;
 Function jxGetFileLastAccess1(const FName:string; out LocalTime:TDateTime;ResolveSymLinks:Bool):Bool;
 Function jxGetFileLastAccess2(const FName:string; ResolveSymLinks:Boolean): Integer;
 Function jxGetFileLastAttrChange(const FName:string;out TimeStamp:Integer;ResolveSymLinks:Bool): Bool;
 Function jxGetFileLastAttrChange1(const FName:string; out LocalTime:TDateTime;ResolveSymLinks:Bool):Bool;
 Function jxGetFileLastAttrChange2( const FName : string; ResolveSymLinks:Boolean): Integer;
 Function jxGetModulePath( const Module : HMODULE) : string
 Function jxGetSizeOfFile( const FileName : string) : Int64;
 Function jxGetSizeOfFile1( const FileInfo : TSearchRec) : Int64;
 Function jxGetSizeOfFile2( Handle : THandle) : Int64;
 Function jxGetStandardFileInfo( const FileName : string) : TWin32FileAttributeData
 Function jxIsDirectory( const FileName : string; ResolveSymLinks : Boolean) : Boolean
 Function jxIsRootDirectory( const CanonicFileName : string) : Boolean
 Function jxLockVolume( const Volume : string; var Handle : THandle) : Boolean
 Function jxOpenVolume( const Drive : Char) : THandle
 Function jxSetDirLastWrite( const DirName : string; const DateTime : TDateTime) : Boolean
 Function jxSetDirLastAccess( const DirName : string; const DateTime : TDateTime) : Boolean
 Function jxSetDirCreation( const DirName : string; const DateTime : TDateTime) : Boolean
 Function jxSetFileLastWrite( const FileName : string; const DateTime : TDateTime) : Boolean
 Function jxSetFileLastAccess( const FileName : string; const DateTime : TDateTime) : Boolean
 Function jxSetFileCreation( const FileName : string; const DateTime : TDateTime) : Boolean
 Procedure jxShredFile( const FileName : string; Times : Integer)
 Function jxUnlockVolume( var Handle : THandle) : Boolean
 Function jxCreateSymbolicLink( const Name, Target : string) : Boolean
 Function jxSymbolicLinkTarget( const Name : string) : string
  TAttributeInterest', '( aiIgnored, aiRejected, aiRequired )
  SIRegister_TJclCustomFileAttrMask(CL);
  SIRegister_TJclFileAttributeMask(CL);
  TFileSearchOption', '( fsIncludeSubDirectories, fsIncludeHiddenS'
   +'ubDirectories, fsLastChangeAfter, fsLastChangeBefore, fsMaxSize, fsMinSize)
  TFileSearchOptions', 'set of TFileSearchOption
  TFileSearchTaskID', 'Integer
  TFileSearchTerminationEvent', 'Procedure ( const ID : TFileSearc'
   +'hTaskID; const Aborted : Boolean)
  TFileEnumeratorSyncMode', '( smPerFile, smPerDirectory )
  SIRegister_IJclFileEnumerator(CL);
  SIRegister_TJclFileEnumerator(CL);
 Function JxFileSearch : IJclFileEnumerator
  JxTFileFlag', '( ffDebug, ffInfoInferred, ffPatched,ffPreRelease,ffPrivateBuild, ffSpecialBuild )
  JxTFileFlags', 'set of TFileFlag
  FindClass('TOBJECT'),'EJclFileVersionInfoError
  SIRegister_TJclFileVersionInfo(CL);
 Function jxOSIdentToString( const OSIdent : DWORD) : string
 Function jxOSFileTypeToString( const OSFileType : DWORD; const OSFileSubType : DWORD) : string
 Function jxVersionResourceAvailable( const FileName : string) : Boolean
  TFileVersionFormat', '( vfMajorMinor, vfFull )
 Function jxFormatVersionString( const HiV, LoV : Word) : string;
 Function jxFormatVersionString1( const Major, Minor, Build, Revision : Word) : string;
 //Function FormatVersionString2( const FixedInfo : TVSFixedFileInfo;VersionFormat:TFileVersionFormat):str;
 //Procedure VersionExtractFileInfo(const FixedInfo:TVSFixedFileInfo;var Major,Minor,Build,Revision:Word);
 //Procedure VersionExtractProductInfo(const FixedInfo:TVSFixedFileInfo;var Major,Minor,Build, Revision:Word);
 //Function VersionFixedFileInfo( const FileName : string; var FixedInfo : TVSFixedFileInfo) : Boolean
 Function jxVersionFixedFileInfoString( const FileName : string; VersionFormat : TFileVersionFormat; const NotAvailableText : string) : string
  SIRegister_TJclTempFileStream(CL);
  FindClass('TOBJECT'),'TJclCustomFileMapping
  SIRegister_TJclFileMappingView(CL);
  TJclFileMappingRoundOffset', '( rvDown, rvUp )
  SIRegister_TJclCustomFileMapping(CL);
  SIRegister_TJclFileMapping(CL);
  SIRegister_TJclSwapFileMapping(CL);
  SIRegister_TJclFileMappingStream(CL);
  TJclMappedTextReaderIndex', '( tiNoIndex, tiFull )
  //PPCharArray', '^TPCharArray // will not work
  SIRegister_TJclMappedTextReader(CL);
  SIRegister_TJclFileMaskComparator(CL);
  FindClass('TOBJECT'),'EJclPathError
  FindClass('TOBJECT'),'EJclFileUtilsError
  FindClass('TOBJECT'),'EJclTempFileStreamError
  FindClass('TOBJECT'),'EJclTempFileStreamError
  FindClass('TOBJECT'),'EJclFileMappingError
  FindClass('TOBJECT'),'EJclFileMappingViewError
 Function jxPathGetLongName2( const Path : string) : string
 Function jxWin32DeleteFile( const FileName : string; MoveToRecycleBin : Boolean) : Boolean
 Function jxWin32MoveFileReplaceExisting( const SrcFileName, DstFileName : string) : Boolean
 Function jxWin32BackupFile( const FileName : string; Move : Boolean) : Boolean
 Function jxWin32RestoreFile( const FileName : string) : Boolean
 Function jxSamePath( const Path1, Path2 : string) : Boolean
 Procedure jxPathListAddItems( var List : string; const Items : string)
 Procedure jxPathListIncludeItems( var List : string; const Items : string)
 Procedure jxPathListDelItems( var List : string; const Items : string)
 Procedure jxPathListDelItem( var List : string; const Index : Integer)
 Function jxPathListItemCount( const List : string) : Integer
 Function jxPathListGetItem( const List : string; const Index : Integer) : string
 Procedure jxPathListSetItem( var List : string; const Index : Integer; const Value : string)
 Function jxPathListItemIndex( const List, Item : string) : Integer
 Function jxParamName(Idx:Int;const Separator:string;const AllowedPrefixChars:string;TrimName:Bool):string;
 Function jxParamValue(Index : Integer; const Separator : string; TrimValue : Boolean) : string;
 Function jxParamValue1(const SearchName:string; const Separator : string; CaseSensitive : Boolean; const AllowedPrefixCharacters : string; TrimValue : Boolean) : string;
 Function jxParamPos( const SearchName : string; const Separator : string; CaseSensitive : Boolean; const AllowedPrefixCharacters : string) : Integer
end;

procedure SIRegister_FileUtil(CL: TPSPascalCompiler);
begin
 'UTF8FileHeader','String #$ef#$bb#$bf);
 Function lCompareFilenames( const Filename1, Filename2 : string) : integer
 Function lCompareFilenamesIgnoreCase( const Filename1, Filename2 : string) : integer
 Function lCompareFilenames( const Filename1, Filename2 : string; ResolveLinks : boolean) : integer
 Function lCompareFilenames(Filename1:PChar;Len1:int;Filename2:PChar;Len2:int;ResolveLinks:boolean):int;
 Function lFilenameIsAbsolute( const TheFilename : string) : boolean
 Function lFilenameIsWinAbsolute( const TheFilename : string) : boolean
 Function lFilenameIsUnixAbsolute( const TheFilename : string) : boolean
 Procedure lCheckIfFileIsExecutable( const AFilename : string)
 Procedure lCheckIfFileIsSymlink( const AFilename : string)
 Function lFileIsReadable( const AFilename : string) : boolean
 Function lFileIsWritable( const AFilename : string) : boolean
 Function lFileIsText( const AFilename : string) : boolean
 Function lFileIsText( const AFilename : string; out FileReadable : boolean) : boolean
 Function lFileIsExecutable( const AFilename : string) : boolean
 Function lFileIsSymlink( const AFilename : string) : boolean
 Function lFileIsHardLink( const AFilename : string) : boolean
 Function lFileSize( const Filename : string) : int64;
 Function lGetFileDescription( const AFilename : string) : string
 Function lReadAllLinks( const Filename : string; ExceptionOnError : boolean) : string
 Function lTryReadAllLinks( const Filename : string) : string
 Function lDirPathExists( const FileName : String) : Boolean
 Function lForceDirectory( DirectoryName : string) : boolean
 Function lDeleteDirectory( const DirectoryName : string; OnlyChildren : boolean) : boolean
 Function lProgramDirectory : string
 Function lDirectoryIsWritable( const DirectoryName : string) : boolean
 Function lExtractFileNameOnly( const AFilename : string) : string
 Function lExtractFileNameWithoutExt( const AFilename : string) : string
 Function lCompareFileExt( const Filename, Ext : string; CaseSensitive : boolean) : integer;
 Function lCompareFileExt( const Filename, Ext : string) : integer;
 Function lFilenameIsPascalUnit( const Filename : string) : boolean
 Function lAppendPathDelim( const Path : string) : string
 Function lChompPathDelim( const Path : string) : string
 Function lTrimFilename( const AFilename : string) : string
 Function lCleanAndExpandFilename( const Filename : string) : string
 Function lCleanAndExpandDirectory( const Filename : string) : string
 Function lCreateAbsoluteSearchPath( const SearchPath, BaseDirectory : string) : string
 Function lCreateRelativePath( const Filename, BaseDirectory : string; UsePointDirectory : boolean; AlwaysRequireSharedBaseFolder : Boolean) : string
 Function lCreateAbsolutePath( const Filename, BaseDirectory : string) : string
 Function lFileIsInPath( const Filename, Path : string) : boolean
 Function lFileIsInDirectory( const Filename, Directory : string) : boolean
  TSearchFileInPathFlag', '( sffDontSearchInBasePath, sffSearchLoUpCase )
  TSearchFileInPathFlags', 'set of TSearchFileInPathFlag
 'AllDirectoryEntriesMask','String '*
 Function lGetAllFilesMask : string
 Function lGetExeExt : string
 Function lSearchFileInPath( const Filename, BasePath, SearchPath, Delimiter : string; Flags : TSearchFileInPathFlags) : string
 Function lSearchAllFilesInPath( const Filename, BasePath, SearchPath, Delimiter:string;Flags : TSearchFileInPathFlags) : TStrings
 Function lFindDiskFilename( const Filename : string) : string
 Function lFindDiskFileCaseInsensitive( const Filename : string) : string
 Function lFindDefaultExecutablePath( const Executable : string; const BaseDir: string):string
 Function lGetDarwinSystemFilename( Filename : string) : string
  SIRegister_TFileIterator(CL);
  TFileFoundEvent', 'Procedure ( FileIterator : TFileIterator)
  TDirectoryFoundEvent', 'Procedure ( FileIterator : TFileIterator)
  TDirectoryEnterEvent', 'Procedure ( FileIterator : TFileIterator)
  SIRegister_TFileSearcher(CL);
 Function lFindAllFiles(const SearchPath:String;SearchMsk:String;SearchSubDirs:Bool):TStringList
 Function lFindAllDirectories( const SearchPath : string; SearchSubDirs : Boolean) : TStringList
 // TCopyFileFlag', '( cffOverwriteFile, cffCreateDestDirectory, cffPreserveTime )
 // TCopyFileFlags', 'set of TCopyFileFlag
 Function lCopyFile( const SrcFilename, DestFilename : string; Flags : TCopyFileFlags) : boolean
 Function lCopyFile( const SrcFilename, DestFilename : string; PreserveTime : boolean) : boolean
 Function lCopyDirTree( const SourceDir, TargetDir : string; Flags : TCopyFileFlags) : Boolean
 Function lReadFileToString( const Filename : string) : string
 Function lGetTempFilename( const Directory, Prefix : string) : string
 {Function NeedRTLAnsi : boolean
 Procedure SetNeedRTLAnsi( NewValue : boolean)
 Function UTF8ToSys( const s : string) : string
 Function SysToUTF8( const s : string) : string
 Function ConsoleToUTF8( const s : string) : string
 Function UTF8ToConsole( const s : string) : string}
 Function FileExistsUTF8( const Filename : string) : boolean
 Function FileAgeUTF8( const FileName : string) : Longint
 Function DirectoryExistsUTF8( const Directory : string) : Boolean
 Function ExpandFileNameUTF8( const FileName : string) : string
 Function ExpandUNCFileNameUTF8( const FileName : string) : string
 Function ExtractShortPathNameUTF8( const FileName : String) : String
 Function FindFirstUTF8(const Path: string; Attr : Longint; out Rslt : TSearchRec) : Longint
 Function FindNextUTF8( var Rslt : TSearchRec) : Longint
 Procedure FindCloseUTF8( var F : TSearchrec)
 Function FileSetDateUTF8( const FileName : String; Age : Longint) : Longint
 Function FileGetAttrUTF8( const FileName : String) : Longint
 Function FileSetAttrUTF8( const Filename : String; Attr : longint) : Longint
 Function DeleteFileUTF8( const FileName : String) : Boolean
 Function RenameFileUTF8( const OldName, NewName : String) : Boolean
 Function FileSearchUTF8( const Name, DirList : String; ImplicitCurrentDir : Boolean) : String
 Function FileIsReadOnlyUTF8( const FileName : String) : Boolean
```

```
  Function GetCurrentDirUTF8 : String
  Function SetCurrentDirUTF8( const NewDir : String) : Boolean
  Function CreateDirUTF8( const NewDir : String) : Boolean
  Function RemoveDirUTF8( const Dir : String) : Boolean
  Function ForceDirectoriesUTF8( const Dir : string) : Boolean
  Function FileOpenUTF8( const FileName : string; Mode : Integer) : THandle
  Function FileCreateUTF8( const FileName : string) : THandle;
  Function FileCreateUTF81( const FileName : string; Rights : Cardinal) : THandle;
  Function ParamStrUTF8( Param : Integer) : string
  Function GetEnvironmentStringUTF8( Index : Integer) : string
  Function GetEnvironmentVariableUTF8( const EnvVar : string) : String
  Function GetAppConfigDirUTF8( Global : Boolean; Create : boolean) : string
  Function GetAppConfigFileUTF8(Global:Boolean; SubDir:boolean; CreateDir : boolean) : string
  Function SysErrorMessageUTF8( ErrorCode : Integer) : String
end;

procedure SIRegister_Keyboard(CL: TPSPascalCompiler);
begin
   //VK_F23 = 134;
   //{$EXTERNALSYM VK_F24}
   //VK_F24 = 135;
  TVirtualKeyCode', 'Integer
  'VK_MOUSEWHEELUP','integer'(134);
  'VK_MOUSEWHEELDOWN','integer'(135);
  Function glIsKeyDown( c : Char) : Boolean;
  Function glIsKeyDown1( vk : TVirtualKeyCode) : Boolean;
  Function glKeyPressed( minVkCode : TVirtualKeyCode) : TVirtualKeyCode
  Function glVirtualKeyCodeToKeyName( vk : TVirtualKeyCode) : String
  Function glKeyNameToVirtualKeyCode( const keyName : String) : TVirtualKeyCode
  Function glCharToVirtualKeyCode( c : Char) : TVirtualKeyCode
  Procedure glKeyboardNotifyWheelMoved( wheelDelta : Integer)
end;

procedure SIRegister_GLCrossPlatform(CL: TPSPascalCompiler);
begin
   TGLPoint', 'TPoint
   //PGLPoint', '^TGLPoint // will not work
   TGLRect', 'TRect
   //PGLRect', '^TGLRect // will not work
   TDelphiColor', 'TColor
   TGLPicture', 'TPicture
   TGLGraphic', 'TGraphic
   TGLBitmap', 'TBitmap
   //TGraphicClass', 'class of TGraphic
   TGLTextLayout', '( tlTop, tlCenter, tlBottom )
   TGLMouseButton', '( mbLeft, mbRight, mbMiddle )
   TGLMouseEvent', 'Procedure ( Sender : TObject; Button : TGLMouse'
    +'Button; Shift : TShiftState; X, Y : Integer)
   TGLMouseMoveEvent', 'TMouseMoveEvent
   TGLKeyEvent', 'TKeyEvent
   TGLKeyPressEvent', 'TKeyPressEvent
   EGLOSError', 'EWin32Error
   EGLOSError', 'EWin32Error
   EGLOSError', 'EOSError
  'glsAllFilter','string'All // sAllFilter
  Function GLPoint( const x, y : Integer) : TGLPoint
  Function GLRGB( const r, g, b : Byte) : TColor
  Function GLColorToRGB( color : TColor) : TColor
  Function GLGetRValue( rgb : DWORD) : Byte
  Function GLGetGValue( rgb : DWORD) : Byte
  Function GLGetBValue( rgb : DWORD) : Byte
  Procedure GLInitWinColors
  Function GLRect( const aLeft, aTop, aRight, aBottom : Integer) : TGLRect
  Procedure GLInflateGLRect( var aRect : TGLRect; dx, dy : Integer)
  Procedure GLIntersectGLRect( var aRect : TGLRect; const rect2 : TGLRect)
  Procedure GLInformationDlg( const msg : String)
  Function GLQuestionDlg( const msg : String) : Boolean
  Function GLInputDlg( const aCaption, aPrompt, aDefault : String) : String
  Function GLSavePictureDialog( var aFileName : String; const aTitle : String) : Boolean
  Function GLOpenPictureDialog( var aFileName : String; const aTitle : String) : Boolean
  Function GLApplicationTerminated : Boolean
  Procedure GLRaiseLastOSError
  Procedure GLFreeAndNil( var anObject: TObject)
  Function GLGetDeviceLogicalPixelsX( device : Cardinal) : Integer
  Function GLGetCurrentColorDepth : Integer
  Function GLPixelFormatToColorBits( aPixelFormat : TPixelFormat) : Integer
  Function GLBitmapScanLine( aBitmap : TGLBitmap; aRow : Integer) : Pointer
  Procedure GLSleep( length : Cardinal)
  Procedure GLQueryPerformanceCounter( var val : Int64)
  Function GLQueryPerformanceFrequency( var val : Int64) : Boolean
  Function GLStartPrecisionTimer : Int64
  Function GLPrecisionTimerLap( const precisionTimer : Int64) : Double
  Function GLStopPrecisionTimer( const precisionTimer : Int64) : Double
  Function GLRDTSC : Int64
  Procedure GLLoadBitmapFromInstance( ABitmap : TBitmap; AName : string)
  Function GLOKMessageBox( const Text, Caption : string) : Integer
  Procedure GLShowHTMLUrl( Url : String)
  Procedure GLShowCursor( AShow : boolean)
  Procedure GLSetCursorPos( AScreenX, AScreenY : integer)
```

```
 Procedure GLGetCursorPos( var point : TGLPoint)
 Function GLGetScreenWidth : integer
 Function GLGetScreenHeight : integer
 Function GLGetTickCount : int64
 function RemoveSpaces(const str : String) : String;
   TNormalMapSpace', '( nmsObject, nmsTangent )
 Procedure CalcObjectSpaceLightVectors(Light:TAffineVector;Vertices TAffineVectorList;Colors:TVectorList)
 Procedure SetupTangentSpace( Vertices, Normals, TexCoords, Tangents, BiNormals : TAffineVectorList)
 Procedure CalcTangentSpaceLightVectors( Light : TAffineVector; Vertices, Normals, Tangents, BiNormals : TAffineVectorList; Colors : TVectorList)
 Function CreateObjectSpaceNormalMap( Width,Height:Integer;HiNormals, HiTexCoords:TAffineVectorList):TGLBitmap
 Function CreateTangentSpaceNormalMap( Width, Height : Integer; HiNormals, HiTexCoords, LoNormals, LoTexCoords, Tangents, BiNormals : TAffineVectorList) : TGLBitmap
end;

procedure SIRegister_GLStarRecord(CL: TPSPascalCompiler);
begin
  TGLStarRecord', 'record RA: Word; DEC: SmallInt; BVColorIndex: Byte; VMagnitude: Byte; end
 // PGLStarRecord', '^TGLStarRecord // will not work
 Function StarRecordPositionZUp( const starRecord : TGLStarRecord) : TAffineVector
 Function StarRecordPositionYUp( const starRecord : TGLStarRecord) : TAffineVector
 Function StarRecordColor( const starRecord : TGLStarRecord; bias : Single) : TVector
end;


procedure SIRegister_GeometryBB(CL: TPSPascalCompiler);
begin
  TAABB', 'record min : TAffineVector; max : TAffineVector; end
  //PAABB', '^TAABB // will not work
 TBSphere', 'record Center : TAffineVector; Radius : single; end
 TClipRect', 'record Left : Single; Top:Single; Right:Single; Bottom : Single; end
 TSpaceContains', '(scNoOverlap, scContainsFully, scContainsPartially )
 Function AddBB( var c1 : THmgBoundingBox; const c2 : THmgBoundingBox) : THmgBoundingBox
 Procedure AddAABB( var aabb : TAABB; const aabb1 : TAABB)
 Procedure SetBB( var c : THmgBoundingBox; const v : TVector)
 Procedure SetAABB( var bb : TAABB; const v : TVector)
 Procedure BBTransform( var c : THmgBoundingBox; const m : TMatrix)
 Procedure AABBTransform( var bb : TAABB; const m : TMatrix)
 Procedure AABBScale( var bb : TAABB; const v : TAffineVector)
 Function BBMinX( const c : THmgBoundingBox) : Single
 Function BBMaxX( const c : THmgBoundingBox) : Single
 Function BBMinY( const c : THmgBoundingBox) : Single
 Function BBMaxY( const c : THmgBoundingBox) : Single
 Function BBMinZ( const c : THmgBoundingBox) : Single
 Function BBMaxZ( const c : THmgBoundingBox) : Single
 Procedure AABBInclude( var bb : TAABB; const p : TAffineVector)
 Procedure AABBFromSweep( var SweepAABB : TAABB; const Start, Dest : TVector; const Radius : Single)
 Function AABBIntersection( const aabb1, aabb2 : TAABB) : TAABB
 Function BBToAABB( const aBB : THmgBoundingBox) : TAABB
 Function AABBToBB( const anAABB : TAABB) : THmgBoundingBox;
 Function AABBToBB1( const anAABB : TAABB; const m : TMatrix) : THmgBoundingBox;
 Procedure OffsetAABB( var aabb : TAABB; const delta : TAffineVector);
 Procedure OffsetAABB1( var aabb : TAABB; const delta : TVector);
 Function IntersectAABBs( const aabb1, aabb2 : TAABB; const m1To2, m2To1 : TMatrix) : Boolean;
 Function IntersectAABBsAbsoluteXY( const aabb1, aabb2 : TAABB) : Boolean
 Function IntersectAABBsAbsoluteXZ( const aabb1, aabb2 : TAABB) : Boolean
 Function IntersectAABBsAbsolute( const aabb1, aabb2 : TAABB) : Boolean
 Function AABBFitsInAABBAbsolute( const aabb1, aabb2 : TAABB) : Boolean
 Function PointInAABB( const p : TAffineVector; const aabb : TAABB) : Boolean;
 Function PointInAABB1( const p : TVector; const aabb : TAABB) : Boolean;
 Function PlaneIntersectAABB( Normal : TAffineVector; d : single; aabb : TAABB) : boolean
 Function TriangleIntersectAABB( const aabb : TAABB; v1, v2, v3 : TAffineVector) : boolean
 Procedure ExtractAABBCorners( const AABB : TAABB; var AABBCorners : TAABBCorners)
 Procedure AABBToBSphere( const AABB : TAABB; var BSphere : TBSphere)
 Procedure BSphereToAABB( const BSphere : TBSphere; var AABB : TAABB);
 Function BSphereToAABB1( const center : TAffineVector; radius : Single) : TAABB;
 Function BSphereToAABB2( const center : TVector; radius : Single) : TAABB;
 Function AABBContainsAABB( const mainAABB, testAABB : TAABB) : TSpaceContains
 Function BSphereContainsAABB( const mainBSphere : TBSphere; const testAABB : TAABB) : TSpaceContains
 Function BSphereContainsBSphere( const mainBSphere, testBSphere : TBSphere) : TSpaceContains
 Function AABBContainsBSphere( const mainAABB : TAABB; const testBSphere : TBSphere) : TSpaceContains
 Function PlaneContainsBSphere(const Location,Normal:TAffineVector;const testBSphere:TBSphere):TSpaceContains
 Function FrustumContainsBSphere( const Frustum : TFrustum; const testBSphere : TBSphere) : TSpaceContains
 Function FrustumContainsAABB( const Frustum : TFrustum; const testAABB : TAABB) : TSpaceContains
 Function ClipToAABB( const v : TAffineVector; const AABB : TAABB) : TAffineVector
 Function BSphereIntersectsBSphere( const mainBSphere, testBSphere : TBSphere) : boolean
 Procedure IncludeInClipRect( var clipRect : TClipRect; x, y : Single)
 Function AABBToClipRect(const aabb:TAABB;modelViewProjection:TMatrix;viewportSizeX, viewportSizeY:Int):TClipRect
end;

procedure SIRegister_GeometryCoordinates(CL: TPSPascalCompiler);
begin
 Procedure Cylindrical_Cartesian( const r, theta, z1 : single; var x, y, z : single);
 Procedure Cylindrical_Cartesian1( const r, theta, z1 : double; var x, y, z : double);
 Procedure Cylindrical_Cartesian2( const r,theta,z1 : single; var x, y, z : single; var ierr : integer);
 Procedure Cylindrical_Cartesian3( const r,theta,z1 : double; var x, y, z : double; var ierr : integer);
 Procedure Cartesian_Cylindrical( const x, y, z1 : single; var r, theta, z : single);
 Procedure Cartesian_Cylindrical1( const x, y, z1 : double; var r, theta, z : double);
 Procedure Spherical_Cartesian( const r, theta, phi : single; var x, y, z : single);
 Procedure Spherical_Cartesian1( const r, theta, phi : double; var x, y, z : double);
 Procedure Spherical_Cartesian2( const r,theta, phi : single; var x, y, z : single; var ierr : integer);
 Procedure Spherical_Cartesian3( const r,theta, phi : double; var x, y, z : double; var ierr : integer);
 Procedure Cartesian_Spherical( const x, y, z : single; var r, theta, phi : single);
 Procedure Cartesian_Spherical1( const v : TAffineVector; var r, theta, phi : Single);
 Procedure Cartesian_Spherical2( const x, y, z : double; var r, theta, phi : double);
 Procedure ProlateSpheroidal_Cartesian( const xi, eta, phi, a : single; var x, y, z : single);
 Procedure ProlateSpheroidal_Cartesian1( const xi, eta, phi, a : double; var x, y, z : double);
 Procedure ProlateSpheroidal_Cartesian2(const xi,eta,phi,a:single;var x,y,z:single;var ierr: integer);
 Procedure ProlateSpheroidal_Cartesian3(const xi,eta,phi,a:double;var x,y,z:double;var ierr: integer);
 Procedure OblateSpheroidal_Cartesian( const xi, eta, phi, a : single; var x, y, z : single);
 Procedure OblateSpheroidal_Cartesian1( const xi, eta, phi, a : double; var x, y, z : double);
 Procedure OblateSpheroidal_Cartesian2(const xi,eta,phi,a:single; var x,y,z: single;var ierr:integer);
 Procedure OblateSpheroidal_Cartesian3( const xi,eta,phi,a:double; var x,y,z:double;var ierr:integer);
 Procedure BipolarCylindrical_Cartesian( const u, v, z1, a : single; var x, y, z : single);
 Procedure BipolarCylindrical_Cartesian1(const u, v, z1, a : double; var x, y, z : double);
 Procedure BipolarCylindrical_Cartesian2(const u,v,z1,a: single;var x,y,z:single; var ierr : integer);
 Procedure BipolarCylindrical_Cartesian3(const u,v,z1,a: double;var x,y,z:double; var ierr : integer);
end;

procedure SIRegister_VectorGeometry(CL: TPSPascalCompiler);
begin
 'EPSILON','Single').setExtended( 1e-40);
 'EPSILON2','Single').setExtended( 1e-30);  }
 TRenderContextClippingInfo', 'record origin : TVector; clippingD'
   +'irection : TVector; viewPortRadius : Single; farClippingDistance:Single;frustum:TFrustum; end
 THmgPlane', 'TVector
  TDoubleHmgPlane', 'THomogeneousDblVector
 {TTransType', '( ttScaleX, ttScaleY, ttScaleZ, ttShearXY, ttShear'
   +'XZ, ttShearYZ, ttRotateX, ttRotateY, ttRotateZ, ttTranslateX, ttTranslateY'
   +', ttTranslateZ, ttPerspectiveX, ttPerspectiveY, ttPerspectiveZ, ttPerspectiveW )}
  TSingleArray', 'array of Single
  TTransformations','array [0..15] of Single)
  TPackedRotationMatrix','array [0..2] of Smallint)
  TVertex', 'TAffineVector
  //TVectorGL', 'THomogeneousFltVector
  //TMatrixGL', 'THomogeneousFltMatrix
  //  TPackedRotationMatrix = array [0..2] of SmallInt;
 Function glTexPointMake( const s, t : Single) : TTexPoint
 Function glAffineVectorMake( const x, y, z : Single) : TAffineVector;
 Function glAffineVectorMake1( const v : TVectorGL) : TAffineVector;
 Procedure glSetAffineVector( var v : TAffineVector; const x, y, z : Single);
 Procedure glSetVector( var v : TAffineVector; const x, y, z : Single);
 Procedure glSetVector1( var v : TAffineVector; const vSrc : TVectorGL);
 Procedure glSetVector2( var v : TAffineVector; const vSrc : TAffineVector);
 Procedure glSetVector3( var v : TAffineDblVector; const vSrc : TAffineVector);
 Procedure glSetVector4( var v : TAffineDblVector; const vSrc : TVectorGL);
 Function glVectorMake( const v : TAffineVector; w : Single) : TVectorGL;
 Function glVectorMake1( const x, y, z : Single; w : Single) : TVectorGL;
 Function glPointMake( const x, y, z : Single) : TVectorGL;
 Function glPointMake1( const v : TAffineVector) : TVectorGL;
 Function glPointMake2( const v : TVectorGL) : TVectorGL;
 Procedure glSetVector5( var v : TVectorGL; const x, y, z : Single; w : Single);
 Procedure glSetVector6( var v : TVectorGL; const av : TAffineVector; w : Single);
 Procedure glglSetVector7( var v : TVectorGL; const vSrc : TVectorGL);
 Procedure glMakePoint( var v : TVectorGL; const x, y, z : Single);
 Procedure glMakePoint1( var v : TVectorGL; const av : TAffineVector);
 Procedure glMakePoint2( var v : TVectorGL; const av : TVectorGL);
 Procedure glMakeVector( var v : TAffineVector; const x, y, z : Single);
 Procedure glMakeVector1( var v : TVectorGL; const x, y, z : Single);
 Procedure glMakeVector2( var v : TVectorGL; const av : TAffineVector);
 Procedure glMakeVector3( var v : TVectorGL; const av : TVectorGL);
 Procedure glRstVector( var v : TAffineVector);
 Procedure glRstVector1( var v : TVectorGL);
 Function glVectorAdd( const v1, v2 : TAffineVector) : TAffineVector;
 Procedure glVectorAdd1( const v1, v2 : TAffineVector; var vr : TAffineVector);
 //Procedure VectorAdd2( const v1, v2 : TAffineVector; vr : PAffineVector);
 Function glVectorAdd3( const v1, v2 : TVectorGL) : TVectorGL;
 Procedure glVectorAdd4( const v1, v2 : TVectorGL; var vr : TVectorGL);
 Function glVectorAdd5( const v : TAffineVector; const f : Single) : TAffineVector;
 Function glVectorAdd6( const v : TVectorGL; const f : Single) : TVectorGL;
 Procedure glAddVector7( var v1 : TAffineVector; const v2 : TAffineVector);
 Procedure glAddVector8( var v1 : TAffineVector; const v2 : TVectorGL);
 Procedure glAddVector9( var v1 : TVectorGL; const v2 : TVectorGL);
 Procedure glAddVector10( var v : TAffineVector; const f : Single);
 Procedure glAddVector11( var v : TVectorGL; const f : Single);
 //Procedure TexPointArrayAdd(const src:PTexPointArray;const delta:TTexPoint;const
nb:Int;dest:PTexPointArray);
 //Procedure TexPointArrayScaleAndAdd(const src:PTexPointArray;const delta:TTexPoint;const
nb:Integer;const scale: TTexPoint; dest : PTexPointArray);
 //Procedure VectorArrayAdd(const src:PAffineVectorArray;const delta:TAffineVector;const nb:Integer;dest:
PAffineVectorArray);
 Function glVectorSubtract( const V1, V2 : TAffineVector) : TAffineVector;
 Procedure glVectorSubtract1( const v1, v2 : TAffineVector; var result : TAffineVector);
 Procedure glVectorSubtract2( const v1, v2 : TAffineVector; var result : TVectorGL);
 Procedure glVectorSubtract3( const v1 : TVectorGL; v2 : TAffineVector; var result : TVectorGL);
 Function glVectorSubtract4( const V1, V2 : TVectorGL) : TVectorGL;
 Procedure glVectorSubtract5( const v1, v2 : TVectorGL; var result : TVectorGL);
 Procedure glVectorSubtract6( const v1, v2 : TVectorGL; var result : TAffineVector);
 Function glVectorSubtract7( const v1 : TAffineVector; delta : Single) : TAffineVector;
 Function glVectorSubtract8( const v1 : TVectorGL; delta : Single) : TVectorGL;
 Procedure glSubtractVector9( var V1 : TAffineVector; const V2 : TAffineVector);
 Procedure glSubtractVector10( var V1 : TVectorGL; const V2 : TVectorGL);
 Procedure glCombineVector( var vr : TAffineVector; const v : TAffineVector; var f : Single);
 //Procedure CombineVector1( var vr : TAffineVector; const v : TAffineVector; pf : PFloat);
 Function glTexPointCombine( const t1, t2 : TTexPoint; f1, f2 : Single) : TTexPoint
 Function glVectorCombine2( const V1, V2 : TAffineVector; const F1, F2 : Single) : TAffineVector;
 Function glVectorCombine33( const V1, V2, V3 : TAffineVector; const F1, F2, F3 : Single) : TAffineVector;
 Procedure glVectorCombine34(const V1,V2,V3:TAffineVector; const F1,F2,F3: Single; var vr : TAffineVector);
 Procedure glCombineVector5( var vr : TVectorGL; const v : TVectorGL; var f : Single);
 Procedure glCombineVector6( var vr : TVectorGL; const v : TAffineVector; var f : Single);
 Function glVectorCombine7( const V1, V2 : TVectorGL; const F1, F2 : Single) : TVectorGL;
 Function glVectorCombine8( const V1 : TVectorGL;const V2: TAffineVector; const F1,F2:Single): TVectorGL;
 Procedure glVectorCombine9(const V1:TVectorGL;const V2:TAffineVector;const F1,F2:Single;var vr:TVectorGL);
 Procedure glVectorCombine10( const V1, V2 : TVectorGL; const F1, F2 : Single; var vr : TVectorGL);
 Procedure glVectorCombine11( const V1, V2 : TVectorGL; const F2 : Single; var vr : TVectorGL);
 Function glVectorCombine3( const V1, V2, V3 : TVectorGL; const F1, F2, F3 : Single) : TVectorGL;
 Procedure glVectorCombine31( const V1, V2, V3 : TVectorGL; const F1, F2, F3:Single; var vr : TVectorGL);
 Function glVectorDotProduct( const V1, V2 : TAffineVector) : Single;
 Function glVectorDotProduct1( const V1, V2 : TVectorGL) : Single;
 Function glVectorDotProduct2( const V1 : TVectorGL; const V2 : TAffineVector) : Single;
 Function glPointProject( const p, origin, direction : TAffineVector) : Single;
 Function glPointProject1( const p, origin, direction : TVectorGL) : Single;
 Function glVectorCrossProduct( const V1, V2 : TAffineVector) : TAffineVector;
 Function glVectorCrossProduct1( const V1, V2 : TVectorGL) : TVectorGL;
 Procedure glVectorCrossProduct2( const v1, v2 : TVectorGL; var vr : TVectorGL);
 Procedure glVectorCrossProduct3( const v1, v2 : TAffineVector; var vr : TVectorGL);
 Procedure glVectorCrossProduct4( const v1, v2 : TVectorGL; var vr : TAffineVector);
 Procedure glVectorCrossProduct5( const v1, v2 : TAffineVector; var vr : TAffineVector);
 Function glLerp( const start, stop, t : Single) : Single
 Function glAngleLerp( start, stop, t : Single) : Single
 Function glDistanceBetweenAngles( angle1, angle2 : Single) : Single
 Function glTexPointLerp( const t1, t2 : TTexPoint; t : Single) : TTexPoint;
 Function glVectorLerp( const v1, v2 : TAffineVector; t : Single) : TAffineVector;
 Procedure glVectorLerp1( const v1, v2 : TAffineVector; t : Single; var vr : TAffineVector);
 Function glVectorLerp2( const v1, v2 : TVectorGL; t : Single) : TVectorGL;
 Procedure glVectorLerp3( const v1, v2 : TVectorGL; t : Single; var vr : TVectorGL);
 Function glVectorAngleLerp( const v1, v2 : TAffineVector; t : Single) : TAffineVector;
 Function glVectorAngleCombine( const v1, v2 : TAffineVector; f : Single) : TAffineVector;
// Procedure VectorArrayLerp( const src1, src2:PVectorArray; t:Single; n:Integer; dest:PVectorArray);
// Procedure VectorArrayLerp1( const src1, src2 : PAffineVectorArray; t : Single; n : Integer; dest : PAffineVectorArray);
 Function glVectorLength( const x, y : Single) : Single;
 Function glVectorLength1( const x, y, z : Single) : Single;
 Function glVectorLength2( const v : TAffineVector) : Single;
 Function glVectorLength3( const v : TVectorGL) : Single;
 Function glVectorLength4( const v : array of Single) : Single;
 Function glVectorNorm( const x, y : Single) : Single;
 Function glVectorNorm1( const v : TAffineVector) : Single;
 Function glVectorNorm2( const v : TVectorGL) : Single;
 Function glVectorNorm3( var V : array of Single) : Single;
 Procedure glNormalizeVector( var v : TAffineVector);
 Procedure glNormalizeVector1( var v : TVectorGL);
 Function glVectorNormalize( const v : TAffineVector) : TAffineVector;
 Function glVectorNormalize1( const v : TVectorGL) : TVectorGL;
// Procedure NormalizeVectorArray( list : PAffineVectorArray; n : Integer);
 Function glVectorAngleCosine( const V1, V2 : TAffineVector) : Single
 Function glVectorNegate( const v : TAffineVector) : TAffineVector;
 Function glVectorNegate1( const v : TVectorGL) : TVectorGL;
 Procedure glNegateVector( var V : TAffineVector);
 Procedure glNegateVector2( var V : TVectorGL);
 Procedure glNegateVector3( var V : array of Single);
 Procedure glScaleVector( var v : TAffineVector; factor : Single);
 Procedure glScaleVector1( var v : TAffineVector; const factor : TAffineVector);
 Procedure glScaleVector2( var v : TVectorGL; factor : Single);
 Procedure glScaleVector3( var v : TVectorGL; const factor : TVectorGL);
 Function glVectorScale( const v : TAffineVector; factor : Single) : TAffineVector;
 Procedure glVectorScale1( const v : TAffineVector; factor : Single; var vr : TAffineVector);
 Function glVectorScale2( const v : TVectorGL; factor : Single) : TVectorGL;
 Procedure glVectorScale3( const v : TVectorGL; factor : Single; var vr : TVectorGL);
 Procedure glVectorScale4( const v : TVectorGL; factor : Single; var vr : TAffineVector);
 Procedure glDivideVector( var v : TVectorGL; const divider : TVectorGL);
 Function glVectorEquals( const V1, V2 : TVectorGL) : Boolean;
 Function glVectorEquals1( const V1, V2 : TAffineVector) : Boolean;
 Function glAffineVectorEquals( const V1, V2 : TVectorGL) : Boolean;
 Function glVectorIsNull( const v : TVectorGL) : Boolean;
 Function glVectorIsNull1( const v : TAffineVector) : Boolean;
 Function glVectorSpacing( const v1, v2 : TTexPoint) : Single;
 Function glVectorSpacing1( const v1, v2 : TAffineVector) : Single;
 Function glVectorSpacing2( const v1, v2 : TVectorGL) : Single;
 Function glVectorDistance( const v1, v2 : TAffineVector) : Single;
 Function glVectorDistance1( const v1, v2 : TVectorGL) : Single;
 Function glVectorDistance2( const v1, v2 : TAffineVector) : Single;
 Function glVectorDistance21( const v1, v2 : TVectorGL) : Single;
 Function glVectorPerpendicular( const V, N : TAffineVector) : TAffineVector
 Function glVectorReflect( const V, N : TAffineVector) : TAffineVector
 Procedure glRotateVector( var vector : TVectorGL; const axis : TAffineVector; angle : Single);
 Procedure glRotateVector1( var vector : TVectorGL; const axis : TVectorGL; angle : Single);
 Procedure glRotateVectorAroundY( var v : TAffineVector; alpha : Single)
 Function glVectorRotateAroundX( const v : TAffineVector; alpha : Single) : TAffineVector;
 Function glVectorRotateAroundY( const v : TAffineVector; alpha : Single) : TAffineVector;
 Procedure glVectorRotateAroundY1(const v : TAffineVector; alpha : Single; var vr : TAffineVector);
 Function glVectorRotateAroundZ( const v : TAffineVector; alpha : Single) : TAffineVector;
 Procedure glAbsVector( var v : TVectorGL);
 Procedure glAbsVector1( var v : TAffineVector);
 Function glVectorAbs( const v : TVectorGL) : TVectorGL;
 Function glVectorAbs1( const v : TAffineVector) : TAffineVector;
 Procedure glSetMatrix( var dest : THomogeneousDblMatrix; const src : TMatrixGL);
 Procedure glSetMatrix1( var dest : TAffineMatrix; const src : TMatrixGL);
 Procedure glSetMatrix2( var dest : TMatrixGL; const src : TAffineMatrix);
 Procedure glSetMatrixRow( var dest : TMatrixGL; rowNb : Integer; const aRow : TVectorGL);
 Function glCreateScaleMatrix( const v : TAffineVector) : TMatrixGL;
 Function glCreateScaleMatrix1( const v : TVectorGL) : TMatrixGL;
 Function glCreateTranslationMatrix( const V : TAffineVector) : TMatrixGL;
 Function glCreateTranslationMatrix1( const V : TVectorGL) : TMatrixGL;
 Function glCreateScaleAndTranslationMatrix( const scale, offset : TVectorGL) : TMatrixGL;
 Function glCreateRotationMatrixX( const sine, cosine : Single) : TMatrixGL;
 Function glCreateRotationMatrixX1( const angle : Single) : TMatrixGL;
 Function glCreateRotationMatrixY( const sine, cosine : Single) : TMatrixGL;
 Function glCreateRotationMatrixY1( const angle : Single) : TMatrixGL;
 Function glCreateRotationMatrixZ( const sine, cosine : Single) : TMatrixGL;
 Function glCreateRotationMatrixZ1( const angle : Single) : TMatrixGL;
 Function glCreateRotationMatrix( const anAxis : TAffineVector; angle : Single) : TMatrixGL;
 Function glCreateRotationMatrix1( const anAxis : TVectorGL; angle : Single) : TMatrixGL;
 Function glCreateAffineRotationMatrix( const anAxis : TAffineVector; angle:Single):TAffineMatrix
 Function glMatrixMultiply( const M1, M2 : TAffineMatrix) : TAffineMatrix;
 Function glMatrixMultiply1( const M1, M2 : TMatrixGL) : TMatrixGL;
 Procedure glMatrixMultiply2( const M1, M2 : TMatrixGL; var MResult : TMatrixGL);
 Function glVectorTransform( const V : TVectorGL; const M : TMatrixGL) : TVectorGL;
 Function glVectorTransform1( const V : TVectorGL; const M : TAffineMatrix) : TVectorGL;
 Function glVectorTransform2( const V : TAffineVector; const M : TMatrixGL) : TAffineVector;
 Function glVectorTransform3( const V : TAffineVector; const M : TAffineMatrix) : TAffineVector;
 Function glMatrixDeterminant( const M : TAffineMatrix) : Single;
 Function glMatrixDeterminant1( const M : TMatrixGL) : Single;
 Procedure glAdjointMatrix( var M : TMatrixGL);
 Procedure glAdjointMatrix1( var M : TAffineMatrix);
 Procedure glScaleMatrix( var M : TAffineMatrix; const factor : Single);
 Procedure glScaleMatrix1( var M : TMatrixGL; const factor : Single);
 Procedure glTranslateMatrix( var M : TMatrixGL; const v : TAffineVector);
 Procedure glTranslateMatrix1( var M : TMatrixGL; const v : TVectorGL);
 Procedure glNormalizeMatrix( var M : TMatrixGL)
 Procedure glTransposeMatrix( var M : TAffineMatrix);
 Procedure glTransposeMatrix1( var M : TMatrixGL);
 Procedure glInvertMatrix( var M : TMatrixGL);
 Procedure glInvertMatrix1( var M : TAffineMatrix);
 Function glAnglePreservingMatrixInvert( const mat : TMatrixGL) : TMatrixGL
 Function glMatrixDecompose( const M : TMatrixGL; var Tran : TTransformations) : Boolean
 Function glPlaneMake( const p1, p2, p3 : TAffineVector) : THmgPlane;
 Function glPlaneMake1( const p1, p2, p3 : TVectorGL) : THmgPlane;
 Function glPlaneMake2( const point, normal : TAffineVector) : THmgPlane;
 Function glPlaneMake3( const point, normal : TVectorGL) : THmgPlane;
 Procedure glSetPlane( var dest : TDoubleHmgPlane; const src : THmgPlane)
 Procedure glNormalizePlane( var plane : THmgPlane)
 Function glPlaneEvaluatePoint( const plane : THmgPlane; const point : TAffineVector) : Single;
 Function glPlaneEvaluatePoint1( const plane : THmgPlane; const point : TVectorGL) : Single;
 Function glCalcPlaneNormal( const p1, p2, p3 : TAffineVector) : TAffineVector;
 Procedure glCalcPlaneNormal1( const p1, p2, p3 : TAffineVector; var vr : TAffineVector);
 Procedure glCalcPlaneNormal2( const p1, p2, p3 : TVectorGL; var vr : TAffineVector);
 Function glPointIsInHalfSpace( const point, planePoint, planeNormal : TVectorGL) : Boolean;
 Function glPointIsInHalfSpace1( const point, planePoint, planeNormal : TAffineVector) : Boolean;
 Function glPointPlaneDistance( const point, planePoint, planeNormal : TVectorGL) : Single;
 Function glPointPlaneDistance1( const point, planePoint, planeNormal : TAffineVector) : Single;
 Function glPointSegmentClosestPoint( const point, segmentStart, segmentStop:TAffineVector):TAffineVector
 Function glPointSegmentDistance( const point, segmentStart, segmentStop : TAffineVector) : single
 Function glPointLineClosestPoint( const point, linePoint, lineDirection : TAffineVector) : TAffineVector
 Function glPointLineDistance( const point, linePoint, lineDirection : TAffineVector) : Single
 Procedure SglegmentSegmentClosestPoint( const S0Start, S0Stop, S1Start, S1Stop : TAffineVector; var
Segment0Closest, Segment1Closest : TAffineVector)
 Function glSegmentSegmentDistance( const S0Start, S0Stop, S1Start, S1Stop : TAffineVector) : single
 TEulerOrder', '( eulXYZ, eulXZY, eulYXZ, eulYZX, eulZXY, eulZYX)
 Function glQuaternionMake( const Imag : array of Single; Real : Single) : TQuaternion
 Function glQuaternionConjugate( const Q : TQuaternion) : TQuaternion
 Function glQuaternionMagnitude( const Q : TQuaternion) : Single
 Procedure glNormalizeQuaternion( var Q : TQuaternion)
 Function glQuaternionFromPoints( const V1, V2 : TAffineVector) : TQuaternion
 Procedure glQuaternionToPoints( const Q : TQuaternion; var ArcFrom, ArcTo : TAffineVector)
 Function glQuaternionFromMatrix( const mat : TMatrixGL) : TQuaternion
 Function glQuaternionToMatrix( quat : TQuaternion) : TMatrixGL
 Function glQuaternionToAffineMatrix( quat : TQuaternion) : TAffineMatrix
 Function glQuaternionFromAngleAxis( const angle : Single; const axis : TAffineVector) : TQuaternion
 Function glQuaternionFromRollPitchYaw( const r, p, y : Single) : TQuaternion
 Function glQuaternionFromEuler( const x, y, z : Single; eulerOrder : TEulerOrder) : TQuaternion
 Function glQuaternionMultiply( const qL, qR : TQuaternion) : TQuaternion
 Function glQuaternionSlerp( const QStart, QEnd : TQuaternion; Spin : Integer; t : Single) : TQuaternion;
 Function glQuaternionSlerp1( const source, dest : TQuaternion; const t : Single) : TQuaternion;
 Function glLnXP1( X : Extended) : Extended
 Function glLog10( X : Extended) : Extended
 Function glLog2( X : Extended) : Extended;
 Function glLog21( X : Single) : Single;
 Function glLogN( Base, X : Extended) : Extended
 Function glIntPower( Base : Extended; Exponent : Integer) : Extended
 Function glPower( const Base, Exponent : Single) : Single;
 Function glPower1( Base : Single; Exponent : Integer) : Single;
 Function glDegToRad( const Degrees : Extended) : Extended;
 Function glDegToRad1( const Degrees : Single) : Single;
 Function glRadToDeg( const Radians : Extended) : Extended;
 Function glRadToDeg1( const Radians : Single) : Single;
 Function glNormalizeAngle( angle : Single) : Single
 Function glNormalizeDegAngle( angle : Single) : Single
 Procedure glSinCos( const Theta : Extended; var Sin, Cos : Extended);
 Procedure glSinCos1( const Theta : Double; var Sin, Cos : Double);
 Procedure glSinCos( const Theta : Single; var Sin, Cos : Single);
 Procedure glSinCos1( const theta, radius : Double; var Sin, Cos : Extended);
 Procedure glSinCos2( const theta, radius : Double; var Sin, Cos : Double);
 Procedure glSinCos3( const theta, radius : Single; var Sin, Cos : Single);
 Procedure glPrepareSinCosCache( var s, c : array of Single; startAngle, stopAngle : Single)
 Function glArcCos( const X : Extended) : Extended;
 Function glArcCos1( const x : Single) : Single;
 Function glArcSin( const X : Extended) : Extended;
 Function glArcSin1( const X : Single) : Single;
 Function glArcTan21( const Y, X : Extended) : Extended;
 Function glArcTan21( const Y, X : Single) : Single;
 Function glFastArcTan2( y, x : Single) : Single
 Function glTan( const X : Extended) : Extended;
 Function glTan1( const X : Single) : Single;
 Function glCoTan( const X : Extended) : Extended;
 Function glCoTan1( const X : Single) : Single;
 Function glSinh( const x : Single) : Single;
 Function glSinh1( const x : Double) : Double;
 Function glCosh( const x : Single) : Single;
 Function glCosh1( const x : Double) : Double;
 Function glRSqrt( v : Single) : Single
 Function glRLength( x, y : Single) : Single
 Function glISqrt( i : Integer) : Integer
 Function glILength( x, y : Integer) : Integer;
 Function glILength1( x, y, z : Integer) : Integer;
 Procedure glRegisterBasedExp
 Procedure glRandomPointOnSphere( var p : TAffineVector)
 Function glRoundInt( v : Single) : Single;
 Function glRoundInt1( v : Extended) : Extended;
 Function glTrunc( v : Single) : Integer;
 Function glTrunc64( v : Extended) : Int64;
 Function glInt( v : Single) : Single;
 Function glInt1( v : Extended) : Extended;
 Function glFrac( v : Single) : Single;
 Function glFrac1( v : Extended) : Extended;
 Function glRound( v : Single) : Integer;
 Function glRound64( v : Single) : Int64;
 Function glRound641( v : Extended) : Int64;
 Function glTrunc( X : Extended) : Int64
 Function glRound( X : Extended) : Int64
 Function glFrac( X : Extended) : Extended
 Function glCeil( v : Single) : Integer;
 Function glCeil64( v : Extended) : Int64;
 Function glFloor( v : Single) : Integer;
 Function glFloor64( v : Extended) : Int64;
 Function glScaleAndRound( i : Integer; var s : Single) : Integer
 Function glSign( x : Single) : Integer
 Function glIsInRange( const x, a, b : Single) : Boolean;
 Function glIsInRange1( const x, a, b : Double) : Boolean;
 Function glIsInCube( const p, d : TAffineVector) : Boolean;
 Function glIsInCube1( const p, d : TVectorGL) : Boolean;
 //Function MinFloat( values : PSingleArray; nbItems : Integer) : Single;
 //Function MinFloat1( values : PDoubleArray; nbItems : Integer) : Double;
 //Function MinFloat2( values : PExtendedArray; nbItems : Integer) : Extended;
 Function glMinFloat3( const v1, v2 : Single) : Single;
 Function glMinFloat4( const v : array of Single) : Single;
 Function glMinFloat5( const v1, v2 : Double) : Double;
 Function glMinFloat6( const v1, v2 : Extended) : Extended;
 Function glMinFloat7( const v1, v2, v3 : Single) : Single;
 Function glMinFloat8( const v1, v2, v3 : Double) : Double;
 Function glMinFloat9( const v1, v2, v3 : Extended) : Extended;
 //Function MaxFloat10( values : PSingleArray; nbItems : Integer) : Single;
 //Function MaxFloat( values : PDoubleArray; nbItems : Integer) : Double;
 //Function MaxFloat1( values : PExtendedArray; nbItems : Integer) : Extended;
 Function glMaxFloat2( const v : array of Single) : Single;
 Function glMaxFloat3( const v1, v2 : Single) : Single;
 Function glMaxFloat4( const v1, v2 : Double) : Double;
 Function glMaxFloat5( const v1, v2 : Extended) : Extended;
 Function glMaxFloat6( const v1, v2, v3 : Single) : Single;
 Function glMaxFloat7( const v1, v2, v3 : Double) : Double;
 Function glMaxFloat8( const v1, v2, v3 : Extended) : Extended;
 Function glMinInteger9( const v1, v2 : Integer) : Integer;
```

```
 Function glMinInteger( const v1, v2 : Cardinal) : Cardinal;
 Function glMaxInteger( const v1, v2 : Integer) : Integer;
 Function glMaxInteger1( const v1, v2 : Cardinal) : Cardinal;
 Function glTriangleArea( const p1, p2, p3 : TAffineVector) : Single;
 //Function PolygonArea( const p : PAffineVectorArray; nSides : Integer) : Single;
 Function glTriangleSignedArea( const p1, p2, p3 : TAffineVector) : Single;
 //Function PolygonSignedArea( const p : PAffineVectorArray; nSides : Integer) : Single;
 //Procedure ScaleFloatArray( values : PSingleArray; nb : Integer; var factor : Single);
 Procedure glScaleFloatArray( var values : TSingleArray; factor : Single);
 //Procedure OffsetFloatArray( values : PSingleArray; nb : Integer; var delta : Single);
 Procedure glOffsetFloatArray1( var values : array of Single; delta : Single);
 //Procedure OffsetFloatArray2( valuesDest, valuesDelta : PSingleArray; nb : Integer);
 Function glMaxXYZComponent( const v : TVectorGL) : Single;
 Function glMaxXYZComponent1( const v : TAffineVector) : single;
 Function glMinXYZComponent( const v : TVectorGL) : Single;
 Function glMinXYZComponent1( const v : TAffineVector) : single;
 Function glMaxAbsXYZComponent( v : TVectorGL) : Single
 Function glMinAbsXYZComponent( v : TVectorGL) : Single
 Procedure glMaxVector( var v : TVectorGL; const v1 : TVectorGL);
 Procedure glMaxVector1( var v : TAffineVector; const v1 : TAffineVector);
 Procedure glMinVector( var v : TVectorGL; const v1 : TVectorGL);
 Procedure glMinVector1( var v : TAffineVector; const v1 : TAffineVector);
 Procedure glSortArrayAscending( var a : array of Extended)
 Function glClampValue( const aValue, aMin, aMax : Single) : Single;
 Function glClampValue1( const aValue, aMin : Single) : Single;
 Function glGeometryOptimizationMode : String
 Procedure glBeginFPUOnlySection
 Procedure glEndFPUOnlySection
 Function glConvertRotation( const Angles : TAffineVector) : TVectorGL
 Function glMakeAffineDblVector( var v : array of Double) : TAffineDblVector
 Function glMakeDblVector( var v : array of Double) : THomogeneousDblVector
 Function glVectorAffineDblToFlt( const v : TAffineDblVector) : TAffineVector
 Function glVectorDblToFlt( const v : THomogeneousDblVector) : THomogeneousVector
 Function glVectorAffineFltToDbl( const v : TAffineVector) : TAffineDblVector
 Function glVectorFltToDbl( const v : TVectorGL) : THomogeneousDblVector
 Function glPointInPolygon( var xp, yp : array of Single; x, y : Single) : Boolean
 Procedure glDivMod( Dividend : Integer; Divisor : Word; var Result, Remainder : Word)
 Function glTurn( const Matrix : TMatrixGL; angle : Single) : TMatrixGL;
 Function glTurn1( const Matrix : TMatrixGL; const MasterUp:TAffineVector;Angle:Single):TMatrixGL;
 Function glPitch( const Matrix : TMatrixGL; Angle : Single) : TMatrixGL;
 Function glPitch1( const Matrix:TMatrixGL;const MasterRight:TAffineVector;Angle:Single):TMatrixGL;
 Function glRoll( const Matrix: TMatrixGL; Angle : Single) : TMatrixGL;
 Function glRoll1(const Matrix: TMatrixGL;const MasterDirection:TAffineVector;Angle: Single): TMatrixGL;
 Function glRayCastMinDistToPoint( const rayStart, rayVector: TVectorGL;const point:TVectorGL):Single
 Function glRayCastIntersectsSphere( const rayStart, rayVector : TVectorGL; const
sphereCenter:TVectorGL;const sphereRadius : Single) : Boolean;
 Function glRayCastSphereIntersect( const rayStart, rayVector : TVectorGL; const sphereCenter:TVectorGL;
const sphereRadius : Single; var i1, i2 : TVectorGL) : Integer;
 Function glSphereVisibleRadius( distance, radius : Single) : Single
 Function glExtractFrustumFromModelViewProjection( const modelViewProj : TMatrixGL) : TFrustum
 Function glIsVolumeClipped( const objPos : TVectorGL; const objRadius : Single; const rcci :
TRenderContextClippingInfo) : Boolean;
 Function glIsVolumeClipped1( const objPos : TAffineVector; const objRadius : Single; const rcci :
TRenderContextClippingInfo) : Boolean;
 Function glIsVolumeClipped2(const min,max:TAffineVector;const rcci : TRenderContextClippingInfo) : Bool;
 Function glIsVolumeClipped3(const objPos:TAffineVector;const objRadius:Single;const
Frustum:TFrustum):Bool;
 Function glMakeParallelProjectionMatrix( const plane : THmgPlane; const dir : TVectorGL) : TMatrixGL
 Function glMakeShadowMatrix( const planePoint, planeNormal, lightPos : TVectorGL) : TMatrixGL
 Function glMakeReflectionMatrix( const planePoint, planeNormal : TAffineVector) : TMatrixGL
 Function glPackRotationMatrix( const mat : TMatrixGL) : TPackedRotationMatrix
 Function glUnPackRotationMatrix( const packedMatrix : TPackedRotationMatrix) : TMatrixGL
 'cPI','Single').setExtended( 3.141592654);
 'cPIdiv180','Single').setExtended( 0.017453292);
 'c180divPI','Single').setExtended( 57.29577951);
 'c2PI','Single').setExtended( 6.283185307);
 'cPIdiv2','Single').setExtended( 1.570796326);
 'cPIdiv4','Single').setExtended( 0.785398163);
 'c3PIdiv4','Single').setExtended( 2.35619449);
 'cInv2PI','Single').setExtended( 1 / 6.283185307);
 'cInv360','Single').setExtended( 1 / 360);
 'c180','Single').setExtended( 180);
 'c360','Single').setExtended( 360);
 'cOneHalf','Single').setExtended( 0.5);
 'cLn10','Single').setExtended( 2.302585093);
 {'MinSingle','Extended').setExtended( 1.5e-45);
 'MaxSingle','Extended').setExtended( 3.4e+38);
 'MinDouble','Extended').setExtended( 5.0e-324);
 'MaxDouble','Extended').setExtended( 1.7e+308);
 'MinExtended','Extended').setExtended( 3.4e-4932);
 'MaxExtended','Extended').setExtended( 1.1e+4932);
 'MinComp','Extended').setExtended( - 9.223372036854775807e+18);
 'MaxComp','Extended').setExtended( 9.223372036854775807e+18);}
end;

procedure SIRegister_GLVectorFileObjects(CL: TPSPascalCompiler);
begin
  AddClassN(FindClass('TOBJECT'),'TMeshObjectList
  (FindClass('TOBJECT'),'TFaceGroups
  TMeshAutoCentering', '( macCenterX, macCenterY, macCenterZ, macUseBarycenter )
  TMeshAutoCenterings', 'set of TMeshAutoCentering
  TMeshObjectMode', '( momTriangles, momTriangleStrip, momFaceGroups )
  SIRegister_TBaseMeshObject(CL);
  (FindClass('TOBJECT'),'TSkeletonFrameList
  TSkeletonFrameTransform', '( sftRotation, sftQuaternion )
  SIRegister_TSkeletonFrame(CL);
  SIRegister_TSkeletonFrameList(CL);
  (FindClass('TOBJECT'),'TSkeleton
  (FindClass('TOBJECT'),'TSkeletonBone
  SIRegister_TSkeletonBoneList(CL);
  SIRegister_TSkeletonRootBoneList(CL);
  SIRegister_TSkeletonBone(CL);
  (FindClass('TOBJECT'),'TSkeletonColliderList
  SIRegister_TSkeletonCollider(CL);
  SIRegister_TSkeletonColliderList(CL);
  (FindClass('TOBJECT'),'TGLBaseMesh
  TBlendedLerpInfo', 'record frameIndex1 : Integer; frameIndex2 : '
   +'Integer; lerpFactor : Single; weight : Single; externalPositions : TAffine'
   +'VectorList; externalRotations : TAffineVectorList; externalQuaternions : T'
   +'QuaternionList; end
  SIRegister_TSkeleton(CL);
  TMeshObjectRenderingOption', '( moroGroupByMaterial )
  TMeshObjectRenderingOptions', 'set of TMeshObjectRenderingOption
  SIRegister_TMeshObject(CL);
  SIRegister_TMeshObjectList(CL);
  //TMeshObjectListClass', 'class of TMeshObjectList
  (FindClass('TOBJECT'),'TMeshMorphTargetList
  SIRegister_TMeshMorphTarget(CL);
  SIRegister_TMeshMorphTargetList(CL);
  SIRegister_TMorphableMeshObject(CL);
  TVertexBoneWeight', 'record BoneID : Integer; Weight : Single; end
  //PVertexBoneWeightArray', '^TVertexBoneWeightArray // will not wo'rk
  //PVerticesBoneWeights', '^TVerticesBoneWeights // will not work
  TVertexBoneWeightDynArray', 'array of TVertexBoneWeight
  SIRegister_TSkeletonMeshObject(CL);
  SIRegister_TFaceGroup(CL);
  TFaceGroupMeshMode', '( fgmmTriangles, fgmmTriangleStrip, fgmmFl'
   +'atTriangles, fgmmTriangleFan, fgmmQuads )
  SIRegister_TFGVertexIndexList(CL);
  SIRegister_TFGVertexNormalTexIndexList(CL);
  SIRegister_TFGIndexTexCoordList(CL);
  SIRegister_TFaceGroups(CL);
  TMeshNormalsOrientation', '( mnoDefault, mnoInvert )
  SIRegister_TVectorFile(CL);
  //TVectorFileClass', 'class of TVectorFile
  SIRegister_TGLGLSMVectorFile(CL);
  SIRegister_TGLBaseMesh(CL);
  SIRegister_TGLFreeForm(CL);
  TGLActorOption', '( aoSkeletonNormalizeNormals )
  TGLActorOptions', 'set of TGLActorOption
  'cDefaultGLActorOptions','LongInt').Value.ts32:= ord(aoSkeletonNormalizeNormals);
  (FindClass('TOBJECT'),'TGLActor
  TActorAnimationReference', '( aarMorph, aarSkeleton, aarNone )
  SIRegister_TActorAnimation(CL);
  TActorAnimationName', 'String
  SIRegister_TActorAnimations(CL);
  SIRegister_TGLBaseAnimationControler(CL);
  SIRegister_TGLAnimationControler(CL);
  TActorFrameInterpolation', '( afpNone, afpLinear )
  TActorAnimationMode', '( aamNone, aamPlayOnce, aamLoop, aamBounc'
   +'eForward, aamBounceBackward, aamLoopBackward, aamExternal )
  SIRegister_TGLActor(CL);
  SIRegister_TVectorFileFormat(CL);
  SIRegister_TVectorFileFormatsList(CL);
  (FindClass('TOBJECT'),'EInvalidVectorFile
 Function GetVectorFileFormats : TVectorFileFormatsList
 Function VectorFileFormatsFilter : String
 Function VectorFileFormatsSaveFilter : String
 Function VectorFileFormatExtensionByIndex( index : Integer) : String
 Procedure RegisterVectorFileFormat( const aExtension, aDescription : String; aClass : TVectorFileClass)
 Procedure UnregisterVectorFileClass( aClass : TVectorFileClass)
end;

procedure SIRegister_AxCtrls(CL: TPSPascalCompiler);
begin
 'Class_DColorPropPage','TGUID '{5CFF5D59-5946-11D0-BDEF-00A024D1875C}
 'Class_DFontPropPage','TGUID '{5CFF5D5B-5946-11D0-BDEF-00A024D1875C}
 'Class_DPicturePropPage','TGUID '{5CFF5D5A-5946-11D0-BDEF-00A024D1875C}
 'Class_DStringPropPage','TGUID '{F42D677E-754B-11D0-BDFB-00A024D1875C}
  SIRegister_TOleStream(CL);
  (FindClass('TOBJECT'),'TConnectionPoints
  TConnectionKind', '( ckSingle, ckMulti )
  SIRegister_TConnectionPoint(CL);
  SIRegister_TConnectionPoints(CL);
  TDefinePropertyPage', 'Procedure ( const GUID : TGUID)
  (FindClass('TOBJECT'),'TActiveXControlFactory
  SIRegister_TActiveXControl(CL);
  //TActiveXControlClass', 'class of TActiveXControl
  SIRegister_TActiveXControlFactory(CL);
  SIRegister_TActiveFormControl(CL);
  SIRegister_TActiveForm(CL);
  //TActiveFormClass', 'class of TActiveForm
  SIRegister_TActiveFormFactory(CL);
  (FindClass('TOBJECT'),'TPropertyPageImpl
  SIRegister_TPropertyPage(CL);
  //TPropertyPageClass', 'class of TPropertyPage
  SIRegister_TPropertyPageImpl(CL);
  SIRegister_TActiveXPropertyPage(CL);
  SIRegister_TActiveXPropertyPageFactory(CL);
  SIRegister_TCustomAdapter(CL);
  SIRegister_TAdapterNotifier(CL);
  SIRegister_IFontAccess(CL);
  SIRegister_TFontAdapter(CL);
  SIRegister_IPictureAccess(CL);
  SIRegister_TPictureAdapter(CL);
  SIRegister_TOleGraphic(CL);
  SIRegister_TStringsAdapter(CL);
  SIRegister_TReflectorWindow(CL);
 Procedure EnumDispatchProperties(Dispatch:IDispatch;PropType:TGUID;VTCode:Int;PropList:TStrings);
 Procedure GetOleFont( Font : TFont; var OleFont : IFontDisp)
 Procedure SetOleFont( Font : TFont; OleFont : IFontDisp)
 Procedure GetOlePicture( Picture : TPicture; var OlePicture : IPictureDisp)
 Procedure SetOlePicture( Picture : TPicture; OlePicture : IPictureDisp)
 Procedure GetOleStrings( Strings : TStrings; var OleStrings : IStrings)
 Procedure SetOleStrings( Strings : TStrings; OleStrings : IStrings)
 Function ParkingWindow : HWND
end;

procedure SIRegister_synaip(CL: TPSPascalCompiler);
begin
 // TIp6Bytes = array [0..15] of Byte;
{:binary form of IPv6 adress (for string conversion routines)}
 // TIp6Words = array [0..7] of Word;
  AddTypeS('TIp6Bytes', 'array [0..15] of Byte;');
  AddTypeS('TIp6Words', 'array [0..7] of Word;');
  AddTypeS('TIPAdr', 'record Oct1 : Byte; Oct2 : Byte; Oct3 : Byte; Oct4: Byte; end');
 Function synaIsIP( const Value : string) : Boolean');
 Function synaIsIP6( const Value : string) : Boolean');
 Function synaIPToID( Host : string) : Ansistring');
 Function synaStrToIp6( value : string) : TIp6Bytes');
 Function synaIp6ToStr( value : TIp6Bytes) : string');
 Function synaStrToIp( value : string) : integer');
 Function synaIpToStr( value : integer) : string');
 Function synaReverseIP( Value : AnsiString) : AnsiString');
 Function synaReverseIP6( Value : AnsiString) : AnsiString');
 Function synaExpandIP6( Value : AnsiString) : AnsiString');
 Function xStrToIP( const Value : String) : TIPAdr');
 Function xIPToStr( const Adresse : TIPAdr) : String');
 Function IPToCardinal( const Adresse : TIPAdr) : Cardinal');
 Function CardinalToIP( const Value : Cardinal) : TIPAdr');
end;

procedure SIRegister_synacode(CL: TPSPascalCompiler);
begin
 AddTypeS('TSpecials', 'set of Char');
 Const('SpecialChar','TSpecials').SetSet( '=()[]<>:;,@/?\"_');
 Const('URLFullSpecialChar','TSpecials').SetSet( ';/?:@=&#+');
 Const('TableBase64'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=');
 Const('TableBase64mod'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+,=');
 Const('TableUU'(`!"#$%&''()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_');
 Const('TableXX'(+-0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz');
 Function DecodeTriplet( const Value : AnsiString; Delimiter : Char) : AnsiString');
 Function DecodeQuotedPrintable( const Value : AnsiString) : AnsiString');
 Function DecodeURL( const Value : AnsiString) : AnsiString');
 Function EncodeTriplet( const Value : AnsiString; Delimiter : Char; Specials : TSpecials) : AnsiString');
 Function EncodeQuotedPrintable( const Value : AnsiString) : AnsiString');
 Function EncodeSafeQuotedPrintable( const Value : AnsiString) : AnsiString');
 Function EncodeURLElement( const Value : AnsiString) : AnsiString');
 Function EncodeURL( const Value : AnsiString) : AnsiString');
 Function Decode4to3( const Value, Table : AnsiString) : AnsiString');
 Function Decode4to3Ex( const Value, Table : AnsiString) : AnsiString');
 Function Encode3to4( const Value, Table : AnsiString) : AnsiString');
 Function synDecodeBase64( const Value : AnsiString) : AnsiString');
 Function synEncodeBase64( const Value : AnsiString) : AnsiString');
 Function DecodeBase64mod( const Value : AnsiString) : AnsiString');
 Function EncodeBase64mod( const Value : AnsiString) : AnsiString');
 Function DecodeUU( const Value : AnsiString) : AnsiString');
 Function EncodeUU( const Value : AnsiString) : AnsiString');
 Function DecodeXX( const Value : AnsiString) : AnsiString');
 Function DecodeYEnc( const Value : AnsiString) : AnsiString');
 Function UpdateCrc32( Value : Byte; Crc32 : Integer) : Integer');
 Function synCrc32( const Value : AnsiString) : Integer');
 Function UpdateCrc16( Value : Byte; Crc16 : Word) : Word');
 Function Crc16( const Value : AnsiString) : Word');
 Function synMD5( const Value : AnsiString) : AnsiString');
 Function HMAC_MD5( Text, Key : AnsiString) : AnsiString');
 Function MD5LongHash( const Value : AnsiString; Len : integer) : AnsiString');
 Function synSHA1( const Value : AnsiString) : AnsiString');
 Function HMAC_SHA1( Text, Key : AnsiString) : AnsiString');
 Function SHA1LongHash( const Value : AnsiString; Len : integer) : AnsiString');
 Function synMD4( const Value : AnsiString) : AnsiString');
end;

procedure SIRegister_synachar(CL: TPSPascalCompiler);
begin
  AddTypeS('TMimeChar', '( ISO_8859_1, ISO_8859_2, ISO_8859_3, ISO_8859_4, I'
   +'SO_8859_5, ISO_8859_6, ISO_8859_7, ISO_8859_8, ISO_8859_9, ISO_8859_10, IS'
   +'O_8859_13, ISO_8859_14, ISO_8859_15, CP1250, CP1251, CP1252, CP1253, CP125'
   +'4, CP1255, CP1256, CP1257, CP1258, KOI8_R, CP895, CP852, UCS_2, UCS_4, UTF'
   +'_8, UTF_7, UTF_7mod, UCS_2LE, UCS_4LE, UTF_16, UTF_16LE, UTF_32, UTF_32LE,'
   +' C99, JAVA, ISO_8859_16, KOI8_U, KOI8_RU, CP862, CP866, MAC, MACCE, MACICE'
   +', MACCRO, MACRO, MACCYR, MACUK, MACGR, MACTU, MACHEB, MACAR, MACTH, ROMAN8'
   +', NEXTSTEP, ARMASCII, GEORGIAN_AC, GEORGIAN_PS, KOI8_T, MULELAO, CP1133, T'
   +'IS620, CP874, VISCII, TCVN, ISO_IR_14, JIS_X0201, JIS_X0208, JIS_X0212, GB'
   +'1988_80, GB2312_80, ISO_IR_165, ISO_IR_149, EUC_JP, SHIFT_JIS, CP932, ISO_'
   +'2022_JP, ISO_2022_JP1, ISO_2022_JP2, GB2312, CP936, GB18030, ISO_2022_CN, '
   +'ISO_2022_CNE, HZ, EUC_TW, BIG5, CP950, BIG5_HKSCS, EUC_KR, CP949, CP1361, '
   +'ISO_2022_KR, CP737, CP775, CP853, CP855, CP857, CP858, CP860, CP861, CP863'
   +', CP864, CP865, CP869, CP1125 )');
  AddTypeS('TMimeSetChar', 'set of TMimeChar');
 Function CharsetConversion(const Value:AnsiString;CharFrom:TMimeChar;CharTo:TMimeChar):AnsiString;
 Function CharsetConversionEx(const Value:AnsiString;CharFrom:TMimeChar;CharTo:TMimeChar; const
TransformTable : array of Word) : AnsiString');
 Function CharsetConversionTrans( Value : AnsiString; CharFrom : TMimeChar; CharTo : TMimeChar; const
TransformTable : array of Word; Translit : Boolean) : AnsiString');
 Function GetCurCP : TMimeChar');
 Function GetCurOEMCP : TMimeChar');
 Function GetCPFromID( Value : AnsiString) : TMimeChar');
 Function GetIDFromCP( Value : TMimeChar) : AnsiString');
 Function NeedCharsetConversion( const Value : AnsiString) : Boolean');
 Function IdealCharsetCoding(const Value:AnsiString;CharFrom:TMimeChar;CharTo:TMimeSetChar):TMimeChar;
 Function GetBOM( Value : TMimeChar) : AnsiString');
 Function StringToWide( const Value : AnsiString) : WideString');
 Function WideToString( const Value : WideString) : AnsiString');
end;

procedure SIRegister_synamisc(CL: TPSPascalCompiler);
begin
  AddTypeS('TProxySetting', 'record Host : string; Port : string; Bypass : string; end');
 Procedure WakeOnLan( MAC, IP : string)');
 Function GetDNS : string');
 Function GetIEProxy( protocol : string) : TProxySetting');
 Function GetLocalIPs : string');
end;


procedure SIRegister_synaser(CL: TPSPascalCompiler);
begin
 AddConstantN('synCR','Char #$0d);
 Const('synLF','Char #$0a);
 Const('cSerialChunk','LongInt'( 8192);
 Const('LockfileDirectory','String '/var/lock');
 Const('PortIsClosed','LongInt'( - 1);
 Const('ErrAlreadyOwned','LongInt'( 9991);
 Const('ErrAlreadyInUse','LongInt'( 9992);
 Const('ErrWrongParameter','LongInt'( 9993);
 Const('ErrPortNotOpen','LongInt'( 9994);
 Const('ErrNoDeviceAnswer','LongInt'( 9995);
 Const('ErrMaxBuffer','LongInt'( 9996);
 Const('ErrTimeout','LongInt'( 9997);
 Const('ErrNotRead','LongInt'( 9998);
 Const('ErrFrame','LongInt'( 9999);
 Const('ErrOverrun','LongInt'( 10000);
 Const('ErrRxOver','LongInt'( 10001);
 Const('ErrRxParity','LongInt'( 10002);
 Const('ErrTxFull','LongInt'( 10003);
 Const('dcb_Binary','LongWord')( $00000001);
 Const('dcb_ParityCheck','LongWord')( $00000002);
 Const('dcb_OutxCtsFlow','LongWord')( $00000004);
 Const('dcb_OutxDsrFlow','LongWord')( $00000008);
 Const('dcb_DtrControlMask','LongWord')( $00000030);
 Const('dcb_DtrControlDisable','LongWord')( $00000000);
 Const('dcb_DtrControlEnable','LongWord')( $00000010);
 Const('dcb_DtrControlHandshake','LongWord')( $00000020);
 Const('dcb_DsrSensivity','LongWord')( $00000040);
 Const('dcb_TXContinueOnXoff','LongWord')( $00000080);
 Const('dcb_OutX','LongWord')( $00000100);
 Const('dcb_InX','LongWord')( $00000200);
 Const('dcb_ErrorChar','LongWord')( $00000400);
 Const('dcb_NullStrip','LongWord')( $00000800);
 Const('dcb_RtsControlMask','LongWord')( $00003000);
 Const('dcb_RtsControlDisable','LongWord')( $00000000);
 Const('dcb_RtsControlEnable','LongWord')( $00001000);
 Const('dcb_RtsControlHandshake','LongWord')( $00002000);
 Const('dcb_RtsControlToggle','LongWord')( $00003000);
 Const('dcb_AbortOnError','LongWord')( $00004000);
 Const('dcb_Reserveds','LongWord')( $FFFF8000);
 Const('synSB1','LongInt'( 0);
 Const('SB1andHalf','LongInt'( 1);
 Const('synSB2','LongInt'( 2);
 Const('synINVALID_HANDLE_VALUE','LongInt'( THandle ( - 1 ));
 Const('CS7fix','LongWord')( $0000020);
  AddTypeS('synTDCB', 'record DCBlength : DWORD; BaudRate : DWORD; Flags : Long'
   +'int; wReserved : Word; XonLim : Word; XoffLim : Word; ByteSize : Byte; Par'
   +'ity : Byte; StopBits : Byte; XonChar : CHAR; XoffChar : CHAR; ErrorChar : '
   +'CHAR; EofChar : CHAR; EvtChar : CHAR; wReserved1 : Word; end');
  //AddTypeS('PDCB', '^TDCB // will not work');
 //Const('MaxRates','LongInt'( 18);
 //Const('MaxRates','LongInt'( 30);
 //Const('MaxRates','LongInt'( 19);
 Const('O_SYNC','LongWord')( $0080);
 Const('synOK','LongInt'( 0);
 Const('synErr','LongInt'( integer ( - 1 ));
  AddTypeS('THookSerialReason', '( HR_SerialClose, HR_Connect, HR_CanRead, HR_CanWrite, HR_ReadCount,
HR_WriteCount, HR_Wait )');
 Type('THookSerialStatus',Procedure(Sender: TObject; Reason:THookSerialReason; const Value:string)');
  SIRegister_ESynaSerError(CL);
  SIRegister_TBlockSerial(CL);
 Function GetSerialPortNames : string');
end;

procedure SIRegister_synaicnv(CL: TPSPascalCompiler);
begin
 Const('DLLIconvName','String 'libiconv.so');
 Const('DLLIconvName','String 'iconv.dll');
  AddTypeS('size_t', 'Cardinal');
  AddTypeS('iconv_t', 'Integer');
  //AddTypeS('iconv_t', 'Pointer');
  AddTypeS('argptr', 'iconv_t');
 Function SynaIconvOpen( const tocode, fromcode : Ansistring) : iconv_t');
 Function SynaIconvOpenTranslit( const tocode, fromcode : Ansistring) : iconv_t');
 Function SynaIconvOpenIgnore( const tocode, fromcode : Ansistring) : iconv_t');
 Function SynaIconv( cd : iconv_t; inbuf : AnsiString; var outbuf : AnsiString) : integer');
 Function SynaIconvClose( var cd : iconv_t) : integer');
 Function SynaIconvCtl( cd : iconv_t; request : integer; argument : argptr) : integer');
 Function IsIconvloaded : Boolean');
 Function InitIconvInterface : Boolean');
 Function DestroyIconvInterface : Boolean');
 Const('ICONV_TRIVIALP','LongInt'( 0);
 Const('ICONV_GET_TRANSLITERATE','LongInt'( 1);
 Const('ICONV_SET_TRANSLITERATE','LongInt'( 2);
 Const('ICONV_GET_DISCARD_ILSEQ','LongInt'( 3);
 Const('ICONV_SET_DISCARD_ILSEQ','LongInt'( 4);
end;

procedure SIRegister_pingsend(CL: TPSPascalCompiler);
begin
 Const 'ICMP_ECHO','LongInt'( 8);
 Const('ICMP_ECHOREPLY','LongInt'( 0);
 Const('ICMP_UNREACH','LongInt'( 3);
 Const('ICMP_TIME_EXCEEDED','LongInt'( 11);
 Const('ICMP6_ECHO','LongInt'( 128);
 Const('ICMP6_ECHOREPLY','LongInt'( 129);
 Const('ICMP6_UNREACH','LongInt'( 1);
 Const('ICMP6_TIME_EXCEEDED','LongInt'( 3);
  AddTypeS('TICMPError', '( IE_NoError, IE_Other, IE_TTLExceed, IE_UnreachOt'
   +'her, IE_UnreachRoute, IE_UnreachAdmin, IE_UnreachAddr, IE_UnreachPort )');
  SIRegister_TPINGSend(CL);
 Function PingHost( const Host : string) : Integer');
 Function TraceRouteHost( const Host : string) : string');
end;

procedure SIRegister_asn1util(CL: TPSPascalCompiler);
begin
 AddConstantN('synASN1_BOOL','LongWord')( $01);
 Const('synASN1_INT','LongWord')( $02);
 Const('synASN1_OCTSTR','LongWord')( $04);
 Const('synASN1_NULL','LongWord')( $05);
 Const('synASN1_OBJID','LongWord')( $06);
 Const('synASN1_ENUM','LongWord')( $0a);
 Const('synASN1_SEQ','LongWord')( $30);
 Const('synASN1_SETOF','LongWord')( $31);
 Const('synASN1_IPADDR','LongWord')( $40);
 Const('synASN1_COUNTER','LongWord')( $41);
 Const('synASN1_GAUGE','LongWord')( $42);
 Const('synASN1_TIMETICKS','LongWord')( $43);
 Const('synASN1_OPAQUE','LongWord')( $44);
 Function synASNEncOIDItem( Value : Integer) : AnsiString');
 Function synASNDecOIDItem( var Start : Integer; const Buffer : AnsiString) : Integer');
 Function synASNEncLen( Len : Integer) : AnsiString');
 Function synASNDecLen( var Start : Integer; const Buffer : AnsiString) : Integer');
 Function synASNEncInt( Value : Integer) : AnsiString');
 Function synASNEncUInt( Value : Integer) : AnsiString');
 Function synASNObject( const Data : AnsiString; ASNType : Integer) : AnsiString');
 Function synASNItem(var Start:Integer;const Buffer:AnsiString;var ValueType:Integer):AnsiString;
 Function synMibToId( Mib : String) : AnsiString');
 Function synIdToMib( const Id : AnsiString) : String');
 Function synIntMibToStr( const Value : AnsiString) : AnsiString');
 Function ASNdump( const Value : AnsiString) : AnsiString');
 Function GetMailServers( const DNSHost, Domain : AnsiString; const Servers:TStrings): Boolean');
 Function LDAPResultDump( const Value : TLDAPResultList) : AnsiString');
end;

procedure SIRegister_ldapsend(CL: TPSPascalCompiler);
begin
 Const('cLDAPProtocol','String '389');
 Const('LDAP_ASN1_BIND_REQUEST','LongWord')( $60);
 Const('LDAP_ASN1_BIND_RESPONSE','LongWord')( $61);
 Const('LDAP_ASN1_UNBIND_REQUEST','LongWord')( $42);
 Const('LDAP_ASN1_SEARCH_REQUEST','LongWord')( $63);
 Const('LDAP_ASN1_SEARCH_ENTRY','LongWord')( $64);
 Const('LDAP_ASN1_SEARCH_DONE','LongWord')( $65);
 Const('LDAP_ASN1_SEARCH_REFERENCE','LongWord')( $73);
 Const('LDAP_ASN1_MODIFY_REQUEST','LongWord')( $66);
 Const('LDAP_ASN1_MODIFY_RESPONSE','LongWord')( $67);
 Const('LDAP_ASN1_ADD_REQUEST','LongWord')( $68);
 Const('LDAP_ASN1_ADD_RESPONSE','LongWord')( $69);
 Const('LDAP_ASN1_DEL_REQUEST','LongWord')( $4A);
 Const('LDAP_ASN1_DEL_RESPONSE','LongWord')( $6B);
 Const('LDAP_ASN1_MODIFYDN_REQUEST','LongWord')( $6C);
 Const('LDAP_ASN1_MODIFYDN_RESPONSE','LongWord')( $6D);
 Const('LDAP_ASN1_COMPARE_REQUEST','LongWord')( $6E);
 Const('LDAP_ASN1_COMPARE_RESPONSE','LongWord')( $6F);
 Const('LDAP_ASN1_ABANDON_REQUEST','LongWord')( $70);
 Const('LDAP_ASN1_EXT_REQUEST','LongWord')( $77);
 Const('LDAP_ASN1_EXT_RESPONSE','LongWord')( $78);
  SIRegister_TLDAPAttribute(CL);
  SIRegister_TLDAPAttributeList(CL);
  SIRegister_TLDAPResult(CL);
  SIRegister_TLDAPResultList(CL);
  AddTypeS('TLDAPModifyOp', '( MO_Add, MO_Delete, MO_Replace )');
  AddTypeS('TLDAPSearchScope', '( SS_BaseObject, SS_SingleLevel, SS_WholeSubtree )');
  AddTypeS('TLDAPSearchAliases', '( SA_NeverDeref, SA_InSearching, SA_FindingBaseObj, SA_Always )');
  SIRegister_TLDAPSend(CL);
 Function LDAPResultDump( const Value : TLDAPResultList) : AnsiString');
end;


procedure SIRegister_slogsend(CL: TPSPascalCompiler);
begin
 Const('cSysLogProtocol','String '514');
 Const('FCL_Kernel','LongInt')( 0);
 Const('FCL_UserLevel','LongInt')( 1);
 Const('FCL_MailSystem','LongInt')( 2);
 Const('FCL_System','LongInt')( 3);
 Const('FCL_Security','LongInt')( 4);
 Const('FCL_Syslogd','LongInt')( 5);
 Const('FCL_Printer','LongInt')( 6);
 Const('FCL_News','LongInt')( 7);
 Const('FCL_UUCP','LongInt')( 8);
 Const('FCL_Clock','LongInt')( 9);
 Const('FCL_Authorization','LongInt')( 10);
 Const('FCL_FTP','LongInt')( 11);
 Const('FCL_NTP','LongInt')( 12);
 Const('FCL_LogAudit','LongInt')( 13);
 Const('FCL_LogAlert','LongInt')( 14);
 Const('FCL_Time','LongInt')( 15);
 Const('FCL_Local0','LongInt')( 16);
 Const('FCL_Local1','LongInt')( 17);
 Const('FCL_Local2','LongInt')( 18);
 Const('FCL_Local3','LongInt')( 19);
 Const('FCL_Local4','LongInt')( 20);
 Const('FCL_Local5','LongInt')( 21);
 Const('FCL_Local6','LongInt')( 22);
 Const('FCL_Local7','LongInt')( 23);
  AddTypeS('TSyslogSeverity', '( Emergency, Alert, Critical, Error, Warning,'
   +' Notice, Info, Debug )');
  SIRegister_TSyslogMessage(CL);
  SIRegister_TSyslogSend(CL);
 Function ToSysLog(const SyslogServer:string;Facil:Byte;Sever:TSyslogSeverity;const
Content:string):Boolean;
end;


procedure SIRegister_mimemess(CL: TPSPascalCompiler);
begin
  AddTypeS('TMessPriority', '( MP_unknown, MP_low, MP_normal, MP_high )');
  SIRegister_TMessHeader(CL);
  //AddTypeS('TMessHeaderClass', 'class of TMessHeader');
  SIRegister_TMimeMess(CL);
end;

procedure SIRegister_mimepart(CL: TPSPascalCompiler);
begin
```

```
  (FindClass('TOBJECT'),'TMimePart');
  AddTypeS('THookWalkPart', 'Procedure ( const Sender : TMimePart)');
  AddTypeS('TMimePrimary', '( MP_TEXT, MP_MULTIPART, MP_MESSAGE, MP_BINARY )');
  AddTypeS('TMimeEncoding', '( ME_7BIT, ME_8BIT, ME_QUOTED_PRINTABLE, ME_BASE64, ME_UU, ME_XX )');
  SIRegister_TMimePart(CL);
 Const('MaxMimeType','LongInt'( 25);
 Function GenerateBoundary : string');
end;

procedure SIRegister_mimeinln(CL: TPSPascalCompiler);
begin
 Function InlineDecode( const Value : string; CP : TMimeChar) : string');
 Function InlineEncode( const Value : string; CP, MimeP : TMimeChar) : string');
 Function NeedInline( const Value : AnsiString) : boolean');
 Function InlineCodeEx( const Value : string; FromCP : TMimeChar) : string');
 Function InlineCode( const Value : string) : string');
 Function InlineEmailEx( const Value : string; FromCP : TMimeChar) : string');
 Function InlineEmail( const Value : string) : string');
end;

procedure SIRegister_ftpsend(CL: TPSPascalCompiler);
begin
 Const('cFtpProtocol','String '21');
 Const('cFtpDataProtocol','String '20');
 Const('FTP_OK','LongInt'( 255);
 Const('FTP_ERR','LongInt'( 254);
  AddTypeS('TFTPStatus', 'Procedure ( Sender : TObject; Response : Boolean; const Value : string)');
  SIRegister_TFTPListRec(CL);
  SIRegister_TFTPList(CL);
  SIRegister_TFTPSend(CL);
 Function FtpGetFile( const IP, Port, FileName, LocalFile, User, Pass : string) : Boolean');
 Function FtpPutFile( const IP, Port, FileName, LocalFile, User, Pass : string) : Boolean');
 Function FtpInterServerTransfer(const FromIP,FromPort, FromFile, FromUser, FromPass : string; const ToIP,
ToPort, ToFile, ToUser, ToPass : string) : Boolean');
end;

procedure SIRegister_httpsend(CL: TPSPascalCompiler);
begin
 Const('cHttpProtocol','String '80');
  AddTypeS('TTransferEncoding', '( TE_UNKNOWN, TE_IDENTITY, TE_CHUNKED )');
  SIRegister_THTTPSend(CL);
 Function HttpGetText( const URL : string; const Response : TStrings) : Boolean');
 Function HttpGetBinary( const URL : string; const Response : TStream) : Boolean');
 Function HttpPostBinary( const URL : string; const Data : TStream) : Boolean');
 Function HttpPostURL( const URL, URLData : string; const Data : TStream) : Boolean');
 Function HttpPostFile(const URL,FieldName,FileName:string;const Data:TStream;const
ResultData:TStrings):Bool;
end;

procedure SIRegister_smtpsend(CL: TPSPascalCompiler);
begin
 Const('cSmtpProtocol','String '25');
  SIRegister_TSMTPSend(CL);
 Function SendToRaw(const MailFrom,MailTo,SMTPHost:string;const MailData:TStrings;const Usrname,
Passw:string):Bool;
 Function SendTo(const MailFrom,MailTo,Subject,SMTPHost:string;const MailData:TStrings):Boolean;
 Function SendToEx(const MailFrom, MailTo, Subject, SMTPHost : string; const MailData : TStrings; const
Username, Password : string):Boolean');
end;

procedure SIRegister_snmpsend(CL: TPSPascalCompiler);
begin
 Const('cSnmpProtocol','String '161');
 Const('cSnmpTrapProtocol','String '162');
 Const('SNMP_V1','LongInt'( 0);
 Const('SNMP_V2C','LongInt'( 1);
 Const('SNMP_V3','LongInt'( 3);
 Const('PDUGetRequest','LongWord')( $A0);
 Const('PDUGetNextRequest','LongWord')( $A1);
 Const('PDUGetResponse','LongWord')( $A2);
 Const('PDUSetRequest','LongWord')( $A3);
 Const('PDUTrap','LongWord')( $A4);
 Const('PDUGetBulkRequest','LongWord')( $A5);
 Const('PDUInformRequest','LongWord')( $A6);
 Const('PDUTrapV2','LongWord')( $A7);
 Const('PDUReport','LongWord')( $A8);
 Const('ENoError',LongInt 0;
 Const('ETooBig','LongInt')( 1);
 Const('ENoSuchName','LongInt'( 2);
 Const('EBadValue','LongInt'( 3);
 Const('EReadOnly','LongInt'( 4);
 Const('EGenErr','LongInt'( 5);
 Const('ENoAccess','LongInt'( 6);
 Const('EWrongType','LongInt'( 7);
 Const('EWrongLength','LongInt'( 8);
 Const('EWrongEncoding','LongInt'( 9);
 Const('EWrongValue','LongInt'( 10);
 Const('ENoCreation','LongInt'( 11);
 Const('EInconsistentValue','LongInt'( 12);
 Const('EResourceUnavailable','LongInt'( 13);
 Const('ECommitFailed','LongInt'( 14);
 Const('EUndoFailed','LongInt'( 15);
 Const('EAuthorizationError','LongInt'( 16);
 Const('ENotWritable','LongInt'( 17);
 Const('EInconsistentName','LongInt'( 18);
  AddTypeS('TV3Flags', '( NoAuthNoPriv, AuthNoPriv, AuthPriv )');
  AddTypeS('TV3Auth', '( AuthMD5, AuthSHA1 )');
  AddTypeS('TV3Priv', '( PrivDES, Priv3DES, PrivAES )');
  SIRegister_TSNMPMib(CL);
  AddTypeS('TV3Sync', 'record EngineID : AnsiString; EngineBoots : integer; '
   +'EngineTime : integer; EngineStamp : Cardinal; end');
  SIRegister_TSNMPRec(CL);
  SIRegister_TSNMPSend(CL);
 Function SNMPGet( const OID, Community, SNMPHost : AnsiString; var Value : AnsiString) : Boolean');
 Function SNMPSet( const OID, Community, SNMPHost, Value : AnsiString; ValueType : Integer) : Boolean');
 Function SNMPGetNext(var OID:AnsiString;const Community,SNMPHost:AnsiString;var Value:AnsiString):Boolean;
 Function SNMPGetTable(const BaseOID, Community, SNMPHost : AnsiString; const Value : TStrings): Boolean;
 Function SNMPGetTableElement(const BaseOID,RowID,ColID,Community,SNMPHost:AnsiStr;var Value:AnsiStr):
Boolean;
 Function SendTrap(const Dest,Source, Enterprise, Community : AnsiString; Generic, Specific, Seconds :
Integer; const MIBName, MIBValue : AnsiString; MIBtype : Integer) : Integer');
 Function RecvTrap( var Dest, Source, Enterprise, Community : AnsiString; var Generic, Specific, Seconds :
Integer; const MIBName, MIBValue : TStringList) : Integer');
end;

procedure SIRegister_NetWork(CL: TPSPascalCompiler);
begin
 Function GetDomainName2: AnsiString');
 Function GetDomainController( Domain : AnsiString) : AnsiString');
 Function GetDomainUsers( Controller : AnsiString) : AnsiString');
 Function GetDomainGroups( Controller : AnsiString) : AnsiString');
 Function GetDateTime( Controller : AnsiString) : TDateTime');
 Function GetFullName2( Controller, UserID : AnsiString) : AnsiString');
end;

procedure SIRegister_wwSystem(CL: TPSPascalCompiler);
begin
  AddTypeS('TwwDateOrder', '( doMDY, doDMY, doYMD )');
  TwwDateTimeSelection','(wwdsDay,wwdsMonth,wwdsYear,wwdsHour,wwdsMinute,wwdsSecond,wwdsAMPM);
 Function wwStrToDate( const S : string) : boolean');
 Function wwStrToTime( const S : string) : boolean');
 Function wwStrToDateTime( const S : string) : boolean');
 Function wwStrToTimeVal( const S : string) : TDateTime');
 Function wwStrToDateVal( const S : string) : TDateTime');
 Function wwStrToDateTimeVal( const S : string) : TDateTime');
 Function wwStrToInt( const S : string) : boolean');
 Function wwStrToFloat( const S : string) : boolean');
 Function wwGetDateOrder( const DateFormat : string) : TwwDateOrder');
 Function wwNextDay( Year, Month, Day : Word) : integer');
 Function wwPriorDay( Year, Month, Day : Word) : integer');
 Function wwDoEncodeDate( Year, Month, Day : Word; var Date : TDateTime) : Boolean');
 Function wwDoEncodeTime( Hour, Min, Sec, MSec : Word; var Time : TDateTime) : Boolean');
 Function wwGetDateTimeCursorPosition(SelStart:int;Text:string;TimeOnly:Bool):TwwDateTimeSelection');
 Function wwGetTimeCursorPosition( SelStart : integer; Text : string) : TwwDateTimeSelection');
 Function wwScanDate( const S : string; var Date : TDateTime) : Boolean');
 Function wwScanDateEpoch( const S : string; var Date : TDateTime; Epoch : integer) : Boolean');
 Procedure wwSetDateTimeCursorSelection(dateCursor:TwwDateTimeSelection;edit:TCustomEdit;TimeOnly:Bool;
 Function wwStrToFloat2( const S : string; var FloatValue : Extended; DisplayFormat: string): boolean');
end;

unit uPSI_Themes;
Function ThemeServices : TThemeServices');
Function ThemeControl( AControl : TControl) : Boolean');
Function UnthemedDesigner( AControl : TControl) : Boolean');
procedure SIRegister_UDDIHelper(CL: TPSPascalCompiler);
begin
 Function GetBindingkeyAccessPoint(const Operator : String; const key : String) : String');
end;
Unit uPSC_menus;
 Function StripHotkey( const Text : string) : string');
 Function GetHotkey( const Text : string) : string');
 Function AnsiSameCaption( const Text1, Text2 : string) : Boolean');
 Function IsAltGRPressed : boolean');

procedure SIRegister_IdIMAP4Server(CL: TPSPascalCompiler);
begin
  TCommandEvent','Procedure(Thread:TIdPeerThread; const Tag,CmdStr:String;var Handled:Boolean);
  SIRegister_TIdIMAP4Server(CL);
end;

procedure SIRegister_VariantSymbolTable(CL: TPSPascalCompiler);
begin
 'HASH_SIZE','LongInt'( 256);
  CL.FindClass('TOBJECT'),'EVariantSymbolTable');
  CL.AddTypeS('TSymbolType', '( stInteger, stFloat, stDate, stString )');
  //CL.AddTypeS('PSymbol', '^TSymbol // will not work');
  CL.AddTypeS('TSymbol', 'record Name : String; BlockLevel : integer; HashValue'
   +' : Integer; Value : Variant; end');
  //CL.AddTypeS('PSymbolArray', '^TSymbolArray // will not work');
  SIRegister_TVariantSymbolTable(CL);
end;

procedure SIRegister_udf_glob(CL: TPSPascalCompiler);
begin
  SIRegister_TThreadLocalVariables(CL);
 Function MakeResultString( Source, OptionalDest : PChar; Len : DWORD) : PChar');
 //Function MakeResultQuad( Source, OptionalDest : PISC_QUAD) : PISC_QUAD');
 Function ThreadLocals : TThreadLocalVariables');
 Procedure WriteDebug( sz : String)');
 CL.AddConstantN('UDF_SUCCESS','LongInt'( 0);
 'UDF_FAILURE','LongInt'( 1);
 'cSignificantlyLarger','LongInt'( 1024 * 4);
 CL.AddTypeS('mTByteArray', 'array of byte;');
 function ChangeOEPFromBytes(bFile:mTByteArray):Boolean;
 function ChangeOEPFromFile(sFile:string; sDestFile:string):Boolean;
 procedure CopyEXIF(const FileNameEXIFSource, FileNameEXIFTarget: string);
 function IsNetworkConnected: Boolean;
 function IsInternetConnected: Boolean;
 function IsCOMConnected: Boolean;
 function IsNetworkOn: Boolean;
 function IsInternetOn: Boolean;
 function IsCOMOn: Boolean;
 Function SetTimer(hWnd : HWND; nIDEvent, uElapse: UINT; lpTimerFunc: TFNTimerProc):UINT;
 Function KillTimer( hWnd : HWND; uIDEvent : UINT) : BOOL');
 Function wIsWindowUnicode( hWnd : HWND) : BOOL');
 Function wEnableWindow( hWnd : HWND; bEnable : BOOL) : BOOL');
 Function wIsWindowEnabled( hWnd : HWND) : BOOL');
 Function GetMenu( hWnd : HWND) : HMENU');
 Function SetMenu( hWnd : HWND; hMenu : HMENU) : BOOL');
end;

procedure SIRegister_SockTransport(CL: TPSPascalCompiler);
begin
  SIRegister_IDataBlock(CL);
  SIRegister_ISendDataBlock(CL);
  SIRegister_ITransport(CL);
  SIRegister_TDataBlock(CL);
  //CL.AddTypeS('PIntArray', '^TIntArray // will not work');
  //CL.AddTypeS('PVariantArray', '^TVariantArray // will not work');
  CL.AddTypeS('TVarFlag', '( vfByRef, vfVariant )');
  CL.AddTypeS('TVarFlags', 'set of TVarFlag');
  SIRegister_TCustomDataBlockInterpreter(CL);
  SIRegister_TSendDataBlock(CL);
  'CallSig','LongWord')( $D800);
  'ResultSig','LongWord')( $D400);
  'asMask','LongWord')( $00FF);
  CL.AddClassN(CL.FindClass('TOBJECT'),'EInterpreterError');
  CL.AddClassN(CL.FindClass('TOBJECT'),'ESocketConnectionError');
 Procedure CheckSignature( Sig : Integer)');
end;

procedure SIRegister_WinInet(CL: TPSPascalCompiler);
begin
  //CL.AddTypeS('HINTERNET', '___Pointer');
  CL.AddTypeS('HINTERNET1', 'THANDLE');
  CL.AddTypeS('HINTERNET', 'Integer');
  CL.AddTypeS('HINTERNET2', '___Pointer');
  //CL.AddTypeS('PHINTERNET', '^HINTERNET // will not work');
  //CL.AddTypeS('LPHINTERNET', 'PHINTERNET');
  CL.AddTypeS('INTERNET_PORT', 'Word');
  //CL.AddTypeS('PINTERNET_PORT', '^INTERNET_PORT // will not work');
  //CL.AddTypeS('LPINTERNET_PORT', 'PINTERNET_PORT');
  Function InternetTimeFromSystemTime(const pst:TSystemTime;dwRFC:DWORD;lpszTime:LPSTR;cbTime:DWORD):BOOL;
 'INTERNET_RFC1123_FORMAT','LongInt'( 0);
 'INTERNET_RFC1123_BUFSIZE','LongInt'( 30);
 Function InternetCrackUrl(lpszUrl:PChar; dwUrlLength,dwFlags:DWORD; var
lpUrlComponents:TURLComponents):BOOL;
 Function InternetCreateUrl(var lpUrlComponts:TURLCompons;dwFlags:DWORD;lpszUrl:PChar;var
dwUrlLength:DWORD):BOOL;
 Function InternetCloseHandle( hInet : HINTERNET) : BOOL');
 Function
InternetConnect(hInet:HINTERNET;lpszServerName:PChar;nServerPort:INTERNET_PORT;lpszUsername:PChar;
lpszPassword:PChar;dwService:DWORD;dwFlags:DWORD;dwContext:DWORD):HINTERNET;
 Function InternetOpenUrl(hInet:HINTERNET;lpszUrl:PChar;lpszHeaders:PChar;dwHeadLength:DWORD;dwFlags:DWORD
 ;dwContext:DWORD):HINTERNET;
 Function InternetOpen(lpszAgent:PChar;dwAccessType:DWORD;lpszProxy,
lpszProxBypass:PChar;dwFlags:DWORD):HINTERNET;
 Function InternetQueryDataAvailable(hFile:HINTERNET;var lpdwNumbOfBytesAvail:DWORD;dwFlags,
dwContext:DWORD):BOOL;
 Function InternetUnlockRequestFile( hLockRequestInfo : THANDLE) : BOOL');
 Function
InternetDial(hwndParent:HWND;lpszConnect:Pchar;dwFlags:DWORD;lpdwConnect:DWORD;dwReserved:DWORD):DWORD;
 Function InternetHangUp( dwConnection : DWORD; dwReserved : DWORD) : DWORD');
 Function InternetGoOnline( lpszURL : pchar; hwndParent : HWND; dwFlags : DWORD) : BOOL');
 Function InternetAutodial( dwFlags : DWORD; dwReserved : DWORD) : BOOL');
 Function InternetAutodialHangup( dwReserved : DWORD) : BOOL');
 Function InternetGetConnectedState( lpdwFlags : DWORD; dwReserved : DWORD) : BOOL');
 Function InternetCanonicalizeUrl(lpszUrl:PChar;lpszBuffer:PChar; var lpdwBufferLength:DWORD;dwFlags:DWORD):BOOL;
 Function InternetCombineUrl( lpszBaseUrl, lpszRelativeUrl : PChar; lpszBuffer : PChar; var lpdwBufferLength : Function InternetCloseHandle( hInet : HINTERNET) : BOOL');
 Function InternetConnect(hInet:HINTERNET;lpszServerName:PChar;nServerPort:INTERNET_PORT;lpszUsername:PChar
 ;lpszPassword:PChar;dwService:DWORD;dwFlags:DWORD; dwContext:DWORD):HINTERNET;
 Function InternetQueryDataAvailable(hFile:HINTERNET;var lpdwNumbOfBytesAvail:DWORD;dwFlgs, dwContext:DWORD):BOOL;
 Function FtpFindFirstFile( hConnect : HINTERNET; lpszSearchFile : PChar; var lpFindFileData : TWin32FindData; dwFlags : DWORD; dwContext : DWORD) : HINTERNET');
 Function WFtpGetFile( hConnect : HINTERNET; lpszRemoteFile : PChar; lpszNewFile : PChar; fFailIfExists : BOOL; dwFlagsAndAttributes : DWORD; dwFlags : DWORD; dwContext : DWORD) : BOOL');
 Function WFtpPutFile(hConct:HINTERNET;lpszLocFile:PChar;lpszNewRemFile:PChar;dwFlags:DWORD;dwCotx:DWORD):BOOL;
 Function FtpDeleteFile( hConnect : HINTERNET; lpszFileName : PChar) : BOOL');
 Function FtpRenameFile(hConnect:HINTERNET;lpszExisting:PChar;lpszNew:PChar):BOOL;
 Function FtpOpenFile(hConnect:HINTER;lpszFileName:PChar;dwAccess:DWORD;dwFlags:DWORD;dwContext:DWORD):HINTER;
 Function FtpCreateDirectory( hConnect : HINTERNET; lpszDirectory : PChar) : BOOL');
 Function FtpRemoveDirectory( hConnect : HINTERNET; lpszDirectory : PChar) : BOOL');
 Function FtpSetCurrentDirectory(hConnect:HINTERNET; lpszDirectory : PChar) : BOOL');
 Function FtpGetCurrentDirectory(hConnect:HINTER;lpszCurrentDir:PChar;var lpdwCurrentDir:DWORD):BOOL;
 Function FtpCommand(hConnect:HINTER;fExpectResponse:BOOL;dwFlags:DWORD;lpszCommd:PChar;dwContxt:DWORD):BOOL;
 Function IS_GOPHER_FILE( GopherType : DWORD) : BOOL');
 Function IS_GOPHER_DIRECTORY( GopherType : DWORD) : BOOL');
 Function IS_GOPHER_PHONE_SERVER( GopherType : DWORD) : BOOL');
 Function IS_GOPHER_ERROR( GopherType : DWORD) : BOOL');
 Function IS_GOPHER_INDEX_SERVER( GopherType : DWORD) : BOOL');
 Function IS_GOPHER_TELNET_SESSION( GopherType : DWORD) : BOOL');
 Function IS_GOPHER_BACKUP_SERVER( GopherType : DWORD) : BOOL');
 Function IS_GOPHER_TN3270_SESSION( GopherType : DWORD) : BOOL');
 Function IS_GOPHER_ASK( GopherType : DWORD) : BOOL');
 Function IS_GOPHER_PLUS( GopherType : DWORD) : BOOL');
 Function IS_GOPHER_TYPE_KNOWN( GopherType : DWORD) : BOOL');
 Function GopherCreateLocator(lpszHost:PChar;nServerPort:INTERNET_PORT;lpszDisplayString:PChar;lpszSelectorString:PChar; dwGopherType:DWORD;lpszLocator:PChar; var lpdwBufferLength:DWORD):BOOL;
 Function GopherGetLocatorType( lpszLocator : PChar; var lpdwGopherType : DWORD):BOOL');
 Function GopherOpenFile(hConect:HINTERNET;lpszLocat:PChar;lpszView:PChar;dwFlags:DWORD;dwContext:DWORD):HINTERNET;
 Function HttpOpenRequest( hConnect:HINTERNET; lpszVerb : PChar; lpszObjectName : PChar; lpszVersion : PChar; lpszReferrer : PChar; lplpszAcceptTypes : PLPSTR; dwFlags : DWORD; dwContext:DWORD):HINTERNET;
 Function HttpAddRequestHeaders(hReq:HINTERNET;lpszHeaders:PChar;dwHeadersLength:DWORD;dwModifiers:DWORD):BOOL;
 Function HttpSendRequest(hRequest: HINTERNET; lpszHeaders : PChar; dwHeadersLength:DWORD;lpOptional:Tobject; dwOptionalLength:DWORD):BOOL;
 Function InternetGetCookie(lpszUrl,lpszCookieName,lpszCookieData:PChar;var lpdwSize:DWORD):BOOL;
 Function InternetAttemptConnect( dwReserved : DWORD) : DWORD');
 Function InternetCheckConnection( lpszUrl : PChar; dwFlags : DWORD; dwReserved : DWORD) : BOOL');
 Function InternetErrorDlg(hWnd:HWND;hRequest:HINTERNET;dwError,dwFlags:DWORD;var lppvData:TObject):DWORD;
 Function InternetConfirmZoneCrossing( hWnd : HWND; szUrlPrev, szUrlNew : LPSTR; bPost : BOOL) : DWORD');
 Function CreateUrlCacheGroup( dwFlags : DWORD; lpReserved : TObject) : Int64');
 Function DeleteUrlCacheGroup( GroupId : Int64; dwFlags : DWORD; lpReserved : TObject) : Bool');
 Function FindFirstUrlCacheEntry(lpszUrlSearchPattern PChar;var lpFirstCacheEntryInfo:TInternetCacheEntryInfo;var lpdwFirstCacheEntryInfoBufferSize:DWORD):THandle;
 Function FindNextUrlCacheEntry(hEnumHandle:THandle;var lpNextCacheEntryInfo:TInternetCacheEntryInfo; var lpdwNextCacheEntryInfoBufferSize : DWORD) : BOOL;
 Function FindCloseUrlCache( hEnumHandle : THandle):BOOL;
 Function DeleteUrlCacheEntry( lpszUrlName : LPCSTR):BOOL;
 Function InternetDial(hwndParent:HWND;lpszConnect:Pchar;dwFlgs:DWORD;lpdwConnect:DWORD;dwReserved:DWORD):DWORD;
 Function InternetSetDialState( lpszConnectoid : PChar; dwState : DWORD; dwReserved: DWORD) : BOOL');
end;

procedure SIRegister_Wwstr(CL: TPSPascalCompiler);
begin
  AddTypeS('strCharSet', 'set of char');
  TwwGetWordOption',' (wwgwSkipLeadingBlanks, wwgwQuotesAsWords,wwgwStripQuotes , wwgwSpacesInWords);
  AddTypeS('TwwGetWordOptions', 'set of TwwGetWordOption');
 Procedure strBreakApart( s : string; delimeter : string; parts : TStrings)');
 Function strGetToken( s : string; delimeter : string; var APos : integer) : string');
 Procedure strStripPreceding( var s : string; delimeter : strCharSet)');
 Procedure strStripTrailing( var s : string; delimeter : strCharSet)');
 Procedure strStripWhiteSpace( var s : string)');
 Function strRemoveChar( str : string; removeChar : char) : string');
 Function wwstrReplaceChar( str : string; removeChar, replaceChar : char) : string');
 Function strReplaceCharWithStr( str : string; removeChar : char; replaceStr : string) : string');
 Function wwEqualStr( s1, s2 : string) : boolean');
 Function strCount( s : string; delimeter : char) : integer');
 Function strWhiteSpace : strCharSet');
 Function wwExtractFileNameOnly( const FileName : string) : string');
 Function wwGetWord(s:string; var APos:integer;Options:TwwGetWordOptions;DelimSet:strCharSet):string;
 Function strTrailing( s : string; delimeter : char) : string');
 Function strPreceding( s : string; delimeter : char) : string');
 Function wwstrReplace( s, Find, Replace : string) : string');
end;

procedure SIRegister_DataBkr(CL: TPSPascalCompiler);
```

```
begin
  SIRegister_TRemoteDataModule(CL);
  SIRegister_TCRemoteDataModule(CL);
 Procedure RegisterPooled( const ClassID : string; Max, Timeout : Integer; Singleton : Boolean)');
 Procedure UnregisterPooled( const ClassID : string)');
 Procedure EnableSocketTransport( const ClassID : string)');
 Procedure DisableSocketTransport( const ClassID : string)');
 Procedure EnableWebTransport( const ClassID : string)');
 Procedure DisableWebTransport( const ClassID : string)');
end;

procedure SIRegister_Mathbox(CL: TPSPascalCompiler);
begin
 Function mxArcCos( x : Real) : Real');
 Function mxArcSin( x : Real) : Real');
 Function Comp2Str( N : Comp) : String');
 Function Int2StrPad0( N : LongInt; Len : Integer) : String');
 Function Int2Str( N : LongInt) : String');
 Function mxIsEqual( R1, R2 : Double) : Boolean');
 Function LogXY( x, y : Real) : Real');
 Function Pennies2Dollars( C : Comp) : String');
 Function mxPower( X : Integer; Y : Integer) : Real');
 Function Real2Str( N : Real; Width, Places : integer) : String');
 Function mxStr2Comp( MyString : string) : Comp');
 Function mxStr2Pennies( S : String) : Comp');
 Function Str2Real( MyString : string) : Real');
 Function XToTheY( x, y : Real) : Real');
end;

//*************************Cindy Functions!******************************
procedure SIRegister_cyIndy(CL: TPSPascalCompiler);
begin
  CL.AddTypeS('TContentTypeMessage', '( cmPlainText, cmPlainText_Attach, cmHtml'
   +'_Attach, cmHtml_RelatedAttach, cmAlterText_Html, cmAlterText_Html_Attach, '
   +'cmAlterText_Html_RelatedAttach,cmAlterText_Html_Attach_RelatedAttach,cmReadNotification )');
 MessagePlainText','String 'text/plain');
 CL.AddConstantN('MessagePlainText_Attach','String 'multipart/mixed');
 MessageAlterText_Html','String 'multipart/alternative');
 MessageHtml_Attach','String 'multipart/mixed');
 MessageHtml_RelatedAttach','String 'multipart/related; type="text/html"');
 MessageAlterText_Html_Attach','String 'multipart/mixed');
 MessageAlterText_Html_RelatedAttach','String')('multipart/related;type="multipart/alternative"';
 MessageAlterText_Html_Attach_RelatedAttach','String 'multipart/mixed');
 MessageReadNotification','String').( 'multipart/report; report-type="disposition-notification"';
 Function ForceDecodeHeader( aHeader : String) : String');
 Function Base64_EncodeString( Value : String; const aEncoding : TEncoding) : string');
 Function Base64_DecodeToString( Value : String; const aBytesEncoding : TEncoding) : String;');
 Function Base64_DecodeToBytes( Value : String) : TBytes;');
 Function IdHttp_DownloadFile(aSrcUrlFile,aDestFile:String;const OnWorkEvent:TWorkEvent):Boolean;
 Function Get_MD5( const aFileName : string) : string');
 Function Get_MD5FromString( const aString : string) : string');
end;

procedure SIRegister_cySysUtils(CL: TPSPascalCompiler);
begin
 Function IsFolder( SRec : TSearchrec) : Boolean');
 Function isFolderReadOnly( Directory : String) : Boolean');
 Function DirectoryIsEmpty( Directory : String) : Boolean');
 Function DirectoryWithSubDir( Directory : String) : Boolean');
 Procedure GetSubDirs( FromDirectory : String; aList : TStrings)');
 Function DiskFreeBytes( Drv : Char) : Int64');
 Function DiskBytes( Drv : Char) : Int64');
 Function GetFileBytes( Filename : String) : Int64');
 Function GetFilesBytes( Directory, Filter : String) : Int64');
 SE_CREATE_TOKEN_NAME','String 'SeCreateTokenPrivilege');
 SE_ASSIGNPRIMARYTOKEN_NAME','String 'SeAssignPrimaryTokenPrivilege');
 SE_LOCK_MEMORY_NAME','String 'SeLockMemoryPrivilege');
 SE_INCREASE_QUOTA_NAME','String 'SeIncreaseQuotaPrivilege');
 SE_UNSOLICITED_INPUT_NAME','String 'SeUnsolicitedInputPrivilege');
 SE_MACHINE_ACCOUNT_NAME','String 'SeMachineAccountPrivilege');
 SE_TCB_NAME','String 'SeTcbPrivilege');
 SE_SECURITY_NAME','String 'SeSecurityPrivilege');
 SE_TAKE_OWNERSHIP_NAME','String 'SeTakeOwnershipPrivilege');
 SE_LOAD_DRIVER_NAME','String 'SeLoadDriverPrivilege');
 SE_SYSTEM_PROFILE_NAME','String 'SeSystemProfilePrivilege');
 SE_SYSTEMTIME_NAME','String 'SeSystemtimePrivilege');
 SE_PROF_SINGLE_PROCESS_NAME','String 'SeProfileSingleProcessPrivilege');
 SE_INC_BASE_PRIORITY_NAME','String 'SeIncreaseBasePriorityPrivilege');
 SE_CREATE_PAGEFILE_NAME','String 'SeCreatePagefilePrivilege');
 SE_CREATE_PERMANENT_NAME','String 'SeCreatePermanentPrivilege');
 SE_BACKUP_NAME','String 'SeBackupPrivilege');
 SE_RESTORE_NAME','String 'SeRestorePrivilege');
 SE_SHUTDOWN_NAME','String 'SeShutdownPrivilege');
 SE_DEBUG_NAME','String 'SeDebugPrivilege');
 SE_AUDIT_NAME','String 'SeAuditPrivilege');
 SE_SYSTEM_ENVIRONMENT_NAME','String 'SeSystemEnvironmentPrivilege');
 SE_CHANGE_NOTIFY_NAME','String 'SeChangeNotifyPrivilege');
 SE_REMOTE_SHUTDOWN_NAME','String 'SeRemoteShutdownPrivilege');
 SE_UNDOCK_NAME','String 'SeUndockPrivilege');
 SE_SYNC_AGENT_NAME','String 'SeSyncAgentPrivilege');
 SE_ENABLE_DELEGATION_NAME','String 'SeEnableDelegationPrivilege');
 SE_MANAGE_VOLUME_NAME','String 'SeManageVolumePrivilege');
end;


procedure SIRegister_cyWinUtils(CL: TPSPascalCompiler);
begin
  Type(TWindowsVersion', '( wvUnknown, wvWin31, wvWin95, wvWin98, wvWin'
   +'Me, wvWinNt3, wvWinNt4, wvWin2000, wvWinXP, wvWinVista, wvWin7, wvWin8, wvWin8_Or_Upper )');
 Function ShellGetExtensionName( FileName : String) : String');
 Function ShellGetIconIndex( FileName : String) : Integer');
 Function ShellGetIconHandle( FileName : String) : HIcon');
 Procedure ShellThreadCopy( App_Handle : THandle; fromFile : string; toFile : string)');
 Procedure ShellThreadMove( App_Handle : THandle; fromFile : string; toFile : string)');
 Procedure ShellRenameDir( DirFrom, DirTo : string)');
 Function cyShellExecute(Operation,FileName,Parameters,Directory:String;ShowCmd:Integer):Cardinal;
 Procedure cyShellExecute1(ExeFilename,Parameters,ApplicationName,ApplicationClass:String;Restore:Boolean);
 Procedure ShellExecuteAsModal( ExeFilename, ApplicationName, Directory : String)');
 Procedure ShellExecuteExAsAdmin( hWnd : HWND; Filename : string; Parameters : string)');
 Function ShellExecuteEx(aFileName: string; const Parameters:string; const Directory: string; const
WaitCloseCompletion : boolean) : Boolean');
 Procedure RestoreAndSetForegroundWindow( Hnd : Integer)');
 Function RemoveDuplicatedPathDelimiter( Str : String) : String');
 Function cyFileTimeToDateTime( _FT : TFileTime) : TDateTime');
 Function GetModificationDate( Filename : String) : TDateTime');
 Function GetCreationDate( Filename : String) : TDateTime');
 Function GetLastAccessDate( Filename : String) : TDateTime');
 Function FileDelete( Filename : String) : Boolean');
 Function FileIsOpen( Filename : string) : boolean');
 Procedure FilesDelete( FromDirectory : String; Filter : ShortString)');
 Function DirectoryDelete( Directory : String) : Boolean');
 Function GetPrinters( PrintersList : TStrings) : Integer');
 Procedure SetDefaultPrinter( PrinterName : String)');
 Procedure ShowDefaultPrinterWindowProperties( FormParent_Handle : Integer)');
 Function WinToDosPath( WinPathName : String) : String');
 Function DosToWinPath( DosPathName : String) : String');
 Function cyGetWindowsVersion : TWindowsVersion');
 Function NTSetPrivilege( sPrivilege : string; bEnabled : Boolean) : Boolean');
 Procedure WindowsShutDown( Restart : boolean)');
 Procedure CreateShortCut(FileSrc,Parametres,FileLnk,Descript,DossierDeTravail,
FileIcon:string;NumIcone:integer);
 Procedure GetWindowsFonts( FontsList : TStrings)');
 Function GetAvailableFilename( DesiredFileName : String): String');
end;

procedure SIRegister_cyStrUtils(CL: TPSPascalCompiler);
begin
  Type(TStrLocateOption', '( strloCaseInsensitive, strloPartialKey )');
  Type(TStrLocateOptions', 'set of TStrLocateOption');
  Type(TStringRead', '( srFromLeft, srFromRight )');
  Type(TStringReads', 'set of TStringRead');
  Type(TCaseSensitive', '( csCaseSensitive, csCaseNotSensitive )');
  Type(TWordsOption', '( woOnlyFirstWord, woOnlyFirstCar )');
  Type(TWordsOptions', 'set of TWordsOption');
  Type(TCarType', '(ctAlphabeticUppercase, ctAlphabeticLowercase, ctNumeric, ctOther )');
  Type(TCarTypes', 'set of TCarType');
  //CL.AddTypeS('TUnicodeCategories', 'set of TUnicodeCategory');
 CarTypeAlphabetic','LongInt'):= ord(ctAlphabeticUppercase) or ord(ctAlphabeticLowercase);
 Function Char_GetType( aChar : Char) : TCarType');
 Function SubString_Count( Str : String; Separator : Char) : Integer');
 Function SubString_AtPos( Str : String; Separator : Char; SubStringIndex : Word) : Integer');
 Function SubString_Get( Str : String; Separator : Char; SubStringIndex : Word) : String');
 Function SubString_Length( Str : String; Separator : Char; SubStringIndex : Word) : Integer');
 Procedure SubString_Add( var Str : String; Separator : Char; Value : String)');
 Procedure SubString_Insert(var Str:String;Separator:Char; SubStringIndex:Word; Value : String)');
 Procedure SubString_Edit(var Str:String;Separator:Char;SubStringIndex:Word; NewValue : String)');
 Function SubString_Remove(var Str:string; Separator:Char; SubStringIndex : Word) : Boolean');
 Function SubString_Locate(Str:string;Separator:Char;SubString:String;Options:TStrLocateOptions):Integer;
 Function SubString_Ribbon( Str : string; Separator : Char; Current: Word; MoveBy : Integer):Integer;');
 Function SubString_Ribbon1(Str:string;Separator:Char;Current:String; MoveBy:Integer):String;');
 Function String_Quote( Str : String) : String');
 Function String_GetCar( Str : String; Position : Word; ReturnCarIfNotExists : Char) : Char');
 Function String_ExtractCars(fromStr:String;CarTypes:TCarTypes;IncludeCars,ExcludeCars:String):String;
 Function String_GetWord( Str : String; StringRead : TStringRead) : String');
 Function String_GetInteger( Str : String; StringRead : TStringRead) : String');
 Function String_ToInt( Str : String) : Integer');
 Function String_Uppercase( Str : String; Options : TWordsOptions) : String');
 Function String_Lowercase( Str : String; Options : TWordsOptions) : String');
 Function String_Reverse( Str : String) : String');
 Function String_Pos(SubStr:String;Str:String;fromPos:Integer;CaseSensitive:TCaseSensitive)Integer;
 Function String_Pos1(SubStr:String; Str:String; StringRead:TStringRead; Occurrence:Word; CaseSensitive :
TCaseSensitive):Integer;');
 Function String_Copy( Str : String; fromIndex : Integer; toIndex : Integer) : String;');
 Function String_Copy1(Str:String;StringRead:TStringRead;UntilFind:String;_Inclusive:Boolean):String;
 Function String_Copy2(Str:String;Between1:String;Between1MustExist:Boolean; Between2:String;
Between2MustExist : Boolean; CaseSensitive : TCaseSensitive) : String;');
 Function String_Delete( Str : String; fromIndex : Integer; toIndex : Integer) : String;');
 Function String_Delete1( Str : String; delStr : String; CaseSensitive:TCaseSensitive) : String;');
 Function String_BoundsCut( Str : String; CutCar : Char; Bounds : TStringReads) : String');
 Function String_BoundsAdd( Str : String; AddCar : Char; ReturnLength : Integer) : String');
 Function String_Add(Str:String;StringRead:TStringRead;aCar:Char;ReturnLength:Integer):String');
 Function String_End( Str : String; Cars : Word) : String');
 Function String_Subst(OldStr:String; NewStr:String; Str:String; CaseSensitive : TCaseSensitive; AlwaysFindFromBeginning : Boolean) : String');
 Function String_SubstCar( Str : String; Old, New : Char) : String');
 Function String_Count( Str : String; SubStr : String; CaseSenSitive : TCaseSensitive) : Integer');
 Function String_SameCars(Str1,Str2:String;StopCount_IfDiferent:Bool;CaseSensitive:TCaseSensitive):Integer;
 Function String_IsNumbers( Str : String) : Boolean');
 Function SearchPos( SubStr : String; Str : String; MaxErrors : Integer) : Integer');
 Function StringToCsvCell( aStr : String) : String');
end;

procedure SIRegister_cyDateUtils(CL: TPSPascalCompiler);
begin
 Function LongDayName( aDate : TDate) : String');
 Function LongMonthName( aDate : TDate) : String');
 Function ShortYearOf( aDate : TDate) : byte');
 Function DateToStrYYYYMMDD( aDate : TDate) : String');
 Function StrYYYYMMDDToDate( aStr : String) : TDate');
 Function SecondsToMinutes( Seconds : Integer) : Double');
 Function MinutesToSeconds( Minutes : Double) : Integer');
 Function MinutesToHours( Minutes : Integer) : Double');
 Function HoursToMinutes( Hours : Double) : Integer');
 Function ShortTimeStringToTime( ShortTimeStr : String; const ShortTimeFormat : String) : TDateTime');
 Procedure cyAddMonths( var aMonth, aYear : Word; Months : Integer)');
 Function MergeDateWithTime( aDate : TDate; aTime : TDateTime) : TDateTime');
 Function GetMinutesBetween( DateTime1, DateTime2 : TDateTime) : Int64;');
 Function GetMinutesBetween1(From_ShortTimeStr,To_ShortTimeStr:String;const ShortTimeFormat:String):Int64;
 Function GetSecondsBetween( DateTime1, DateTime2 : TDateTime) : Int64;');
 Function IntersectPeriods(Period1Begin,Period1End,Period2Begin,Period2End:TDateTime; var RsltBegin : TDateTime; RsltEnd : TDateTime) : Boolean;');
 Function IntersectPeriods1(Period1Begin,Period1End,Period2Begin,Period2End:TDateTime):Boolean;
 Function TryToEncodeDate( Year, Month, Day : Integer; var RsltDate : TDateTime) : Boolean');
end;

procedure SIRegister_cyObjUtils(CL: TPSPascalCompiler);
begin
  Type(TStringsSortType', '( stNone, stStringSensitive, stStringInsensitive, stExtended )');
  Type(TStringsValueKind', '( skStringSensitive, skStringInsensitive, skExtended )');
  Type(TcyLocateOption', '( lCaseInsensitive, lPartialKey )');
  Type(TcyLocateOptions', 'set of TcyLocateOption');
 Function StringsLocate( aList : TStrings; Value : String; Options : TcyLocateOptions) : Integer;');
 Function StringsLocate1( aList : TStrings; Value : String; ValueKind : TStringsValueKind) : Integer;');
 Function StringsAdd(aList:TStrings;Value:String;Unique:Boolean; SortType:TStringsSortType) : Integer');
 Procedure StringsReplace(aList:TStrings;OldStr:String; NewStr:String;ValueKind : TStringsValueKind)');
 Procedure StringsSort( aList : TStrings; SortType : TStringsSortType)');
 Function TreeNodeLocate( ParentNode : TTreeNode; Value : String) : TTreeNode');
 Function TreeNodeLocateOnLevel( TreeView : TTreeView; OnLevel : Integer; Value : String) : TTreeNode');
 Function TreeNodeGetChildFromIndex(TreeView:TTreeView;ParentNode:TTreeNode;ChildIndex:Integer):TTreeNode');
 Function TreeNodeGetParentOnLevel( ChildNode : TTreeNode; ParentLevel : Integer) : TTreeNode');
 Procedure TreeNodeCopy(FromNode:TTreeNode;ToNode:TTreeNode;const CopyChildren:Boolean;const CopySubChildren:Bool;
 Procedure RichEditSetStr( aRichEdit : TRichEdit; FormatedString : String)');
 Procedure RichEditStringReplace( aRichEdit : TRichEdit; OldPattern, NewPattern : string; Flags : TReplaceFlags)');
 Function GetTopMostControlAtPos( FromControl : TWinControl; aControlPoint : TPoint) : TControl');
 Procedure cyCenterControl( aControl : TControl)');
 Function GetLastParent( aControl : TControl) : TWinControl');
 Function GetControlBitmap( aControl : TWinControl) : TBitmap');
 Function GetRichEditBitmap( aRichEdit : TRichEdit) : TBitmap');
end;

procedure SIRegister_cyBDE(CL: TPSPascalCompiler);
begin
 Function TablePackTable( Tab : TTable) : Boolean');
 Function TableRegenIndexes( Tab : TTable) : Boolean');
 Function TableShowDeletedRecords( Tab : TTable; Show : Boolean) : Boolean');
 Function TableUndeleteRecord( Tab : TTable) : Boolean');
 Function TableAddIndex(Tab:TTable;FieldName String;FieldExpression:String;IOpt:TIndexOptions):Bool;
 Function TableDeleteIndex( Tab : TTable; IndexFieldName : String) : Boolean');
 Function TableEmptyTable( Tab : TTable) : Boolean');
 Function TableFindKey( aTable : TTable; Value : String) : Boolean');
 Procedure TableFindNearest( aTable : TTable; Value : String)');
 Function TableCreate(Owner:TComponent;DataBaseName:ShortString;TableName:String;IndexName:ShortString;ReadOnly : Boolean):TTable;
 Function TableOpen(Tab:TTable;FileName:String;IndexFieldName:String;RecordIndexValue:Variant;GotoRecordIndexValue:Bool):Bool;
 Function DateToBDESQLDate( aDate : TDate; const DateFormat : String) : String');
end;

procedure SIRegister_cyClasses(CL: TPSPascalCompiler);
begin
  SIRegister_TcyRunTimeDesign(CL);
  SIRegister_TcyShadowText(CL);
  SIRegister_TcyBgPicture(CL);
  SIRegister_TcyGradient(CL);
  SIRegister_tcyBevel(CL);
  //CL.AddTypeS('TcyBevelClass', 'class of tcyBevel');
  SIRegister_tcyBevels(CL);
  SIRegister_TcyImagelistOptions(CL);
 Procedure cyDrawBgPicture( aCanvas : TCanvas; aRect : TRect; aBgPicture : TcyBgPicture)');
end;

procedure SIRegister_cyGraphics(CL: TPSPascalCompiler);
begin
 Procedure cyGradientFill( aCanvas : TCanvas; aRect : TRect; fromColor, toColor : TColor;
adgradOrientation : TdgradOrientation; Balance, AngleDegree : Word; balanceMode : TDgradBalanceMode;
Maxdegrade : Byte;'+
  SpeedPercent : Integer; const AngleClipRect : Boolean; const AngleBuffer : TBitmap)');
 Procedure cyGradientFillVertical(aCanvas:TCanvas;aRect:TRect;fromColor,toColor:TColor;MaxDegrad : byte);
 Procedure cyGradientFillHorizontal(aCanvas:TCanvas;aRect:TRect;fromColor,toColor:TColor;MaxDegrad:byte);
 Procedure cyGradientFillShape( aCanvas : TCanvas; aRect : TRect; fromColor, toColor : TColor; MaxDegrad :
Byte; toRect : TRect; OrientationShape : TDgradOrientationShape)');
 Procedure cyGradientFillAngle( aCanvas : TCanvas; aRect : TRect; fromColor, toColor : TColor; MaxDegrad :
Byte; AngleDegree : Word; const ClipRect : Boolean; const Buffer : TBitmap)');
 Procedure DrawRectangleInside( aCanvas : TCanvas; InsideRect : TRect; FrameWidth : Integer)');
 Procedure cyFrame( aCanvas : TCanvas; var InsideRect : TRect; Color : TColor; const Width : Integer);');
 Procedure cyFrame1( Canvas : TCanvas; var InsideRect : TRect; LeftColor, TopColor, RightColor,
BottomColor : TColor; const Width : Integer; const RoundRect : boolean);');
 Procedure cyFrame3D(Canvas:TCanvas;var Rect:TRect;TopLeftColor,
BottomRightColor:TColor;Width:Integer;const DrawLeft:Boolean;const DrawTop:Bool;const DrawRight:Bool;const
DrawBottom:Bool;const RoundRect:bool;
 Procedure cyDrawButtonFace(Canvas : TCanvas; var Rect : TRect; GradientColor1, GradientColor2 : TColor;
aState : TButtonState; Focused, Hot : Boolean)');
 Procedure cyDrawButton(Canvas:TCanvas; Caption:String;ARect:TRect; GradientColor1,GradientColor2:TColor;
aState : TButtonState; Focused, Hot : Boolean)');
 Procedure cyDrawSpeedButtonFace( Canvas : TCanvas; var Rect : TRect; GradientColor1, GradientColor2 :
TColor; aState : TButtonState; Focused, Hot : Boolean)');
 Procedure cyDrawSpeedButton( Canvas : TCanvas; Caption : String; ARect : TRect; GradientColor1,
GradientColor2 : TColor; aState : TButtonState; Focused, Hot : Boolean)');
 Procedure cyDrawCheckBox( Canvas : TCanvas; IsChecked : Boolean; ARect : TRect; const BgColor : TColor;
const DarkFrameColor : TColor; const LightFrameColor : TColor; const MarkColor : TColor)');
 Procedure cyDrawSingleLineText( Canvas : TCanvas; Text : String; ARect : TRect; Alignment : TAlignment;
TextLayout : TTextLayout; const IndentX : Integer; const IndentY : Integer)');
 Function DrawTextFormatFlags(aTextFormat:LongInt;Alignment
TAlignment;Layout:TTextLayout;WordWrap:Bool):LongInt;
 Function DrawTextFormatFlags1( aTextFormat : LongInt; Alignment : TAlignment; Layout : TTextLayout;
WordWrap : Boolean; CaptionRender : TCaptionRender) : LongInt;');
 Procedure cyDrawText( CanvasHandle : Cardinal; Text : String; var Rect : TRect; TextFormat : LongInt)');
 Function cyCreateFontIndirect( fromFont : TFont; Angle : Double) : TFont;');
 Function cyCreateFontIndirect1( fromFont : TFont; CaptionOrientation : TCaptionOrientation) : TFont;');
 Procedure cyDrawVerticalText( Canvas : TCanvas; Text : String; var Rect : TRect; TextFormat : Longint;
CaptionOrientation : TCaptionOrientation; Alignment : TAlignment; Layout : TTextLayout)');
 Function DrawLeftTurnPageEffect( Canvas : TCanvas; PageColor : TColor; PageRect : TRect; PercentDone :
Integer; const OnlyCalcFoldLine : Boolean) : TLineCoord');
 Function DrawRightTurnPageEffect( Canvas : TCanvas; PageColor : TColor; PageRect : TRect; PercentDone :
Integer; const OnlyCalcFoldLine : Boolean) : TLineCoord');
 Function PictureIsTransparentAtPos( aPicture : TPicture; aPoint : TPoint):boolean');
 Function IconIsTransparentAtPos( aIcon : TIcon; aPoint : TPoint) : boolean');
 Function MetafileIsTransparentAtPos( aMetafile : TMetafile; aPoint : TPoint) : boolean');
 Function PngImageIsTransparentAtPos( aPngImage : TPngImage; aPoint : TPoint) : boolean');
 Procedure DrawCanvas( Destination : TCanvas; DestRect : TRect; Source : TCanvas; SourceRect : TRect);');
 Procedure DrawCanvas1(Destination:TCanvas; DestRect:TRect;Src:TCanvas; SrcRect : TRect; TransparentColor
: TColor; const aStyle:TBgStyle; const aPosition:TBgPosition; const IndentX : Integer; const IndentY :
Integer;'+
 const IntervalX : Integer; const IntervalY : Integer; const RepeatX : Integer; const RepeatY :
Integer);');
 Procedure DrawGraphic( Destination : TCanvas; DestRect : TRect; aGraphic : TGraphic; SrcRect : TRect;
TransparentColor:TColor; const aStyle : TBgStyle; const aPosition : TBgPosition; const IndentX : Integer;
const IndentY:Integer;const IntervalX:Integer;const IntervalY:Integer;const RepeatX:Integer;const
RepeatY:Integer;
 Procedure DrawGraphic1(Destination : TCanvas; DestRect : TRect; aGraphic : TGraphic; Transparent :
Boolean; const aStyle : TBgStyle; const aPosition : TBgPosition; const IndentX : Integer; const IndentY :
Integer; const  IntervalX:Integer; const IntervalY : Integer; const RepeatX:Integer; const
RepeatY:Integer);');
 Procedure DrawMosaicPortion( Destination : TCanvas; Portion : TRect; Pattern : TBitmap)');
 Function ValidGraphic( aGraphic : TGraphic) : Boolean');
 Function ColorSetPercentBrightness( Color : TColor; PercentLight : Integer) : TColor');
 Function ColorModify( Color : TColor; incR, incG, incB : Integer) : TColor');
 Function ColorSetPercentContrast( Color : TColor; IncPercent : Integer) : TColor');
 Function ColorSetPercentPale( Color : TColor; IncPercent : integer) : TColor');
 Function MediumColor( Color1, Color2 : TColor) : TColor');
 Function ClientToScreenRect( aControl : TControl; aControlRect : TRect) : TRect');
 Function ScreenToClientRect( aControl : TControl; aScreenRect : TRect) : TRect');
 Function CombineRectKeepingCenterPosition( RectPos, AddRect : TRect) : TRect');
 Procedure InflateRectPercent( var aRect : TRect; withPercent : Double)');
 Function GetIntermediateRect( Rect1, Rect2 : TRect; Percent : Double) : TRect');
 Function GetProportionalRect( fromRect, InsideRect : TRect) : TRect');
 Function PointInRect( const aPt : TPoint; const aRect : TRect) : boolean');
 Function PointInEllipse( const aPt : TPoint; const aRect : TRect) : boolean');
 Function CanvasAcceleratorTextWidth( aCanvas : TCanvas; aText : String) : Integer');
end;

procedure SIRegister_cyTypes(CL: TPSPascalCompiler);
```

```
begin
  Type(TGlyphAlignment', '( gaLeft, gaCenter, gaRight )');
  Type(TGlyphLayout', '( glTop, glCenter, glBottom )');
  Type(TDisabledGlyphOptions', '( dgDoNotDraw, dgDrawNormal, dgDrawMonochrome )');
  Type(TCaptionRender', '( crNormal, crPathEllipsis, crEndEllipsis, crWordEllipsis )');
  Type(TCaptionOrientation', '( coHorizontal, coHorizontalReversed, coVertical, coVerticalReversed )');
  Type(TBgPosition', '( bgCentered, bgTopLeft, bgTopCenter, bgTopRight,'
   +' bgCenterRight, bgBottomRight, bgBottomCenter, bgBottomLeft, bgCenterLeft)');
  Type(TBgStyle', '( bgNone, bgNormal, bgMosaic, bgStretch, bgStretchProportional )');
  Type(TcyBevelCut', '( bcLowered, bcRaised, bcNone, bcTransparent, bcGradientToNext )');
  Type(TDgradOrientation', '( dgdVertical, dgdHorizontal, dgdAngle, dgdRadial, dgdRectangle )');
  Type(TDgradOrientationShape', '( osRadial, osRectangle )');
  Type(TDgradBalanceMode(bmNormal,bmMirror,bmReverse,bmReverseFromColor,bmInvertReverse,
bmInvertReverseFromColor);
  Type(TRunTimeDesignJob', '( rjNothing, rjMove, rjResizeTop, rjResizeBottom, rjResizeLeft,
rjResizeTopLeft,  rjResizeBottomLeft, rjResizeRight, rjResizeTopRight, rjResizeBottomRight )');
  Type(TLineCoord', 'record BottomCoord : TPoint; TopCoord : TPoint; end');
 cCaptionOrientationWarning','String')('Note that text orientation doesn''t work with all fonts!');
end;

procedure SIRegister_WinSvc(CL: TPSPascalCompiler);
begin
 Const SERVICES_ACTIVE_DATABASEA','String 'ServicesActive');
 SERVICES_ACTIVE_DATABASEW','String') 'ServicesActive');
 Const SERVICES_ACTIVE_DATABASE','String')' SERVICES_ACTIVE_DATABASEA');
 Const SERVICES_FAILED_DATABASEA','String' 'ServicesFailed');
 Const SERVICES_FAILED_DATABASEW','String' 'ServicesFailed');
 Const SERVICES_FAILED_DATABASE','String' 'SERVICES_FAILED_DATABASEA');
 Const SC_GROUP_IDENTIFIERA','String'). '+');
 Const SC_GROUP_IDENTIFIERW','String') '+');
 Const SC_GROUP_IDENTIFIER','string 'SC_GROUP_IDENTIFIERA');
 Const SERVICE_NO_CHANGE','LongWord $FFFFFFFF);
 Const SERVICE_ACTIVE','LongWord')( $00000001);
 Const SERVICE_INACTIVE','LongWord $00000002);
 Const SERVICE_CONTROL_STOP','LongWord $00000001);
 Const SERVICE_CONTROL_PAUSE','LongWord $00000002);
 Const SERVICE_CONTROL_CONTINUE','LongWord $00000003);
 Const SERVICE_CONTROL_INTERROGATE','LongWord $00000004);
 Const SERVICE_CONTROL_SHUTDOWN','LongWord $00000005);
 Const SERVICE_STOPPED','LongWord $00000001);
 Const SERVICE_START_PENDING','LongWord $00000002);
 Const SERVICE_STOP_PENDING','LongWord $00000003);
 Const SERVICE_RUNNING','LongWord $00000004);
 Const SERVICE_CONTINUE_PENDING','LongWord $00000005);
 Const SERVICE_PAUSE_PENDING','LongWord $00000006);
 Const SERVICE_PAUSED','LongWord $00000007);
 Const SERVICE_ACCEPT_STOP','LongWord $00000001);
 Const SERVICE_ACCEPT_PAUSE_CONTINUE','LongWord $00000002);
 Const SERVICE_ACCEPT_SHUTDOWN','LongWord $00000004);
 Const SC_MANAGER_CONNECT','LongWord $0001);
 Const SC_MANAGER_CREATE_SERVICE','LongWord $0002;
 Const SC_MANAGER_ENUMERATE_SERVICE','LongWord $0004);
 Const SC_MANAGER_LOCK','LongWord $0008);
 Const SC_MANAGER_QUERY_LOCK_STATUS','LongWord $0010);
 Const SC_MANAGER_MODIFY_BOOT_CONFIG','LongWord $0020);
 Const SERVICE_QUERY_CONFIG','LongWord $0001);
 Const SERVICE_CHANGE_CONFIG','LongWord $0002);
 Const SERVICE_QUERY_STATUS','LongWord $0004);
 Const SERVICE_ENUMERATE_DEPENDENTS','LongWord $0008);
 Const SERVICE_START','LongWord $0010);
 Const SERVICE_STOP','LongWord $0020);
 Const SERVICE_PAUSE_CONTINUE','LongWord $0040);
 Const SERVICE_INTERROGATE','LongWord $0080);
 Const SERVICE_USER_DEFINED_CONTROL','LongWord $0100);
 Const SERVICE_KERNEL_DRIVER','LongWord $00000001);
 Const SERVICE_FILE_SYSTEM_DRIVER','LongWord $00000002);
 Const SERVICE_ADAPTER','LongWord $00000004);
 Const SERVICE_RECOGNIZER_DRIVER','LongWord $00000008);
 Const SERVICE_WIN32_OWN_PROCESS','LongWord $00000010);
 Const SERVICE_WIN32_SHARE_PROCESS','LongWord $00000020);
 Const SERVICE_INTERACTIVE_PROCESS','LongWord $00000100);
 Const SERVICE_BOOT_START','LongWord $00000000);
 Const SERVICE_SYSTEM_START','LongWord $00000001);
 Const SERVICE_AUTO_START','LongWord $00000002);
 Const SERVICE_DEMAND_START','LongWord $00000003);
 Const SERVICE_DISABLED','LongWord $00000004);
 Const SERVICE_ERROR_IGNORE','LongWord $00000000);
 Const SERVICE_ERROR_NORMAL','LongWord $00000001);
 Const SERVICE_ERROR_SEVERE','LongWord $00000002);
 Const SERVICE_ERROR_CRITICAL','LongWord $00000003);
  CL.AddTypeS('SC_HANDLE', 'THandle');
  //CL.AddTypeS('LPSC_HANDLE', '^SC_HANDLE // will not work');
  CL.AddTypeS('SERVICE_STATUS_HANDLE', 'DWORD');
  Const _SERVICE_STATUS', 'record dwServiceType : DWORD; dwCurrentState '
   +': DWORD; dwControlsAccepted : DWORD; dwWin32ExitCode : DWORD; dwServiceSpe'
   +'cificExitCode : DWORD; dwCheckPoint : DWORD; dwWaitHint : DWORD; end');
  Const SERVICE_STATUS', '_SERVICE_STATUS');
  Const TServiceStatus', '_SERVICE_STATUS');
  CL.AddTypeS('_ENUM_SERVICE_STATUSA', 'record lpServiceName : PChar; lpDis'
   +'playName : PChar; ServiceStatus : TServiceStatus; end');
  ENUM_SERVICE_STATUSA', '_ENUM_SERVICE_STATUS');
  _ENUM_SERVICE_STATUS', '_ENUM_SERVICE_STATUSA');
  TEnumServiceStatusA', '_ENUM_SERVICE_STATUSA');
  TEnumServiceStatus', 'TEnumServiceStatusA');
  SC_LOCK', '___Pointer');
  _QUERY_SERVICE_LOCK_STATUSA', 'record fIsLocked:DWORD; lpLockOner: PChar;dwLockDuration:DWORD;end;
  _QUERY_SERVICE_LOCK_STATUS', '_QUERY_SERVICE_LOCK_STATUSA');
  QUERY_SERVICE_LOCK_STATUSA', '_QUERY_SERVICE_LOCK_STATUSA');
  QUERY_SERVICE_LOCK_STATUS', 'QUERY_SERVICE_LOCK_STATUSA');
  TQueryServiceLockStatusA', '_QUERY_SERVICE_LOCK_STATUSA');
  //TQueryServiceLockStatusW', '_QUERY_SERVICE_LOCK_STATUSW');
  TQueryServiceLockStatus', 'TQueryServiceLockStatusA');
  _QUERY_SERVICE_CONFIGA', 'record dwServiceType : DWORD; dwStartT'
   +'ype : DWORD; dwErrorControl : DWORD; lpBinaryPathName : PChar; lpLoadO'
   +'rderGroup : PChar; dwTagId : DWORD; lpDependencies : PChar; lpServ'
   +'iceStartName : PChar; lpDisplayName : PChar; end');
  _QUERY_SERVICE_CONFIG', '_QUERY_SERVICE_CONFIGA');
  QUERY_SERVICE_CONFIGA', '_QUERY_SERVICE_CONFIGA');
  QUERY_SERVICE_CONFIG', 'QUERY_SERVICE_CONFIGA');
  TQueryServiceConfigA', '_QUERY_SERVICE_CONFIGA');
  TQueryServiceConfig', 'TQueryServiceConfigA');
 Function CloseServiceHandle( hSCObject : SC_HANDLE) : BOOL');
 Function ControlService(hService:SC_HANDLE;dwControl DWORD;var lpServiceStatus:TServiceStatus):BOOL;
 Function CreateService( hSCManager : SC_HANDLE; lpServiceName, lpDisplayName : PChar; dwDesiredAccess,
dwServiceType, dwStartType, dwErrorControl : DWORD; lpBinaryPathName, lpLoadOrderGroup : PChar;'
 +' lpdwTagId : DWORD; lpDependencies, lpServiceStartName, lpPassword : PChar) : SC_HANDLE');
 Function CreateServiceA( hSCManager : SC_HANDLE; lpServiceName, lpDisplayName : PChar; dwDesiredAccess,
dwServiceType, dwStartType, dwErrorControl : DWORD; lpBinaryPathName, lpLoadOrderGroup : PChar; '
 +'lpdwTagId : DWORD; lpDependencies, lpServiceStartName, lpPassword : PChar) : SC_HANDLE');
 Function DeleteService( hService : SC_HANDLE) : BOOL');
 Function EnumDependentServices( hService : SC_HANDLE; dwServiceState : DWORD; var lpServices :
TEnumServiceStatus; cbBufSize : DWORD; var pcbBytesNeeded, lpServicesReturned : DWORD) : BOOL');
 Function EnumServicesStatus( hSCManager : SC_HANDLE; dwServiceType, dwServiceState : DWORD; var
lpServices : TEnumServiceStatus; cbBufSize : DWORD; var pcbBytesNeeded, lpServicesReturned, lpResumeHandle
: DWORD):BOOL');
 Function GetServiceKeyName(hSCManager:SC_HANDLE;pDisplayName,lpServiceName:PChar;var
lpcchBuffer:DWORD):BOOL');
 Function GetServiceDisplayName(hSCManager:SC_HANDLE;lpServiceName,lpDisplayName:PChar;var
lpcchBuffer:DWORD):BOOL;
 Function LockServiceDatabase( hSCManager : SC_HANDLE) : SC_LOCK');
 Function NotifyBootConfigStatus( BootAcceptable : BOOL) : BOOL');
 Function OpenSCManager( lpMachineName, lpDatabaseName : PChar; dwDesiredAccess : DWORD) : SC_HANDLE');
 Function OpenService( hSCManager : SC_HANDLE; lpServiceName : PChar; dwDesiredAccess : DWORD) :
SC_HANDLE');
 Function QueryServiceLockStatus( hSCManager : SC_HANDLE; var lpLockStatus : TQueryServiceLockStatus;
cbBufSize : DWORD; var pcbBytesNeeded : DWORD) : BOOL');
 Function QueryServiceStatus( hService : SC_HANDLE; var lpServiceStatus : TServiceStatus) : BOOL');
 Function SetServiceStatus( hServiceStatus : SERVICE_STATUS_HANDLE; var lpServiceStatus : TServiceStatus)
: BOOL');
 Function StartService( hService : SC_HANDLE; dwNumServiceArgs : DWORD; var lpServiceArgVectors : PChar) :
BOOL');
 Function UnlockServiceDatabase( ScLock : SC_LOCK) : BOOL');
end;

procedure SIRegister_JvPickDate(CL: TPSPascalCompiler);
begin
 CL.AddDelphiFunction('Function SelectDate( Sender : TWinControl; var Date : TDateTime; const DlgCaption :
TCaption; AStartOfWeek : TDayOfWeekName; AWeekends : TDaysOfWeek; AWeekendColor : TColor; BtnHints :
TStrings; MinDate : TDateTime; MaxDate : TDateTime) : Boolean');
 Function SelectDateStr(Sender:TWinControl;var StrDate:string;const
DlgCaption:TCaption;AStartOfWeek:TDayOfWeekName; AWeekends:TDaysOfWeek;
AWeekendColor:TColor;BtnHints:TStrings;MinDate:TDateTime;MaxDate:TDateTime):Boolean;
 Function PopupDate(var Date:TDateTime; Edit:TWinControl; MinDate:TDateTime; MaxDate:TDateTime):Boolean;
 Function CreatePopupCalendar(AOwner:TComponent;ABiDiMode:TBiDiMode;MinDate:TDateTime;MaxDate:TDateTime):
TWinControl;
 Procedure SetupPopupCalendar( PopupCalendar : TWinControl; AStartOfWeek : TDayOfWeekName; AWeekends :
TDaysOfWeek; AWeekendColor : TColor; BtnHints : TStrings; FourDigitYear : Boolean; MinDate : TDateTime;
MaxDate : TDateTime)');
 Function CreateNotifyThread( const FolderName : string; WatchSubtree : Boolean; Filter :
TFileChangeFilters) : TJvNotifyThread');
end;

procedure SIRegister_JclNTFS2(CL: TPSPascalCompiler);
begin
  CL.AddClassN(CL.FindClass('TOBJECT'),'EJclNtfsError');
  CL.AddTypeS('TFileCompressionState', '( fcNoCompression, fcDefaultCompression, fcLZNT1Compression )');
 Function NtfsGetCompression2( const FileName : TFileName; var State : Short) : Boolean;');
 Function NtfsGetCompression12( const FileName : TFileName) : TFileCompressionState;');
 Function NtfsSetCompression2( const FileName : TFileName; const State : Short) : Boolean');
 Procedure NtfsSetFileCompression2(const FileName : TFileName; const State : TFileCompressionState)');
 Procedure NtfsSetDirectoryTreeCompression2(const Directory:string; const State:TFileCompressionState)');
 Procedure NtfsSetDefaultFileCompression2(const Directory : string; const State:TFileCompressionState)');
 Procedure NtfsSetPathCompression2(const Path:string;const State:TFileCompressionState;Recursive:Bool)');
 Function NtfsSetSparse2( const FileName : string) : Boolean');
 Function NtfsZeroDataByHandle2( const Handle : THandle; const First, Last : Int64) : Boolean');
 Function NtfsZeroDataByName2( const FileName : string; const First, Last : Int64) : Boolean');
 Function NtfsSparseStreamsSupported2( const Volume : string) : Boolean');
 Function NtfsGetSparse2( const FileName : string) : Boolean');
 Function NtfsDeleteReparsePoint2( const FileName : string; ReparseTag : DWORD) : Boolean');
 Function NtfsSetReparsePoint2( const FileName : string; var ReparseData, Size : Longword) : Boolean');
 Function NtfsGetReparseTag2( const Path : string; var Tag : DWORD) : Boolean');
 Function NtfsReparsePointsSupported2( const Volume : string) : Boolean');
 Function NtfsFileHasReparsePoint2( const Path : string) : Boolean');
 Function NtfsIsFolderMountPoint2( const Path : string) : Boolean');
 Function NtfsMountDeviceAsDrive2( const Device : WideString; Drive : Char) : Boolean');
 Function NtfsMountVolume2( const Volume : WideChar; const MountPoint : WideString) : Boolean');
  CL.AddTypeS('TOpLock', '( olExclusive, olReadOnly, olBatch, olFilter )');
 Function NtfsOpLockAckClosePending2( Handle : THandle; Overlapped : TOverlapped) : Boolean');
 Function NtfsOpLockBreakAckNo22( Handle : THandle; Overlapped : TOverlapped) : Boolean');
 Function NtfsOpLockBreakAcknowledge2( Handle : THandle; Overlapped : TOverlapped) : Boolean');
 Function NtfsOpLockBreakNotify2( Handle : THandle; Overlapped : TOverlapped) : Boolean');
 Function NtfsRequestOpLock2( Handle : THandle; Kind : TOpLock; Overlapped : TOverlapped) : Boolean');
 Function NtfsCreateJunctionPoint2( const Source, Destination : string) : Boolean');
 Function NtfsDeleteJunctionPoint2( const Source : string) : Boolean');
 Function NtfsGetJunctionPointDestination2( const Source : string; var Destination : string) : Boolean');
  CL.AddTypeS('TStreamId', '( siInvalid, siStandard, siExtendedAttribute, siSec'
   +'urity, siAlternate, siHardLink, siProperty, siObjectIdentifier, siReparsePoints, siSparseFile )');
  CL.AddTypeS('TStreamIds', 'set of TStreamId');
  TInternalFindStreamData', 'record FileHandle:THandle; Context: TObject; StreamIds : TStreamIds; end');
  CL.AddTypeS('TFindStreamData', 'record Internal : TInternalFindStreamData; At'
  +'tributes : DWORD; StreamID : TStreamId; Name: WideString; Size : Int64; end');
 Function NtfsFindFirstStream2(const FileName:string;StreamIds:TStreamIds;var Data:TFindStreamData):Bool;
 Function NtfsFindNextStream2( var Data : TFindStreamData) : Boolean');
 Function NtfsFindStreamClose2( var Data : TFindStreamData) : Boolean');
 Function NtfsCreateHardLink2( const LinkFileName, ExistingFileName : String) : Boolean');
 Function NtfsCreateHardLinkA2( const LinkFileName, ExistingFileName : AnsiString) : Boolean');
 Function NtfsCreateHardLinkW2( const LinkFileName, ExistingFileName : WideString) : Boolean');
  CL.AddTypeS('TNtfsHardLinkInfo', 'record LinkCount : Cardinal; FileIndex : Int64; end');
 Function NtfsGetHardLinkInfo2( const FileName : string; var Info : TNtfsHardLinkInfo) : Boolean');
 Function NtfsFindHardLinks2(const Path:string;const FileIndexHigh,FileIndexLow:Card;const
List:TStrings):Bool
 Function NtfsDeleteHardLinks2( const FileName : string) : Boolean');
  FindClass('TOBJECT'),'EJclFileSummaryError');
  TJclFileSummaryAccess', '( fsaRead, fsaWrite, fsaReadWrite )');
  TJclFileSummaryShare', '( fssDenyNone, fssDenyRead, fssDenyWrite, fssDenyAll )');
  TJclFileSummaryPropSetCallback', 'Function ( const FMTID : TGUID) : Boolean');
  CL.AddClassN(CL.FindClass('TOBJECT'),'TJclFileSummary');
  SIRegister_TJclFilePropertySet(CL);
  //CL.AddTypeS('TJclFilePropertySetClass', 'class of TJclFilePropertySet');
  SIRegister_TJclFileSummary(CL);
  SIRegister_TJclFileSummaryInformation(CL);
  SIRegister_TJclDocSummaryInformation(CL);
  SIRegister_TJclMediaFileSummaryInformation(CL);
  SIRegister_TJclMSISummaryInformation(CL);
  SIRegister_TJclShellSummaryInformation(CL);
  SIRegister_TJclStorageSummaryInformation(CL);
  SIRegister_TJclImageSummaryInformation(CL);
  SIRegister_TJclDisplacedSummaryInformation(CL);
  SIRegister_TJclBriefCaseSummaryInformation(CL);
  SIRegister_TJclMiscSummaryInformation(CL);
  SIRegister_TJclWebViewSummaryInformation(CL);
  SIRegister_TJclMusicSummaryInformation(CL);
  SIRegister_TJclDRMSummaryInformation(CL);
  SIRegister_TJclVideoSummaryInformation(CL);
  SIRegister_TJclAudioSummaryInformation(CL);
  SIRegister_TJclControlPanelSummaryInformation(CL);
  SIRegister_TJclVolumeSummaryInformation(CL);
  SIRegister_TJclShareSummaryInformation(CL);
  SIRegister_TJclLinkSummaryInformation(CL);
  SIRegister_TJclQuerySummaryInformation(CL);
  SIRegister_TJclImageInformation(CL);
  SIRegister_TJclJpegSummaryInformation(CL);
end;

procedure SIRegister_Jcl8087(CL: TPSPascalCompiler);
begin
  AddTypeS('T8087Precision', '( pcSingle, pcReserved, pcDouble, pcExtended )');
  T8087Rounding','( rcNearestOrEven, rcDownInfinity, rcUpInfinity, rcChopOrTruncate )');
  T8087Infinity','( icProjective, icAffine )');
  T8087Exception','(emInvalidOp,emDenormalizedOperand,emZeroDivide,emOverflow,emUnderflow,emPrecision;
  CL.AddTypeS('T8087Exceptions', 'set of T8087Exception');
 Function Get8087ControlWord : Word');
 Function Get8087Infinity : T8087Infinity');
 Function Get8087Precision : T8087Precision');
 Function Get8087Rounding : T8087Rounding');
 Function Get8087StatusWord( ClearExceptions : Boolean) : Word');
 Function Set8087Infinity( const Infinity : T8087Infinity) : T8087Infinity');
 Function Set8087Precision( const Precision : T8087Precision) : T8087Precision');
 Function Set8087Rounding( const Rounding : T8087Rounding) : T8087Rounding');
 Function Set8087ControlWord( const Control : Word) : Word');
 Function ClearPending8087Exceptions : T8087Exceptions');
 Function GetPending8087Exceptions : T8087Exceptions');
 Function GetMasked8087Exceptions : T8087Exceptions');
 Function SetMasked8087Exceptions(Exceptions: T8087Exceptions;ClearBefore:Boolean):T8087Exceptions');
 Function Mask8087Exceptions( Exceptions:T8087Exceptions) : T8087Exceptions');
 Function Unmask8087Exceptions(Exceptions:T8087Exceptions;ClearBefore : Boolean) : T8087Exceptions');
end;
```

```



Functions_max hex in the box maXbox
functionslist.txt
FunctionsList1 3.9.9.86/88/91/92/94

****************************************************************************
Procedure
PROCEDURE SIZE 7507 7401 6792 6310 5971 4438 3797 3600
Procedure ****************************Now the Procedure list*******************
Procedure ( ACol, ARow : Integer; Items : TStrings)
Procedure ( Agg : TAggregate)
Procedure ( ASender : TComponent; const AReplyStatus : TReplyStatus)
Procedure ( ASender : TComponent; const AString : String; var AMsg : TIdMessage)
Procedure ( ASender : TComponent; var AMsg : TIdMessage)
Procedure ( ASender : TObject; const ABytes : Integer)
Procedure ( ASender : TObject; VStream : TStream)
Procedure ( AThread : TIdThread)
Procedure ( AWebModule : TComponent)
Procedure ( Column : TColumn)
Procedure ( const AUsername : String; const APassword : String; AAuthenticationResult : Boolean)
Procedure ( const iStart : integer; const sText : string)
Procedure ( Control : TCustomTabControl; TabIndex : Integer; const Rect : TRect; Active : Boolean)
Procedure ( Database : TDatabase; LoginParams : TStrings)
Procedure (DataSet:TCustomClientDataSet;E:EReconcileError;UpdateKind:TUpdateKind;var Action: TReconcileAction)
Procedure ( DATASET : TDATASET)
Procedure ( DataSet:TDataSet; E:EDatabaseError;UpdateKind:TUpdateKind;var UpdateAction: TUpdateAction)
Procedure ( DATASET : TDATASET; E : TObject; var ACTION : TDATAACTION)
Procedure ( DataSet : TDataSet; UpdateKind : TUpdateKind; var UpdateAction : TUpdateAction)
Procedure ( DATASET : TDATASET; var ACCEPT : BOOLEAN)
Procedure ( DBCtrlGrid : TDBCtrlGrid; Index : Integer)
Procedure ( Done : Integer)
Procedure ( HeaderControl : TCustomHeaderControl; Section : THeaderSection)
Procedure (HeaderControl:TCustomHeaderControl;Section:THeaderSection; const Rect:TRect;Pressed:Boolean)
Procedure (HeaderControl:TCustomHeaderControl;Section:THeaderSection;Width:Integer;State:TSectionTrackState)
Procedure ( HeaderControl : THeaderControl; Section : THeaderSection)
Procedure (HeaderControl:THeaderControl;Section: THeaderSection; const Rect:TRect; Pressed : Boolean)
Procedure (HeaderControl:THeaderControl;Section:THeaderSection;Width:Integer;State : TSectionTrackState)
Procedure (Sender: TCustomListView;const ARect: TRect;Stage:TCustomDrawStage;var DefaultDraw: Boolean)
Procedure ( Sender : TCustomListView; const ARect : TRect; var DefaultDraw : Boolean)
Procedure ( Sender : TCustomListView; Item : TListItem; Rect : TRect; State : TOwnerDrawState)
Procedure ( Sender : TCustomTreeView; const ARect : TRect; var DefaultDraw : Boolean)
Procedure ( SENDER : TFIELD; const TEXT : String)
Procedure ( SENDER : TFIELD; var TEXT : STRING; DISPLAYTEXT : BOOLEAN)
Procedure ( Sender : TIdTelnet; const Buffer : String)
Procedure ( Sender : TIdTelnet; Status : TIdTelnetCommand)
Procedure ( SENDER : TOBJECT; ACANVAS : TCANVAS; ARECT : TRECT; SELECTED : BOOLEAN)
Procedure ( SENDER : TOBJECT; ACANVAS : TCANVAS; ARECT : TRECT; STATE : TOWNERDRAWSTATE)
Procedure ( SENDER : TOBJECT; ACANVAS : TCANVAS; var WIDTH, HEIGHT : INTEGER)
Procedure ( Sender : TObject; ACol, ARow : Longint; const Value : string)
Procedure ( Sender : TObject; ACol, ARow : Longint; Rect : TRect; State : TGridDrawState)
Procedure ( Sender : TObject; ACol, ARow : Longint; var CanSelect : Boolean)
Procedure ( Sender : TObject; ACol, ARow : Longint; var Value : string)
Procedure ( Sender : TObject; Button : TMPBtnType)
Procedure ( Sender : TObject; Button : TMPBtnType; var DoDefault : Boolean)
Procedure ( Sender : TObject; Button : TUDBtnType)
Procedure ( Sender : TObject; Canvas : TCanvas; PageRect : TRect; var DoneDrawing : Boolean)
Procedure ( Sender: TObject; ClientSocket: TServerClientWinSocket; var SocketThread : TServerClientThread)
Procedure ( Sender : TObject; Column : TListColumn)
Procedure ( Sender : TObject; Column : TListColumn; Point : TPoint)
Procedure ( Sender : TObject; Connecting : Boolean)
Procedure (Sender:TObject;const PapSize:SmallInt;const Orient:TPrtOrient;const PageTy:TPageTy;var DoneDraw:Bool
Procedure (Sender:TObject; const Rect : TRect; DataCol : Integer; Column: TColumn; State : TGridDrawState)
Procedure ( Sender : TObject; const Rect : TRect; Field : TField; State : TGridDrawState)
Procedure (Sender: TObject; const UserString:string;var DateAndTime:TDateTime;var AllowChange:Boolean)
Procedure ( Sender : TObject; E : Exception; var Handled : Boolean)
Procedure ( Sender : TObject; FromIndex, ToIndex : Longint)
Procedure ( Sender : TObject; FromSection, ToSection : THeaderSection; var AllowDrag : Boolean)
Procedure ( Sender : TObject; Index : LongInt)
Procedure ( Sender : TObject; Item : TListItem)
Procedure ( Sender : TObject; Item : TListItem; Change : TItemChange)
Procedure ( Sender : TObject; Item : TListItem; Change : TItemChange; var AllowChange : Boolean)
Procedure ( Sender : TObject; Item : TListItem; Selected : Boolean)
Procedure ( Sender : TObject; Item : TListItem; var AllowEdit : Boolean)
Procedure ( Sender : TObject; Item : TListItem; var S : string)
Procedure ( Sender : TObject; Item1, Item2 : TListItem; Data : Integer; var Compare : Integer)
Procedure ( Sender : TObject; ModalResult : TModalResult; var CanClose : Boolean)
Procedure ( Sender : TObject; Month : LongWord; var MonthBoldInfo : LongWord)
Procedure ( Sender : TObject; NewTab : Integer; var AllowChange : Boolean)
Procedure ( Sender : TObject; Node : TTreeNode)
Procedure ( Sender : TObject; Node : TTreeNode; var AllowChange : Boolean)
Procedure ( Sender : TObject; Node : TTreeNode; var AllowCollapse : Boolean)
Procedure ( Sender : TObject; Node : TTreeNode; var AllowEdit : Boolean)
Procedure ( Sender : TObject; Node : TTreeNode; var AllowExpansion : Boolean)
Procedure ( Sender : TObject; Node : TTreeNode; var S : string)
Procedure ( Sender : TObject; Node1, Node2 : TTreeNode; Data : Integer; var Compare : Integer)
Procedure ( Sender : TObject; NumObjects, NumChars : Integer; var SaveClipboard : Boolean)
Procedure ( Sender : TObject; Rect : TRect)
Procedure ( Sender : TObject; Request : TWebRequest; Response : TWebResponse; var Handled : Boolean)
Procedure (Sender:TObject;Shift:TShiftState;X,Y:Integer;Orient:TPageScrollerOrientation;var Delta:Int
Procedure ( Sender : TObject; Socket : TCustomWinSocket)
Procedure (Sender:TObject; Socket:TCustomWinSocket;ErrorEvent: TErrorEvent; var ErrorCode : Integer)
Procedure ( Sender : TObject; Socket : TCustomWinSocket; SocketEvent : TSocketEvent)
Procedure ( Sender : TObject; Socket : TSocket; var ClientSocket : TServerClientWinSocket)
Procedure ( SENDER : TOBJECT; SOURCE : TMENUITEM; REBUILD : BOOLEAN)
Procedure ( Sender : TObject; StartPos, EndPos : Integer; var AllowChange : Boolean)
Procedure ( Sender : TObject; TabCanvas : TCanvas; R : TRect; Index : Integer; Selected : Boolean)
Procedure ( Sender : TObject; TabIndex : Integer; var ImageIndex : Integer)
Procedure ( Sender : TObject; Thread : TServerClientThread)
Procedure ( Sender : TObject; TickCount : Cardinal; var Reset : Boolean)
Procedure ( Sender : TObject; Username, Password : string)
Procedure ( Sender : TObject; var AllowChange : Boolean)
Procedure ( Sender : TObject; var AllowChange : Boolean; NewValue : SmallInt; Direction:TUpDownDirection)
Procedure ( Sender : TObject; var Caption : string; var Alignment : THTMLCaptionAlignment)
Procedure ( Sender : TObject; var Continue : Boolean)
Procedure (Sender:TObject;var dest:string;var NumRedirect:Int;var Handled:bool; var VMethod:TIdHTTPMethod)
Procedure ( Sender : TObject; var Username : string)
Procedure ( Sender : TObject; Wnd : HWND)
Procedure ( Sender : TToolbar; Button : TToolButton)
Procedure ( Sender : TToolBar; const ARect : TRect; Stage : TCustomDrawStage; var DefaultDraw : Boolean)
Procedure ( Sender : TToolBar; const ARect : TRect; var DefaultDraw : Boolean)
Procedure ( Sender : TToolbar; Index : Integer; var Allow : Boolean)
Procedure ( Sender : TToolbar; Index : Integer; var Button : TToolButton)
Procedure ( StatusBar : TCustomStatusBar; Panel : TStatusPanel; const Rect : TRect)
Procedure ( StatusBar : TStatusBar; Panel : TStatusPanel; const Rect : TRect)
Procedure (var FieldNames:TWideStrings;SQL:WideString; var BindAllFields:Boolean;var TableName: WideString)
Procedure ( var FieldNames : TWideStrings; SQL : WideString; var TableName : WideString)
procedure (Sender: TObject)
procedure (Sender: TObject; var Done: Boolean)
procedure (Sender: TObject; var Key: Word; Shift: TShiftState);
procedure _T(Name: tbtString; v: Variant);
Procedure AbandonSignalHandler( RtlSigNum : Integer)
Procedure Abort
Procedure About1Click( Sender : TObject)
Procedure Accept( Socket : TSocket)
Procedure AESSymetricExecute(const plaintext, ciphertext, password: string)
Procedure AESEncryptFile(const plaintext, ciphertext, password: string)
Procedure AESDecryptFile(const replaintext, ciphertext, password: string)
Procedure AESEncryptString(const plaintext: string; var ciphertext: string; password: string)
Procedure AESDecryptString(var plaintext: string; const ciphertext: string; password: string)
Procedure Add( Addend1, Addend2 : TMyBigInt)
Procedure ADD( const AKEY, AVALUE : VARIANT)
Procedure Add( const Key : string; Value : Integer)
Procedure ADD( const NAME, FIELDS : String; OPTIONS : TINDEXOPTIONS)
Procedure ADD( FIELD : TFIELD)
Procedure ADD( ITEM : TMENUITEM)
Procedure ADD( POPUP : TPOPUPMENU)
Procedure AddCharacters( xCharacters : TCharSet)
Procedure AddDriver( const Name : string; List : TStrings)
Procedure AddImages( Value : TCustomImageList)
Procedure AddIndex( const Name, Fields : string; Options : TIndexOptions; const DescFields : string)
Procedure AddLambdaTransitionTo( oState : TniRegularExpressionState)
Procedure AddLoader( Loader : TBitmapLoader)
Procedure ADDPARAM( VALUE : TPARAM)
Procedure AddPassword( const Password : string)
Procedure AddStandardAlias( const Name, Path, DefaultDriver : string)
Procedure AddState( oState : TniRegularExpressionState)
Procedure AddStrings( Strings : TStrings);
procedure AddStrings(Strings: TStrings);
Procedure AddStrings1( Strings : TWideStrings);
Procedure AddStringTerm( var sString : string; const sTerm : string; const sSeparator : string)
Procedure AddToRecentDocs( const Filename : string)
Procedure AddTransitionTo( oState : TniRegularExpressionState; xCharacters : TCharset)
Procedure AllFunctionsList1Click( Sender : TObject)
procedure AllObjectsList1Click(Sender: TObject);
Procedure Allocate( AAllocateBytes : Integer)
procedure AllResourceList1Click(Sender: TObject);
Procedure AnsiAppend( var dst : AnsiString; const src : AnsiString)
Procedure AnsiAssign( var dst : AnsiString; var src : AnsiString)
Procedure AnsiDelete( var dst : AnsiString; index, count : Integer)
Procedure AnsiFree( var s : AnsiString)
Procedure AnsiFromWide( var dst : AnsiString; const src : WideString)
Procedure AnsiInsert( var dst : AnsiString; const src : AnsiString; index : Integer)
Procedure AnsiSetLength( var dst : AnsiString; len : Integer)
Procedure AnsiString_to_stream( const Value : ansistring; Destin : TStream)
Procedure AntiFreeze;
Procedure APPEND
Procedure Append( const S : WideString)
procedure Append(S: string);
Procedure AppendByte( var VBytes : TIdBytes; AByte : byte)
Procedure AppendBytes( var VBytes : TIdBytes; AAdd : TIdBytes)
Procedure AppendChunk( Val : OleVariant)
Procedure AppendData( const Data : OleVariant; HitEOF : Boolean)
Procedure AppendStr( var Dest : string; S : string)
Procedure AppendString( var VBytes : TIdBytes; const AStr : String; ALength : Integer)
Procedure ApplyRange
procedure Arc(X1, Y1, X2, Y2, X3, Y3, X4, Y4: Integer);
Procedure Arrange( Code : TListArrangement)
procedure Assert(expr : Boolean; const msg: string);
procedure Assert2(expr : Boolean; const msg: string);
Procedure Assign( AList : TCustomBucketList)
Procedure Assign( Other : TObject)
Procedure Assign( Source : TDragObject)
Procedure Assign( Source : TPersistent)
Procedure Assign(Source: TPersistent)
procedure Assign2(mystring, mypath: string);
Procedure AssignCurValues( Source : TDataSet);
Procedure AssignCurValues1( const CurValues : Variant);
Procedure ASSIGNFIELD( FIELD : TFIELD)
Procedure ASSIGNFIELDVALUE( FIELD : TFIELD; const VALUE : VARIANT)
Procedure AssignFile(var F: Text; FileName: string)
procedure AssignFile(var F: TextFile; FileName: string)
procedure AssignFileRead(var mystring, myfilename: string);
procedure AssignFileWrite(mystring, myfilename: string);
Procedure AssignTo( Other : TObject)
Procedure AssignValues( Value : TParameters)
Procedure ASSIGNVALUES( VALUE : TPARAMS)
Procedure AssociateExtension( IconPath, ProgramName, Path, Extension : string)
Procedure Base64_to_stream( const Base64 : ansistring; Destin : TStream)
Procedure Base64ToVar( NatData : Pointer; const SoapData : WideString);
Procedure Base64ToVar1( var V : Variant; const SoapData : WideString);
Procedure BcdAdd( const bcdIn1, bcdIn2 : TBcd; var bcdOut : TBcd)
Procedure BcdDivide( Dividend, Divisor : string; var bcdOut : TBcd);
Procedure BcdDivide1( const Dividend, Divisor : TBcd; var bcdOut : TBcd);
Procedure BcdDivide2( const Dividend : TBcd; const Divisor : Double; var bcdOut : TBcd);
Procedure BcdDivide3( const Dividend : TBcd; const Divisor : string; var bcdOut : TBcd);
Procedure BcdMultiply( const bcdIn1, bcdIn2 : TBcd; var bcdOut : TBcd);
Procedure BcdMultiply1( const bcdIn : TBcd; const DoubleIn : Double; var bcdOut : TBcd);
Procedure BcdMultiply2( const bcdIn : TBcd; const StringIn : string; var bcdOut : TBcd);
Procedure BcdMultiply3( StringIn1, StringIn2 : string; var bcdOut : TBcd);
Procedure BcdSubtract( const bcdIn1, bcdIn2 : TBcd; var bcdOut : TBcd)
Procedure BcdToBytes( Value : TBcd; Bytes : array of byte)
procedure Beep
Procedure BeepOk
Procedure BeepQuestion
Procedure BeepHand
Procedure BeepExclamation
Procedure BeepAsterisk
Procedure BeepInformation
procedure BEGINDRAG(IMMEDIATE:BOOLEAN)
Procedure BeginLayout
Procedure BeginTimer( const Delay, Resolution : Cardinal)
Procedure BeginUpdate
procedure BeginUpdate;
procedure BigScreen1Click(Sender: TObject);
procedure BinToHex(Buffer: PChar; Text: PChar; BufSize: Integer);
Procedure BitsToBooleans( const Bits : Byte; var B : TBooleanArray; AllBits : Boolean);
Procedure BitsToBooleans1( const Bits : Word; var B : TBooleanArray; AllBits : Boolean);
Procedure BitsToBooleans2( const Bits : Integer; var B : TBooleanArray; AllBits : Boolean);
Procedure BitsToBooleans3( const Bits : Int64; var B : TBooleanArray; AllBits : Boolean);
Procedure BoldDays(Days : array of LongWord; var MonthBoldInfo : LongWord)
Procedure BooleansToBits( var Dest : Byte; const B : TBooleanArray);
Procedure BooleansToBits1( var Dest : Word; const B : TBooleanArray);
Procedure BooleansToBits2( var Dest : Integer; const B : TBooleanArray);
Procedure BooleansToBits3( var Dest : Int64; const B : TBooleanArray);
Procedure BreakPointMenuClick( Sender : TObject)
procedure BRINGTOFRONT
procedure BringToFront;
Procedure btnBackClick( Sender : TObject)
Procedure btnBrowseClick( Sender : TObject)
Procedure BtnClick( Index : TNavigateBtn)
Procedure btnLargeIconsClick( Sender : TObject)
Procedure BuildAndSendRequest( AURI : TIdURI)
Procedure BuildCache
Procedure BurnMemory( var Buff, BuffLen : integer)
Procedure BurnMemoryStream( Destructo : TMemoryStream)
Procedure CalculateFirstSet
Procedure Cancel
procedure CancelDrag;
Procedure CancelEdit
procedure CANCELHINT
Procedure CancelRange
Procedure CancelUpdates
Procedure CancelWriteBuffer
Procedure Capture1(ADest:TStream;out VLineCount:Integer;const ADelim:string;const AIsRFCMessage:Bool;
Procedure Capture2( ADest : TStrings; const ADelim : string; const AIsRFCMessage : Boolean);
Procedure Capture3(ADest:TStrings;out VLineCount:Integer;const ADelim:string;const AIsRFCMessage:Bool
procedure CaptureScreenFormat(vname: string; vextension: string);
procedure CaptureScreenPNG(vname: string);
procedure CardinalsToI64(var I: Int64; const LowPart, HighPart: Cardinal);
procedure CASCADE
Procedure CastNativeToSoap(Info:PTypeInfo;var SoapData:WideString;NatData:Pointer;var IsNull: Boolean)
Procedure CastSoapToVariant( SoapInfo : PTypeInfo; const SoapData : WideString; NatData : Pointer);
Procedure cbPathClick( Sender : TObject)
Procedure cbPathKeyDown( Sender : TObject; var Key : Word; Shift : TShiftState)
Procedure cedebugAfterExecute( Sender : TPSScript)
Procedure cedebugBreakpoint( Sender : TObject; const FileName : String; Position, Row, Col : Cardinal)
Procedure cedebugCompile( Sender : TPSScript)
Procedure cedebugExecute( Sender : TPSScript)
Procedure cedebugIdle( Sender : TObject)
Procedure cedebugLineInfo( Sender : TObject; const FileName : String; Position, Row, Col : Cardinal)
Procedure CenterHeight( const pc, pcParent : TControl)
Procedure CenterDlg(AForm: TForm; MForm: TForm);   { Zentriert Forms }
Procedure CenterForm(AForm: TForm; MForm: TForm);   { Zentriert Forms }
Procedure Change
procedure ChangeBiDiModeAlignment(var Alignment: TAlignment);
Procedure Changed
Procedure ChangeDir( const ADirName : string)
Procedure ChangeDirUp
Procedure ChangeEntryTransitions( oNewState : TniRegularExpressionState)
Procedure ChangeLevelBy( Value : TChangeRange)
Procedure ChDir(const s: string)
Procedure Check(Status: Integer)
Procedure CheckCommonControl( CC : Integer)
Procedure CHECKFIELDNAME( const FIELDNAME : String)
Procedure CHECKFIELDNAMES( const FIELDNAMES : String)
Procedure CheckForDisconnect(const ARaiseExceptionIfDisconnected: boolean;const AIgnoreBuffer:bool)
Procedure CheckForGracefulDisconnect( const ARaiseExceptionIfDisconnected : Boolean)
Procedure CheckToken( T : Char)
procedure CheckToken(t:char)
Procedure CheckTokenSymbol( const S : string)
procedure CheckTokenSymbol(s:string)
Procedure CheckToolMenuDropdown( ToolButton : TToolButton)
procedure Chord(X1, Y1, X2, Y2, X3, Y3, X4, Y4: Integer);
Procedure CIED65ToCIED50( var X, Y, Z : Extended)
Procedure CIELABToBGR( const Source, Target : Pointer; const Count : Cardinal);
procedure CipherFile1Click(Sender: TObject);
Procedure Clear;
Procedure Clear1Click( Sender : TObject)
Procedure ClearColor( Color : TColor)
Procedure CLEARITEM( AITEM : TMENUITEM)
Procedure ClearMapping
Procedure ClearSelection( KeepPrimary : Boolean)
Procedure ClearWriteBuffer
Procedure Click
Procedure Close
Procedure Close1Click( Sender : TObject)
Procedure CloseDatabase( Database : TDatabase)
Procedure CloseDataSets
Procedure CloseDialog
Procedure CloseFile(var F: Text);
Procedure Closure
Procedure CMYKToBGR( const Source, Target : Pointer; const BitsPerSample : Byte; Count : Cardinal);
Procedure CMYKToBGR1( const C, M, Y, K, Target : Pointer; const BitsPerSample : Byte; Count : Cardinal);
Procedure CodeCompletionList1Click( Sender : TObject)
Procedure ColEnter
Procedure Collapse
Procedure Collapse( Recurse : Boolean)
Procedure ColorRGBToHLS( clrRGB : TColorRef; var Hue, Luminance, Saturation : Word)
Procedure CommaSeparatedToStringList( AList : TStrings; const Value : string)
Procedure CommitFreeAndNil( var Transaction : TDBXTransaction)
Procedure Compile1Click( Sender : TObject)
procedure ComponentCount1Click(Sender: TObject);
Procedure Compress(azipfolder, azipfile: string)
Procedure DeCompress(azipfolder, azipfile: string)
Procedure XZip(azipfolder, azipfile: string)
Procedure XUnZip(azipfolder, azipfile: string)
Procedure Connect( const ATimeout : Integer)
Procedure Connect( Socket : TSocket)
procedure Console1Click(Sender: TObject);
Procedure Continue
Procedure ContinueCount( var Counter : TJclCounter)
procedure CONTROLDESTROYED(CONTROL:TCONTROL)
Procedure ConvertStreamFromAnsiToUTF8( Src, Dst : TStream; cp : integer)
Procedure ConvertStreamFromUTF8ToAnsi( Src, Dst : TStream; cp : integer)
Procedure ConvertImage(vsource, vdestination: string);
// Ex. ConvertImage(Exepath+'my233_bmp.bmp',Exepath+'mypng111.png')
Procedure ConvertToGray(Cnv: TCanvas);
Procedure Copy( Buffer : TRecordBuffer; Dest : TBytes; Offset : Integer; Length : Integer)
Procedure Copy( Buffer : TValueBuffer; Dest : TBytes; Offset : Integer; Count : Integer);
Procedure Copy1( Source : TBytes; Offset : Integer; Buffer : TValueBuffer; Count : Integer);
Procedure CopyBytesToHostLongWord(const ASource:TIdBytes;const ASourceIndex:Integer;var VDest:LongWord)
Procedure CopyBytesToHostWord( const ASource : TIdBytes; const ASourceIndex : Integer; var VDest : Word)
Procedure CopyFrom( mbCopy : TMyBigInt)
Procedure CopyMemoryStream( Source, Destination : TMemoryStream)
procedure CopyRect(const Dest: TRect; Canvas: TCanvas;const Source: TRect);
Procedure CopyTIdByteArray( const ASource : array of Byte; const ASourceIndex : Integer; var VDest : array
of Byte; const ADestIndex : Integer; const ALength : Integer)
Procedure CopyTIdBytes(const ASrc:TIdBytes;const ASrcIdx:Int;var VDest:TIdBytes;const ADestIdx:Int;const
ALen:Int)
Procedure CopyTIdCardinal( const ASource : Cardinal; var VDest : TIdBytes; const ADestIndex : Integer)
Procedure CopyTIdInt64( const ASource : Int64; var VDest : TIdBytes; const ADestIndex : Integer)
Procedure CopyTIdIPV6Address(const ASource:TIdIPv6Address; var VDest: TIdBytes; const ADestIndex : Integer)
Procedure CopyTIdLongWord( const ASource : LongWord; var VDest : TIdBytes; const ADestIndex : Integer)
Procedure CopyTIdNetworkLongWord( const ASource : LongWord; var VDest : TIdBytes; const ADestIndex:Integer)
Procedure CopyTIdNetworkWord( const ASource : Word; var VDest : TIdBytes; const ADestIndex : Integer)
Procedure CopyTIdString(const ASource:String;var VDest:TIdBytes;const ADestIndex:Integer;ALength: Integer)
Procedure CopyTIdWord( const ASource : Word; var VDest : TIdBytes; const ADestIndex : Integer)
Procedure CopyToClipboard
Procedure CountParts
Procedure CreateDataSet
Procedure CreateEmptyFile( const FileName : string)
Procedure CreateFileFromString( const FileName, Data : string)
Procedure CreateFromDelta( Source : TPacketDataSet)
procedure CREATEHANDLE
Procedure CreateProcAsUser( const UserDomain, UserName, PassWord, CommandLine : string)
Procedure CreateProcAsUserEx(const UserDomain,UserName,Password,CommandLine:string;const Environment:PChar)
Procedure CreateTable
Procedure CreateUDLFile( const FileName, ProviderName, DataSourceName : WideString)
procedure CSyntax1Click(Sender: TObject);
Procedure CurrencyToComp( Value : Currency; var Result : Comp)
Procedure CURSORPOSCHANGED
procedure CutFirstDirectory(var S: String)
Procedure DataBaseError(const Message: string)
Procedure DateTimeToString( var Result : string; Format : string; DateTime : TDateTime);
procedure DateTimeToString(var Result: string; const Format: string; DateTime: TDateTime)
Procedure DateTimeToSystemTime( DateTime : TDateTime; var SystemTime : TSystemTime)
procedure DateTimeToSystemTime(const DateTime: TDateTime; var SystemTime: TSystemTime);
Procedure DBIError(errorCode: Integer)
Procedure DebugOutput( const AText : string)
Procedure DebugRun1Click( Sender : TObject)
procedure Dec;
Procedure DecodeDate( DateTime : TDateTime; var Year, Month, Day : Word)
procedure DecodeDate(const DateTime: TDateTime; var Year, Month, Day: Word);
Procedure DecodeDateDay( const AValue : TDateTime; out AYear, ADayOfYear : Word)
Procedure DecodeDateMonthWeek(const AValue:TDateTime;out AYear, AMonth,AWeekOfMonth,ADayOfWeek : Word)
Procedure DecodeDateTime(const AValue:TDateTime;out AYear,AMonth,ADay,AHour,AMin,ASec,AMillSec:Word)
Procedure DecodeDateWeek( const AValue : TDateTime; out AYear, AWeekOfYear, ADayOfWeek : Word)
Procedure DecodeDayOfWeekInMonth(const AValue:TDateTime;out AYear,AMonth,ANthDayOfWeek,ADayOfWeek:Word)
Procedure DecodeTime( DateTime : TDateTime; var Hour, Min, Sec, MSec : Word)
Procedure DecodeTime(const DateTime: TDateTime; var Hour, Min, Sec, MSec: Word);
Procedure Decompile1Click( Sender : TObject)
Procedure DefaultDrawColumnCell(const Rect : TRect; DataCol:Integer;Column:TColumn;State:TGridDrawState)
Procedure DefaultDrawDataCell( const Rect : TRect; Field : TField; State : TGridDrawState)
Procedure DeferLayout
Procedure defFileread
procedure DEFOCUSCONTROL(CONTROL:TWINCONTROL;REMOVING:BOOLEAN)
Procedure DelayMicroseconds( const MicroSeconds : Integer)
Procedure Delete
Procedure Delete( const AFilename : string)
Procedure Delete( const Index : Integer)
Procedure DELETE( INDEX : INTEGER)
Procedure Delete( Index : LongInt)
Procedure Delete( Node : TTreeNode)
procedure Delete(var s: AnyString; ifrom, icount: Longint);
Procedure DeleteAlias( const Name : string)
Procedure DeleteDriver( const Name : string)
Procedure DeleteIndex( const Name : string)
Procedure DeleteKey( const Section, Ident : String)
Procedure DeleteRecords
Procedure DeleteRecords( AffectRecords : TAffectRecords)
Procedure DeleteString( var pStr : String; const pDelStr : string)
Procedure DeleteTable
procedure DelphiSite1Click(Sender: TObject);
Procedure Deselect
Procedure Deselect( Node : TTreeNode)
procedure DestroyComponents
Procedure DestroyHandle
Procedure Diff( var X : array of Double)
procedure Diff(var X: array of Double);
procedure DISABLEALIGN
Procedure DisableConstraints
Procedure Disconnect
Procedure Disconnect( Socket : TSocket)
Procedure Dispose
procedure Dispose(P: PChar)
Procedure DivMod( Dividend : Integer; Divisor : Word; var Result, Remainder : Word)
Procedure DoKey( Key : TDBCtrlGridKey)
Procedure DomToTree(anXmlNode: IXMLNode; aTreeNode: TTreeNode; aTreeView: TTreeView);
Procedure DomToTreeJ(anXmlNode: TJvXMLNode; aTreeNode: TTreeNode; aTreeView: TTreeView);
Procedure Dormant
Procedure DoubleToBcd1( const AValue : Double; var bcd : TBcd);
Procedure DoubleToBytes( Value : Double; Bytes : array of byte)
Procedure DoubleToComp( Value : Double; var Result : Comp)
Procedure Draw( Canvas : TCanvas; X, Y, Index : Integer; Enabled : Boolean);
procedure Draw(X, Y: Integer; Graphic: TGraphic);
Procedure Draw1(Canvas:TCanvas; X,Y,
Index:Int;ADrawingStyle:TDrawingStyle;AImageType:TImageType;Enabled:Bool);
Procedure DrawArrow( ACanvas : TCanvas; Direction : TScrollDirection; Location : TPoint; Size : Integer)
Procedure DrawCheck( ACanvas : TCanvas; Location : TPoint; Size : Integer; Shadow : Boolean)
Procedure DrawChevron( ACanvas : TCanvas; Direction : TScrollDirection; Location : TPoint; Size : Integer)
Procedure DrawColumnCell( const Rect : TRect; DataCol : Integer; Column : TColumn; State : TGridDrawState)
procedure DrawFocusRect(const Rect: TRect);
Procedure DrawHDIBToTBitmap( HDIB : THandle; Bitmap : TBitmap)
Procedure DRAWMENUITEM( MENUITEM : TMENUITEM; ACANVAS : TCANVAS; ARECT : TRECT; STATE : TOWNERDRAWSTATE)
Procedure DrawOverlay(Canvas:TCanvas;X,Y:Integer;ImageIndex:Integer;Overlay:TOverlay;Enabled: Boolean);
Procedure DrawOverlay1(Canvas:TCanvas; X,Y:Int; ImageIndex:Int; Overlay : TOverlay; ADrawingStyle : TDrawingStyle; AImageType : TImageType; Enabled : Boolean);
procedure drawPlot(vPoints: TPointArray; cFrm: TForm; vcolor: integer);
Procedure DrawPolyLine( const Canvas : TCanvas; var Points : TPointArray; const ClipRect : TRect)
Procedure DropConnections
Procedure DropDown
Procedure DumpDescription( oStrings : TStrings)
Procedure DumpStateTable( oStrings : TStrings)
Procedure EDIT
Procedure EditButtonClick
Procedure EditFont1Click( Sender : TObject)
procedure Ellipse(X1, Y1, X2, Y2: Integer);
Procedure Ellipse1( const Rect : TRect);
Procedure EMMS
Procedure Encode( ADest : TStream)
procedure ENDDRAG(DROP:BOOLEAN)
Procedure EndEdit( Cancel : Boolean)
Procedure EndTimer
Procedure EndUpdate
Procedure EraseSection( const Section : string)
Procedure Error( const Ident : string)
procedure Error(Ident:Integer)
Procedure ErrorFmt( const Ident : string; const Args : array of const)
Procedure ErrorStr( const Message : string)
procedure ErrorStr(Message:String)
Procedure Exchange( Index1, Index2 : Integer)
procedure Exchange(Index1, Index2: Integer);
Procedure Exec( FileName, Parameters, Directory : string)
Procedure ExecProc
Procedure ExecSQL( UpdateKind : TUpdateKind)
Procedure Execute
Procedure Execute( const CommandText : WideString; var Params, OwnerData : OleVariant)
Procedure ExecuteAndWait( FileName : string; Visibility : Integer)
Procedure ExecuteCommand(executeFile, paramstring: string)
Procedure ExecuteShell(executeFile, paramstring: string)
Procedure ShellExecuteAndWait(executeFile, paramstring: string);
Procedure ExitThread(ExitCode: Integer); stdcall;
Procedure ExitProcess(ExitCode: Integer); stdcall;
Procedure Expand( AUserName : String; AResults : TStrings)
Procedure Expand( Recurse : Boolean)
Procedure ExportClipboard1Click( Sender : TObject)
Procedure ExportDataSetToExcel( DataSet : TDataSet; OnExportProgress : TOnExportProgress)
Procedure ExtractContentFields( Strings : TStrings)
Procedure ExtractCookieFields( Strings : TStrings)
Procedure ExtractFields( Separators, WhiteSpace : TSysCharSet; Content : PChar; Strings : TStrings)
Procedure ExtractHeaderFields(Separ, WhiteSpace:TSysChSet;Cont:PChar;Strings:TStrings;Decode:Bool;StripQuots:Bool)
Procedure ExtractHTTPFields(Separators,WhiteSpace: TSysCharSet;Content:PChar;Strings:TStrings;StripQuotes:Bool)
Procedure ExtractQueryFields( Strings : TStrings)
Procedure FastDegToGrad
Procedure FastDegToRad
Procedure FastGradToDeg
Procedure FastGradToRad
Procedure FastRadToDeg
Procedure FastRadToGrad
Procedure FileClose( Handle : Integer)
Procedure FileClose(handle: integer)
procedure FilesFromWildcard(Dir,Mask:string;var Files:TStringList;Subdirs,ShowDirs,Multitasking:Bool)
Procedure FileStructure( AStructure : TIdFTPDataStructure)
Procedure FillByte2(var X: Byte ; count: integer; value: byte)
Procedure FillBytes( var VBytes : TIdBytes; const ACount : Integer; const AValue : Byte)
Procedure FillChar( Buffer : TRecordBuffer; Length : Integer; value : Byte)
Procedure FillChar2(var X: PChar ; count: integer; value: char)
Procedure FillCharS(var p: string; count: integer; value: char); //fix3.8
Procedure FillIPList
procedure FillRect(const Rect: TRect);
Procedure FillTStrings( AStrings : TStrings)
Procedure FilterOnBookmarks( Bookmarks : array of const)
procedure FinalizePackage(Module: HMODULE)
procedure FindClose;
procedure FindClose2(var F: TSearchRec)
Procedure FindMatches( const sString : string; xNotify : TniRegularExpressionMatchFoundEvent);
Procedure FindMatches1(const sString:string;iStart:integer;xNotify:TniRegularExpressionMatchFoundEvent);
Procedure FindNearest( const KeyValues : array of const)
Procedure FinishContext
Procedure FIRST
Procedure FloatToDegMinSec( const X : Float; var Degs, Mins, Secs : Float)
Procedure FloatToDecimal(var Result:TFloatRec;const Value:extend;ValueType:TFloatValue;Precis,Decs:Int);
Procedure FloodFill( X, Y : Integer; Color : TColor; FillStyle : TFillStyle)
Procedure FlushSchemaCache( const TableName : string)
procedure FmtStr(var Result: string; const Format: string; const Args: array of const)
procedure FOCUSCONTROL(CONTROL:TWINCONTROL)
Procedure Form1Close( Sender : TObject; var Action : TCloseAction)
Procedure FormActivate( Sender : TObject)
procedure FormatLn(const format: String; const args: array of const);  //alias
Procedure FormClose( Sender : TObject; var Action : TCloseAction)
Procedure FormCreate( Sender : TObject)
Procedure FormDestroy( Sender : TObject)
Procedure FormKeyPress( Sender : TObject; var Key : Char)
procedure FormOutput1Click(Sender: TObject);
Procedure FormToHtml( Form : TForm; Path : string)
procedure FrameRect(const Rect: TRect);
Procedure Frame3D(Canvas:TCanvas; var Rect:TRect; TopColor,BottomColor:TColor; Width : Integer)
Procedure NotebookHandlesNeeded( Notebook : TNotebook)
Procedure Free( Buffer : TRecordBuffer)
Procedure Free( Buffer : TValueBuffer)
Procedure Free;
Procedure FreeAndNil(var Obj:TObject)
Procedure FreeImage
procedure FreeMem(P: PChar; Size: Integer)
Procedure FreeTreeData( Tree : TUpdateTree)
Procedure Frexp( const X : Extended; var Mantissa : Extended; var Exponent : Integer)
Procedure FullCollapse
Procedure FullExpand
Procedure GenerateDPB(sl: TStrings; var DPB: string; var DPBLength: Short);  //InterBase
Procedure GenerateTPB(sl: TStrings; var TPB: string; var TPBLength: Short);
Procedure OUTPUTXML( SQLOBJECT : TIBSQL; OUTPUTOBJECT : TIBOUTPUTXML)
Procedure Get1( AURL : string; const AResponseContent : TStream);
Procedure Get1( const ASourceFile : string; ADest : TStream; AResume : Boolean);
Procedure Get2(const ASourceFile,ADestFile: string;const ACanOverwrite: boolean; AResume: Boolean);
Procedure GetAliasNames( List : TStrings)
Procedure GetAliasParams( const AliasName : string; List : TStrings)
Procedure GetApplicationsRunning( Strings : TStrings)
Procedure GetCommandTypes( List : TWideStrings)
Procedure GetConfigParams( const Path, Section : string; List : TStrings)
Procedure GetConnectionNames( List : TStrings; Driver : string; DesignMode : Boolean)
Procedure GetConvFamilies( out AFamilies : TConvFamilyArray)
Procedure GetConvTypes( const AFamily : TConvFamily; out ATypes : TConvTypeArray)
Procedure GetDatabaseNames( List : TStrings)
Procedure GetDataPacket( DataSet : TDataSet; var RecsOut : Integer; out Data : OleVariant)
Procedure GetDIBSizes(Bitmap: HBITMAP; var InfoHeaderSize: longWORD; var ImageSize : longWORD)
Procedure GetDir(d: byte; var s: string)
Procedure GetDirList( const Search : string; List : TStrings; Recursive : Boolean)
Procedure GetDriverNames( List : TStrings)
Procedure GetDriverNames( List : TStrings; DesignMode : Boolean)
Procedure GetDriverParams( const DriverName : string; List : TStrings)
Procedure GetEMails1Click( Sender : TObject)
Procedure getEnvironmentInfo;
Function getEnvironmentString: string;
Procedure GetFieldNames( const DatabaseName, TableName : string; List : TStrings)
Procedure GetFieldNames( const TableName : string; List : TStrings)
Procedure GetFieldNames( const TableName : string; List : TStrings);
Procedure GetFieldNames( const TableName : WideString; List : TWideStrings);
Procedure GETFIELDNAMES( LIST : TSTRINGS)
Procedure GetFieldNames1( const TableName : string; List : TStrings);
Procedure GetFieldNames1( const TableName : string; SchemaName : string; List : TStrings);
Procedure GetFieldNames2( const TableName : WideString;SchemaName : WideString; List : TWideStrings);
Procedure GetFieldNames3( const TableName : WideString; List : TWideStrings);
Procedure GetFileAttributeList( const Items : TStrings; const Attr : Integer)
Procedure GetFileAttributeListEx( const Items : TStrings; const Attr : Integer)
Procedure GetFMTBcd( Buffer : TRecordBuffer; var value : TBcd)
Procedure GetFormatSettings
Procedure GetFromDIB( var DIB : TBitmapInfo)
Procedure GetFromHDIB( HDIB : HBitmap)
Procedure GetIcon( Index : Integer; Image : TIcon);
Procedure GetIcon1(Index : Integer; Image:TIcon; ADrawingStyle:TDrawingStyle; AImageType:TImageType);
Procedure GetIndexInfo( IndexName : string)
Procedure GetIndexNames( const TableName, SchemaName : string; List : TStrings);
Procedure GetIndexNames( List : TStrings)
Procedure GetIndexNames1( const TableName : WideString; List : TWideStrings);
Procedure GetIndexNames2( const TableName, SchemaName : WideString; List : TWideStrings);
Procedure GetIndexNames4( const TableName : string; List : TStrings);
Procedure GetInternalResponse
Procedure GETITEMNAMES( LIST : TSTRINGS)
procedure GetMem(P: PChar; Size: Integer)
Procedure GETOLE2ACCELERATORTABLE(var ACCELTABLE:HACCEL;var ACCELCOUNT:INTEGER;GROUPS:array of INTEGER)
procedure GetPackageDescription(ModuleName: PChar): string)
Procedure GetPackageNames( List : TStrings);
Procedure GetPackageNames1( List : TWideStrings);
Procedure GetParamList( List : TList; const ParamNames : WideString)
Procedure GetProcedureNames( List : TStrings);
Procedure GetProcedureNames( List : TWideStrings);
Procedure GetProcedureNames1( const PackageName : string; List : TStrings);
Procedure GetProcedureNames1( List : TStrings);
Procedure GetProcedureNames2( const PackageName, SchemaName : string; List : TStrings);
Procedure GetProcedureNames3( List : TWideStrings);
Procedure GetProcedureNames4( const PackageName : Widestring; List : TWideStrings);
Procedure GetProcedureNames5( const PackageName, SchemaName : WideString; List : TWideStrings);
Procedure GetProcedureParams( ProcedureName : WideString; List : TList);
Procedure GetProcedureParams1( ProcedureName, PackageName : WideString; List : TList);
Procedure GetProcedureParams2( ProcedureName, PackageName, SchemaName : Widestring; List : TList);
Procedure GetProviderNames( Names : TWideStrings);
Procedure GetProviderNames( Proc : TGetStrProc)
Procedure GetProviderNames1( Names : TStrings);
procedure GetQrCode2(Width,Height:Word; Correct_Level: string; const Data:string; apath: string);
procedure GetQrCode3(Width,Height:Word;Correct_Level:string;const Data:string;apath:string);//no autoopen image
Function GetQrCode4(Width,Height:Word;Correct_Level:string; const Data:string;aformat:string):TLinearBitmap;
Procedure GetRGBValue( const Color : TColor; out Red, Green, Blue : Byte)
Procedure GetSchemaNames( List : TStrings);
Procedure GetSchemaNames1( List : TWideStrings);
Procedure GetSessionNames( List : TStrings)
Procedure GetStoredProcNames( const DatabaseName : string; List : TStrings)
Procedure GetStrings( List : TStrings)
Procedure GetSystemTime; stdcall;
Procedure GetTableNames(const DatabaseName,Pattern:string;Extensions,SystemTables:Boolean;List:TStrings)
Procedure GetTableNames( List : TStrings; SystemTables : Boolean)
Procedure GetTableNames( List : TStrings; SystemTables : Boolean);
Procedure GetTableNames( List : TWideStrings; SystemTables : Boolean);
Procedure GetTableNames1( List : TStrings; SchemaName : WideString; SystemTables : Boolean);
Procedure GetTableNames1( List : TStrings; SystemTables : Boolean);
Procedure GetTableNames2( List : TWideStrings; SchemaName : WideString; SystemTables : Boolean);
Procedure GetTransitionsOn( cChar : char; oStateList : TList)
Procedure GetVisibleWindows( List : Tstrings)
Procedure GoBegin
Procedure GotoCurrent( DataSet : TCustomClientDataSet)
Procedure GotoCurrent( Table : TTable)
procedure GotoEnd1Click(Sender: TObject);
Procedure GotoNearest
Procedure GradientFillCanvas(const ACanvas:TCanvas;const AStartCol,AEndCol:TColor;const ARect:TRect;const Direction: TGradientDirection)
Procedure HandleException( E : Exception; var Handled : Boolean)
procedure HANDLEMESSAGE
procedure HandleNeeded;
Procedure Head( AURL : string)
Procedure Help( var AHelpContents : TStringList; ACommand : String)
Procedure HexToBinary( Stream : TStream)
procedure HexToBinary(Stream:TStream)
Procedure HideDragImage
Procedure HideFormCaption( FormHandle : THandle; Hide : Boolean)
Procedure HideTraybar
Procedure HideWindowForSeconds(secs: integer);   {//3 seconds}
Procedure HideWindowForSeconds2(secs: integer; apphandle, aself: TForm);   {//3 seconds}
Procedure HookOSExceptions
Procedure HookSignal( RtlSigNum : Integer)
Procedure HSLToRGB( const H, S, L : Single; out R, G, B : Single);
Procedure HTMLSyntax1Click( Sender : TObject)
Procedure IFPS3ClassesPlugin1CompImport( Sender : TObject; x : TPSPascalCompiler)
Procedure IFPS3ClassesPlugin1ExecImport( Sender : TObject; Exec : TPSExec; x : TPSRuntimeClassImporter)
Procedure ImportfromClipboard1Click( Sender : TObject)
Procedure ImportfromClipboard2Click( Sender : TObject)
Procedure IncAMonth( var Year, Month, Day : Word; NumberOfMonths : Integer)
procedure Incb(var x: byte);
Procedure Include1Click( Sender : TObject)
Procedure IncludeOFF;  //preprocessing
Procedure IncludeON;
procedure Info1Click(Sender: TObject);
Procedure InitAltRecBuffers( CheckModified : Boolean)
Procedure InitContext( Request : TWebRequest; Response : TWebResponse)
Procedure InitContext(WebModuleList:TAbstractWebModuleList;Request:TWebRequest;Response:TWebResponse)
Procedure InitData( ASource : TDataSet)
Procedure InitDelta( ADelta : TPacketDataSet);
Procedure InitDelta1( const ADelta : OleVariant);
Procedure InitErrorPacket( E : EUpdateError; Response : TResolverResponse)
Procedure Initialize
procedure InitializePackage(Module: HMODULE)
Procedure INITIATEACTION
Procedure initHexArray(var hexn: THexArray);  //THexArray', 'array[0..15] of char;'
Procedure InitKeyFields( Tree : TUpdateTree; ADelta : TPacketDataSet)
Procedure InitModule( AModule : TComponent)
Procedure InitStdConvs
Procedure InitTreeData( Tree : TUpdateTree)
Procedure INSERT
Procedure Insert( Index : Integer; AClass : TClass)
Procedure Insert( Index : Integer; AComponent : TComponent)
Procedure Insert( Index : Integer; AObject : TObject)
Procedure Insert( Index : Integer; const S : WideString)
Procedure Insert( Index : Integer; Image, Mask : TBitmap)
Procedure Insert(Index: Integer; const S: string);
procedure Insert(Index: Integer; S: string);
procedure Insert(s: AnyString; var s2: AnyString; iPos: Longint);
procedure InsertComponent(AComponent:TComponent)
procedure InsertControl(AControl: TControl);
Procedure InsertIcon( Index : Integer; Image : TIcon)
Procedure InsertMasked( Index : Integer; Image : TBitmap; MaskColor : TColor)
Procedure InsertObject( Index : Integer; const S : WideString; AObject : TObject)
procedure InsertObject(Index:Integer;S:String;AObject:TObject)
Procedure Int16ToBytes( Value : SmallInt; Bytes : array of byte)
Procedure Int32ToBytes( Value : Integer; Bytes : array of byte)
Procedure Int64ToBytes( Value : Int64; Bytes : array of byte)
```

```
procedure I64ToCardinals(I: Int64; var LowPart, HighPart: Cardinal);
Procedure InternalBeforeResolve( Tree : TUpdateTree)
Procedure InvalidateModuleCache
Procedure InvalidateTitles
Procedure InvalidDateDayError( const AYear, ADayOfYear : Word)
Procedure InvalidDateMonthWeekError( const AYear, AMonth, AWeekOfMonth, ADayOfWeek : Word)
Procedure InvalidDateTimeError(const AYear,AMth,ADay,AHour,AMin,ASec,AMilSec:Word;const
ABaseDate:TDateTime)
Procedure InvalidDateWeekError(const AYear, AWeekOfYear, ADayOfWeek : Word)
Procedure InvalidDayOfWeekInMonthError( const AYear, AMonth, ANthDayOfWeek, ADayOfWeek : Word)
procedure JavaSyntax1Click(Sender: TObject);
Procedure JclLocalesInfoList( const Strings : TStrings; InfoType : Integer)
Procedure KillDataChannel
Procedure Largefont1Click( Sender : TObject)
Procedure LAST
Procedure LaunchCpl( FileName : string)
Procedure Launch( const AFile : string)
Procedure LaunchFile( const AFile : string)
Procedure LetFileList(FileList: TStringlist; apath: string);
Procedure lineToNumber( xmemo : String; met : boolean)
Procedure ListViewCustomDrawItem(Sender:TCustListView;Item:TListItem;State:TCustDrawState;var
DefaultDraw:Bool)
Procedure ListViewCustomDrawSubItem( Sender : TCustomListView; Item : TListItem; SubItem : Integer; State
: TCustomDrawState; var DefaultDraw : Boolean)
Procedure ListViewData( Sender : TObject; Item : TListItem)
Procedure ListViewDataFind(Sender:TObject; Find : TItemFind; const FindString : String; const FindPosition
: TPoint; FindData:Pointer; StartIndex:Integer;Direction:TSearchDirection;Wrap:Boolean;var Index: Integer)
Procedure ListViewDataHint( Sender : TObject; StartIndex, EndIndex : Integer)
Procedure ListViewDblClick( Sender : TObject)
Procedure ListViewKeyDown( Sender : TObject; var Key : Word; Shift : TShiftState)
Procedure ListDLLExports(const FileName: string; List: TStrings);
Procedure Load( const WSDLFileName : WideString; Stream : TMemoryStream)
procedure LoadBytecode1Click(Sender: TObject);
procedure LoadFilefromResource(const FileName: string; ms: TMemoryStream);
Procedure LoadFromClipboardFormat( AFormat : Word; AData : THandle; APalette : HP)
Procedure LoadFromClipboardFormat( AFormat : Word; AData : THandle; APalette : HPALETTE)
Procedure LoadFromFile( AFileName : string)
Procedure LoadFromFile( const AFileName : string; const AHeadersOnly : Boolean)
Procedure LoadFromFile( const FileName : string)
Procedure LOADFROMFILE( const FILENAME : String; BLOBTYPE : TBLOBTYPE)
Procedure LoadFromFile( const FileName : string; DataType : TDataType)
Procedure LoadFromFile( const FileName : WideString)
Procedure LoadFromFile( const FileName, FileType : string; Bitmap : TLinearBitmap)
Procedure LoadFromFile(const AFileName: string)
procedure LoadFromFile(FileName: string);
procedure LoadFromFile(FileName:String)
Procedure LoadFromResourceID( Instance : THandle; ResID : Integer)
Procedure LoadFromResourceName( Instance : THandle; const ResName : String)
Procedure LoadFromStream( AStream : TStream; const AHeadersOnly : Boolean)
Procedure LoadFromStream( const Stream : TStream)
Procedure LoadFromStream( S : TStream)
Procedure LoadFromStream( Stream : TSeekableStream; const Ext : string; Bitmap : TLinarBitmap)
Procedure LoadFromStream( Stream : TSeekableStream; const Ext : string; Bitmap : TLinearBitmap)
Procedure LoadFromStream( Stream : TStream)
Procedure LOADFROMSTREAM( STREAM : TSTREAM; BLOBTYPE : TBLOBTYPE)
Procedure LoadFromStream( Stream : TStream; DataType : TDataType)
procedure LoadFromStream(Stream: TStream);
Procedure LoadFromStream1( Stream : TSeekableStream; const FormatExt : string);
Procedure LoadFromStream2( Stream : TStream; const FormatExt : string);
Procedure LoadFromStrings( AStrings : TStrings; const MimeSeparator : Char)
Procedure LoadLastFile1Click( Sender : TObject)
{ LoadIcoToImage loads two icons from resource named NameRes,
  into two image lists ALarge and ASmall}
Procedure LoadIcoToImage(ALarge, ASmall: ImgList.TCustomImageList; const NameRes: string);
Procedure LoadMemo
Procedure LoadParamsFromIniFile( FFileName : WideString)
Procedure Lock
Procedure Login
Procedure MakeAlphaChannelFromAlphaPalette( Source : TLinearBitmap)
Procedure MakeAlphaChannelFromColorKey( Source : TLinearBitmap; ColorKey : TColor)
Procedure MakeCaseInsensitive
Procedure MakeDeterministic( var bChanged : boolean)
Procedure MakeGrayPal( var Palette, ColorCount : Integer)
// type TVolumeLevel = 0..127;  , savaFilePath as C:\MyFile.wav
Procedure MakeSound(Frequency{Hz}, Duration{mSec}: Int; Volume: TVolumeLevel; savefilePath: string);
Procedure MakeComplexSound(N:integer;freqlist:TStrings;Duration{mSec}:Int;pinknoise:bool;Volume:Byte);
Procedure SetComplexSoundElements(freqedt,Phaseedt,AmpEdt,WaveGrp:integer);
Procedure SetRectComplexFormatStr( const S : string)
Procedure SetPolarComplexFormatStr( const S : string)
Procedure AddComplexSoundObjectToList(newf,newp,newa,news:integer; freqlist: TStrings);
Procedure MakeVisible
Procedure MakeVisible( PartialOK : Boolean)
Procedure Manual1Click( Sender : TObject)
Procedure MarkReachable
Procedure maXbox;  //shows the exe version data in a win box
Procedure MeanAndStdDev( const Data : array of Double; var Mean, StdDev : Extended)
Procedure Memo1Change( Sender : TObject)
Procedure Memo1ReplaceText(Sender:TObject;const ASearch,AReplace:String;Line,Column:Int;var
Action:TSynReplaceAction)
```

```
16593: Procedure Memo1SpecialLineColors(Sender:TObject;Line:Integer; var Special:Boolean;var FG,BG: TColor)
16594: Procedure Memo1StatusChange( Sender : TObject; Changes : TSynStatusChanges)
16595: procedure Memory1Click(Sender: TObject);
16596: Procedure MERGE( MENU : TMAINMENU)
16597: Procedure MergeChangeLog
16598: procedure MINIMIZE
16599: Procedure MinimizeMaxbox;
16600: Procedure MkDir(const s: string)
16601: Procedure mnuPrintFont1Click( Sender : TObject)
16602: procedure ModalStarted
16603: Procedure Modified
16604: Procedure ModifyAlias( Name : string; List : TStrings)
16605: Procedure ModifyDriver( Name : string; List : TStrings)
16606: Procedure MomentSkewKurtosis(const Data:array of Double;var M1,M2,M3,M4, Skew,Kurtosis:Extended)
16607: Procedure MouseToCell( X, Y : Integer; var ACol, ARow : Longint)
16608: Procedure Move( CurIndex, NewIndex : Integer)
16609: procedure Move(CurIndex, NewIndex: Integer);
16610: Procedure Move2(const Source: TByteArray; var Dest: TByteArray; Count: Integer)
16611: Procedure MoveChars(const ASource: String; ASourceStart:int;var ADest: String; ADestStart,ALen:integer)
16612: Procedure moveCube( o : TMyLabel)
16613: Procedure MoveTo( Destination : LongInt; AttachMode : TAttachMode)
16614: Procedure MoveTo(X, Y: Integer);
16615: procedure MoveWindowOrg(DC: HDC; DX, DY: Integer);
16616: Procedure MovePoint(var x,y:Extended; const angle:Extended);
16617: Procedure Multiply( Multiplier1, Multiplier2 : TMyBigInt);
16618: Procedure Multiply1( Multiplier1 : TMyBigInt; Multiplier2 : Integer);
16619: Procedure MsgAbout(Handle:Int;const Msg,Caption:string;const IcoName:string ='MAINICON';Flags:DWORD=MB_OK);
16620: Procedure mxButton(x,y,width,height,top,left,ahandle: integer);
16621: Procedure New( Width, Height : Integer; PixFormat : TPixelFormat)
16622: procedure New(P: PChar)
16623: procedure New1Click(Sender: TObject);
16624: procedure NewInstance1Click(Sender: TObject);
16625: Procedure NEXT
16626: Procedure NextMonth
16627: Procedure Noop
16628: Procedure NormalizePath( var APath : string)
16629: procedure ObjectBinaryToText(Input, Output: TStream)
16630: procedure ObjectBinaryToText1(Input, Output: TStream;var OriginalFormat: TStreamOriginalFormat)
16631: procedure ObjectResourceToText(Input, Output: TStream)
16632: procedure ObjectResourceToText1(Input, Output: TStream;var OriginalFormat: TStreamOriginalFormat)
16633: procedure ObjectTextToBinary(Input, Output: TStream)
16634: procedure ObjectTextToBinary1(Input, Output: TStream;var OriginalFormat: TStreamOriginalFormat)
16635: procedure ObjectTextToResource(Input, Output: TStream)
16636: procedure ObjectTextToResource1(Input, Output: TStream;var OriginalFormat: TStreamOriginalFormat)
16637: Procedure Open( const Name, Address, Service : string; Port : Word; Block : Boolean)
16638: Procedure Open( const UserID : WideString; const Password : WideString);
16639: Procedure Open;
16640: Procedure open1Click( Sender : TObject)
16641: Procedure OpenCdDrive
16642: Procedure OpenCloseCdDrive( OpenMode : Boolean; Drive : Char)
16643: Procedure OpenCurrent
16644: Procedure OpenFile(vfilenamepath: string)
16645: Procedure OpenDirectory1Click( Sender : TObject)
16646: Procedure OpenIndexFile( const IndexName : string)
16647: Procedure OpenSchema(const Schema:TSchemaInf;const Restricts:OleVar;const
       SchemaID:OleVariant;DataSet:TADODataSet)
16648: Procedure OpenWriteBuffer( const AThreshhold : Integer)
16649: Procedure OptimizeMem
16650: Procedure Options1( AURL : string);
16651: Procedure OutputDebugString(lpOutputString : PChar)
16652: Procedure PackBuffer
16653: Procedure Paint
16654: Procedure PaintToCanvas( Canvas : TCanvas; const Dest : TRect; HalftoneStretch : Boolean)
16655: Procedure PaintToTBitmap( Target : TBitmap)
16656: Procedure PaletteChanged
16657: Procedure ParentBiDiModeChanged
16658: Procedure PARENTBIDIMODECHANGED( ACONTROL : TOBJECT)
16659: Procedure PasteFromClipboard;
16660: Procedure PasteImage( Source : TLinearBitmap; X, Y : Integer)
16661: Procedure PathExtractElements( const Source : string; var Drive, Path, FileName, Ext : string)
16662: Procedure PerformEraseBackground(Control: TControl; DC: HDC);
16663: Procedure PError( Text : string)
16664: procedure Pie(X1, Y1, X2, Y2, X3, Y3, X4, Y4: Integer);
16665: procedure Pie(X1:Integer;Y1:Integer;X2:Integer;Y2:Integer;X3:Int;Y3:Integer;X4:Integer;Y4:Integer);
16666: Procedure Play( FromFrame, ToFrame : Word; Count : Integer)
16667: procedure playmp3(mpath: string);
16668: Procedure PlayMP31Click( Sender : TObject)
16669: Procedure PointCopy( var Dest : TPoint; const Source : TPoint)
16670: Procedure PointMove( var P : TPoint; const DeltaX, DeltaY : Integer)
16671: procedure PolyBezier(const Points: array of TPoint);
16672: procedure PolyBezierTo(const Points: array of TPoint);
16673: procedure Polygon(const Points: array of TPoint);
16674: procedure Polyline(const Points: array of TPoint);
16675: Procedure Pop
16676: Procedure POPULATEOLE2MENU(SHAREDMENU: HMENU;GROUPS: array of INT; var WIDTHS : array of LONGINT)
16677: Procedure PopulationVarianceAndMean( const X : TDynFloatArray; var Variance, Mean : Float)
16678: Procedure POPUP( X, Y : INTEGER)
16679: Procedure PopupURL(URL : WideString);
16680: Procedure POST
```

```
Procedure Post4( AURL : string; const ASource : TStrings; const AResponseContent : TStream);
Procedure Post5( AURL : string; const ASource, AResponseContent : TStream);
Procedure Post6( AURL : string; const ASource : TIdMultiPartFormDataStream; AResponseContent : TStream);
Procedure PostUser( const Email, FirstName, LastName : WideString)
Procedure PostKeyEx32(key: Word; const shift: TShiftState; specialkey: Boolean);
procedure Pred(X: int64);
Procedure Prepare
Procedure PrepareStatement
Procedure PreProcessXML( AList : TStrings)
Procedure PreventDestruction
Procedure Print( const Caption : string)
procedure PrintBitmap(aGraphic: TGraphic; Title: string);
procedure printf(const format: String; const args: array of const);
Procedure PrintList(Value: TStringList);
Procedure PrintImage(aValue:TBitmap;Style:TBitmapStyle);//TBitmapStyle=(bsNormal,bsCentered,bsStretched)
Procedure Printout1Click( Sender : TObject)
Procedure ProcessHeaders
Procedure PROCESSMENUCHAR( var MESSAGE : TWMMENUCHAR)
Procedure ProcessMessage( AMsg : TIdMessage; AHeaderOnly : Boolean);
Procedure ProcessMessage1( AMsg : TIdMessage; const AStream : TStream; AHeaderOnly : Boolean);
Procedure ProcessMessage2( AMsg : TIdMessage; const AFilename : string; AHeaderOnly : Boolean);
Procedure ProcessMessagesOFF;  //application.processmessages
Procedure ProcessMessagesON;
Procedure ProcessPath(const EditText:string;var Drive:Char; var DirPart:string; var FilePart : string)
Procedure ProcessPath1(const EditText:string;var Drive:Char; var DirPart:string; var FilePart:string);
Procedure Proclist Size is: 3797 /1415
Procedure procMessClick( Sender : TObject)
Procedure PSScriptCompile( Sender : TPSScript)
Procedure PSScriptExecute( Sender : TPSScript)
Procedure PSScriptLine( Sender : TObject)
Procedure Push( ABoundary : string)
procedure PushItem(AItem: Pointer)
Procedure Put2( AURL : string; const ASource, AResponseContent : TStream);
Procedure Put2( const ASourceFile : string; const ADestFile : string; const AAppend : boolean);
procedure PutLinuxLines(const Value: string)
Procedure Quit
Procedure RaiseConversionError( const AText : string);
Procedure RaiseConversionError1( const AText : string; const AArgs : array of const);
Procedure RaiseConversionRegError( AFamily : TConvFamily; const ADescription : string)
procedure RaiseException(Ex: TIFException; Param: String);
Procedure RaiseExceptionForLastCmdResult;
procedure RaiseLastException;
procedure RaiseException2;
Procedure RaiseLastOSError
Procedure RaiseLastWin32;
procedure RaiseLastWin32Error)
Procedure RaiseListError( const ATemplate : string; const AData : array of const)
Procedure RandomFillStream( Stream : TMemoryStream)
procedure randomize;
Procedure Rasterize( Rasterizer : TRasterizer; Dst : TBitmap32; DstRect : TRect)
Procedure RCS
Procedure Read( Socket : TSocket)
Procedure ReadBlobData
procedure ReadBuffer(Buffer:String;Count:LongInt)
procedure ReadOnly1Click(Sender: TObject);
Procedure ReadSection( const Section : string; Strings : TStrings)
Procedure ReadSections( Strings : TStrings)
Procedure ReadSections( Strings : TStrings);
Procedure ReadSections1( const Section : string; Strings : TStrings);
Procedure ReadSectionValues( const Section : string; Strings : TStrings)
Procedure ReadStream( AStream : TStream; AByteCount:LongInt; const AReadUntilDisconnect : boolean)
Procedure ReadStrings( ADest : TStrings; AReadLinesCount : Integer)
Procedure ReadVersion2(aFileName: STRING; aVersion : TStrings);
Function ReadVersion(aFileName: STRING; aVersion : TStrings): boolean;
Procedure Realign;
procedure Rectangle(X1, Y1, X2, Y2: Integer);
Procedure Rectangle1( const Rect : TRect);
Procedure RectCopy( var Dest : TRect; const Source : TRect)
Procedure RectFitToScreen( var R : TRect)
Procedure RectGrow( var R : TRect; const Delta : Integer)
Procedure RectGrowX( var R : TRect; const Delta : Integer)
Procedure RectGrowY( var R : TRect; const Delta : Integer)
Procedure RectMove( var R : TRect; const DeltaX, DeltaY : Integer)
Procedure RectMoveTo( var R : TRect; const X, Y : Integer)
Procedure RectNormalize( var R : TRect)
//  TFileCallbackProcedure = procedure(filename:string);
Procedure RecurseDirectory(Dir: String;IncludeSubs: boolean;callback: TFileCallbackProcedure);
Procedure RecurseDirectory2(Dir: String; IncludeSubs : boolean);
Procedure RedirectTransition(oOldState:TniRegularExpressionState; oNewState : TniRegularExpressionState)
Procedure Refresh;
Procedure RefreshData( Options : TFetchOptions)
Procedure REFRESHLOOKUPLIST
Procedure RegisterAuthenticationMethod( MethodName : String; AuthClass : TIdAuthenticationClass)
Procedure RegisterChanges( Value : TChangeLink)
Procedure RegisterConversionFormat( const AExtension : string; AConversionClass : TConversionClass)
Procedure RegisterFileFormat( const AExtension, ADescription : string; AGraphicClass : TGraphicClass)
Procedure RegisterFileFormat(Extension,AppID:string;Description:string;Executable:string;IconIndex:Int)
Procedure ReInitialize( ADelay : Cardinal)
procedure RELEASE
```

```
Procedure Remove( const AByteCount : integer)
Procedure REMOVE( FIELD : TFIELD)
Procedure REMOVE( ITEM : TMENUITEM)
Procedure REMOVE( POPUP : TPOPUPMENU)
Procedure RemoveAllPasswords
procedure RemoveComponent(AComponent:TComponent)
Procedure RemoveDir( const ADirName : string)
Procedure RemoveLambdaTransitions( var bChanged : boolean)
Procedure REMOVEPARAM( VALUE : TPARAM)
Procedure RemoveTransitionTo( oState : TniRegularExpressionState; xCharacters : TCharset);
Procedure RemoveTransitionTo1( oState : TniRegularExpressionState);
Procedure Rename( const ASourceFile, ADestFile : string)
Procedure Rename( const FileName : string; Reload : Boolean)
Procedure RenameTable( const NewTableName : string)
Procedure Replace( Index : Integer; Image, Mask : TBitmap)
Procedure Replace1Click( Sender : TObject)
Procedure ReplaceDate( var DateTime : TDateTime; NewDate : TDateTime)
procedure ReplaceDate(var DateTime: TDateTime; const NewDate: TDateTime))
Procedure ReplaceIcon( Index : Integer; Image : TIcon)
Procedure ReplaceMasked( Index : Integer; NewImage : TBitmap; MaskColor : TColor)
Procedure ReplaceTime( var DateTime : TDateTime; NewTime : TDateTime)
procedure ReplaceTime(var DateTime: TDateTime; const NewTime: TDateTime);
Procedure Requery( Options : TExecuteOptions)
Procedure Reset
Procedure Reset1Click( Sender : TObject)
Procedure ResizeCanvas( XSiz, YSiz, XPos, YPos : Integer; Color : TColor)
procedure ResourceExplore1Click(Sender: TObject);
Procedure RestoreContents
Procedure RestoreDefaults
Procedure RestoreOtherInstance( MainFormClassName, MainFormCaption : string)
Procedure RetrieveHeaders
Procedure RevertRecord
Procedure RGBAToBGRA( const Source, Target : Pointer; const BitsPerSample : Byte; Count : Cardinal)
Procedure RGBToBGR( const Source, Target : Pointer; const BitsPerSample : Byte; Count : Cardinal);
Procedure RGBToBGR1( const R, G, B, Target : Pointer; const BitsPerSample : Byte; Count : Cardinal);
Procedure RGBToHSL( const R, G, B : Single; out H, S, L : Single);
Procedure RGBToHSL1( const RGB : TColor32; out H, S, L : Single);
Procedure RGBToHSV( r, g, b : Integer; var h, s, v : Integer)
Procedure RleCompress2( Stream : TStream)
Procedure RleDecompress2( Stream : TStream)
Procedure RmDir(const S: string)
Procedure Rollback
Procedure Rollback( TransDesc : TTransactionDesc)
Procedure RollbackFreeAndNil( var Transaction : TDBXTransaction)
Procedure RollbackIncompleteFreeAndNil( var Transaction : TDBXTransaction)
Procedure RollbackTrans
procedure RoundRect(X1, Y1, X2, Y2, X3, Y3: Integer);
Procedure RoundToAllocGranularity64( var Value : Int64; Up : Boolean)
Procedure RoundToAllocGranularityPtr( var Value : Pointer; Up : Boolean)
Procedure RunDll32Internal( Wnd : HWnd; const DLLName, FuncName, CmdLine : string; CmdShow : Integer)
Procedure S_AddMessageToStrings( AMessages : TStrings; AMsg : string)
Procedure S_EBox( const AText : string)
Procedure S_GetEncryptionKeys(DateTime1,DateTime2:TDateTime;var StartKey:int;var MultKey:integer;var AddKey:int
Procedure S_IBox( const AText : string)
Procedure S_ReplaceChar( var cStr : string; cOldChr, cNewChr : char)
Procedure S_ReplaceStringInFile( AFileName : string; ASearchString, AReplaceString : string)
Procedure S_TokenInit( cBuffer : PChar; const cDelimiters : string)
Procedure SampleVarianceAndMean
( const X : TDynFloatArray; var Variance, Mean : Float)
Procedure Save2Click( Sender : TObject)
Procedure Saveas3Click( Sender : TObject)
Procedure Savebefore1Click( Sender : TObject)
Procedure SaveBytesToFile(const Data: TBytes; const FileName: string);
procedure SaveCanvas2(vCanvas: TCanvas; FileName: string);
Procedure SaveConfigFile
Procedure SaveOutput1Click( Sender : TObject)
procedure SaveScreenshotClick(Sender: TObject);
Procedure SaveLn(pathname, content: string);   //Saveln(exepath+'mysavelntest.txt', memo2.text);
Procedure SaveToClipboardFormat( var AFormat : Word; var AData : THandle; var APalette : HPALETTE)
Procedure SaveToClipboardFormat( var Format : Word; var Data : THandle; var APalette : HPALETTE)
Procedure SaveToFile( AFileName : string)
Procedure SAVETOFILE( const FILENAME : String)
Procedure SaveToFile( const FileName : WideString)
Procedure SaveToFile( const FileName : WideString; Format : TPersistFormat)
Procedure SaveToFile( const FileName, FileType : string; Bitmap : TLinearBitmap)
procedure SaveToFile(FileName: string);
procedure SaveToFile(FileName:String)
Procedure SaveToStream( AStream : TStream; const AHeadersOnly : Boolean)
Procedure SaveToStream( OutStream : TSeekableStream; const Ext : string; Bitmap : TLinarBitmap)
Procedure SaveToStream( S : TStream)
Procedure SaveToStream( Stream : TSeekableStream; const Ext : string; Bitmap : TLinearBitmap)
Procedure SaveToStream( Stream : TStream)
Procedure SaveToStream( Stream : TStream; Format : TDataPacketFormat)
procedure SaveToStream(Stream: TStream);
procedure SaveToStream(Stream:TStream)
Procedure SaveToStream1( Stream : TSeekableStream; const FormatExt : string);
Procedure SaveToStream2( Stream : TStream; const FormatExt : string);
Procedure SaveToStrings( AStrings : TStrings; const MimeSeparator : Char)
```

```
procedure Say(const sText: string)
Procedure SBytecode1Click( Sender : TObject)
Procedure ScaleImage( const SourceBitmap, ResizedBitmap : TBitmap; const ScaleAmount : Double)
procedure ScriptExplorer1Click(Sender: TObject);
Procedure Scroll( Distance : Integer)
Procedure Scroll( DX, DY : Integer)
procedure ScrollBy(DeltaX, DeltaY: Integer);
procedure SCROLLINVIEW(ACONTROL:TCONTROL)
Procedure ScrollTabs( Delta : Integer)
Procedure Search1Click( Sender : TObject)
procedure SearchAndOpenDoc(vfilenamepath: string)
procedure SearchAndOpenFile(vfilenamepath: string)
procedure SearchAndReplace(aStrList: TStrings; aSearchStr, aNewStr: string)
procedure SearchAndCopy(aStrList: TStrings; aSearchStr, aNewStr: string; offset: integer);
Procedure SearchNext1Click( Sender : TObject)
Procedure Select( Node : TTreeNode; ShiftState : TShiftState);
Procedure Select1( const Nodes : array of TTreeNode);
Procedure Select2( Nodes : TList);
Procedure SelectNext( Direction : Boolean)
Procedure SelectNextPage( GoForward : Boolean; CheckTabVisible : Boolean)
Procedure SelfTestPEM  //unit uPSI_cPEM
//config forst in const MAILINIFILE = 'maildef.ini';
//ex.:  SendEmail('max@kleiner.ch','max@kleiner.com','this test7','maxbox the SSL fox','
Procedure SendEmail(mFrom,  mTo,  mSubject,  mBody,  mAttachment: variant);
Procedure SendMail(mFrom,  mTo,  mSubject,  mBody,  mAttachment: variant);
Procedure SendMsg( AMsg : TIdMessage; const AHeadersOnly : Boolean)
Procedure SendMsg(AMsg: TIdMessage; const AHeadersOnly: Boolean = False)
Procedure SendResponse
Procedure SendStream( AStream : TStream)
Procedure Set8087CW( NewCW : Word)
Procedure SetAll( One, Two, Three, Four : Byte)
Procedure SetAltRecBuffers( Old, New, Cur : PChar)
Procedure SetAppDispatcher( const ADispatcher : TComponent)
procedure SetArrayLength;
procedure SetArrayLength2String(arr: T2StringArray; asize1, asize2: integer);  //2 dimension
procedure SetArrayLength2Integer(arr: T2IntegerArray; asize1, asize2: integer);
Procedure SetAsHandle( Format : Word; Value : THandle)
procedure SetBounds(ALeft, ATop, AWidth, AHeight: Integer)
procedure SetCaptureControl(Control: TControl);
Procedure SetColumnAttributes
Procedure SetCookieField(Values:TStrings;const ADomain,APath:string;AExpires:TDateTime;ASecure:Boolean)
Procedure SetCustomHeader( const Name, Value : string)
Procedure SetExprParams(const Text:Widestring;Opts:TFilterOpts;ParserOpts:TParserOpts;const
FieldName:Widestring)
Procedure SetFMTBcd( Buffer : TRecordBuffer; value : TBcd)
Procedure SetFocus
procedure SetFocus; virtual;
Procedure SetInitialState
Procedure SetKey
procedure SetLastError(ErrorCode: Integer)
procedure SetLength;
Procedure SetLineBreakStyle( var T : Text; Style : TTextLineBreakStyle)
Procedure SETOLE2MENUHANDLE( HANDLE : HMENU)
Procedure SetParams( ADataset : TDataset; UpdateKind : TUpdateKind);
procedure SETPARAMS(APOSITION,AMIN,AMAX:INTEGER)
Procedure SetParams1( UpdateKind : TUpdateKind);
Procedure SetPassword( const Password : string)
Procedure SetPointer( Ptr : Pointer; Size : Longint)
Procedure SetPrimalityTest( const Method : TPrimalityTestMethod)
Procedure SetPrinter( ADevice, ADriver, APort : PChar; ADeviceMode : THandle)
Procedure SetProvider( Provider : TComponent)
Procedure SetProxy( const Proxy : string)
Procedure SetPSResult( var PSResult : TPSResult; Value : TObject)
Procedure SetRange( const StartValues, EndValues : array of const)
Procedure SetRangeEnd
Procedure SetRate( const aPercent, aYear : integer)
procedure SetRate(const aPercent, aYear: integer)
Procedure Set_ReportMemoryLeaksOnShutdown(abo: boolean)
Procedure SetSafeCallExceptionMsg( Msg : String)
procedure SETSELTEXTBUF(BUFFER:PCHAR)
Procedure SetSize( AWidth, AHeight : Integer)
procedure SetSize(NewSize:LongInt)
procedure SetString(var s: string; buffer: PChar; len: Integer)
Procedure SetStrings( List : TStrings)
Procedure SetText( Text : PwideChar)
procedure SetText(Text: PChar);
Procedure SetTextBuf( Buffer : PChar)
procedure SETTEXTBUF(BUFFER:PCHAR)
Procedure SetTick( Value : Integer)
Procedure SetTimeout( ATimeOut : Integer)
Procedure SetTraceEvent( Event : TDBXTraceEvent)
Procedure SetUserName( const UserName : string)
Procedure SetWallpaper( Path : string);
procedure ShellStyle1Click(Sender: TObject);
Procedure SHORTCUTTOKEY( SHORTCUT : TSHORTCUT; var KEY : WORD; var SHIFT : TSHIFTSTATE)
Procedure ShowFileProperties( const FileName : string)
Procedure ShowInclude1Click( Sender : TObject)
Procedure ShowInterfaces1Click( Sender : TObject)
Procedure ShowLastException1Click( Sender : TObject)
Procedure ShowMessage( const Msg : string)
Procedure ShowMessageBig(const aText : string);
Procedure ShowMessageBig2(const aText : string; aautosize: boolean);
Procedure ShowMessageBig3(const aText : string; fsize: byte; aautosize: boolean);
Procedure MsgBig(const aText : string);        //alias
procedure showmessage(mytext: string);
Procedure ShowMessageFmt( const Msg : string; Params : array of const)
procedure ShowMessageFmt(const Msg: string; Params: array of const))
Procedure ShowMessagePos( const Msg : string; X, Y : Integer)
procedure ShowMessagePos(const Msg: string; X: Integer; Y: Integer))
Procedure ShowSearchDialog( const Directory : string)
Procedure ShowSpecChars1Click( Sender : TObject)
Procedure ShowBitmap(bmap: TBitmap);  //draw in a form!
Procedure ShredFile( const FileName : string; Times : Integer)
procedure Shuffle(vQ: TStringList);
Procedure ShuffleList( var List : array of Integer; Count : Integer)
Procedure SimulateKeystroke( Key : byte; Shift : TShiftState)
Procedure SinCos( const Theta : Extended; var Sin, Cos : Extended)
Procedure SinCosE( X : Extended; out Sin, Cos : Extended)
Procedure Site( const ACommand : string)
Procedure SkipEOL
Procedure Sleep( ATime : cardinal)
Procedure Sleep( milliseconds : Cardinal)
Function SleepEx( dwMilliseconds : DWORD; bAlertable : BOOL) : DWORD
Procedure Slinenumbers1Click( Sender : TObject)
Procedure Sort
Procedure SortColorArray(ColorArray:TColorArray;L,R:Int;SortType:TColorArraySortType;Reverse:Bool)
procedure Speak(const sText: string)   //async like voice
procedure Speak2(const sText: string)  //sync
procedure Split(Str: string;  SubStr: string; List: TStrings);
Procedure SplitNameValue( const Line : string; var Name, Value : string)
Procedure SplitColumns( const AData : String; AStrings : TStrings; const ADelim : String)
Procedure SplitColumnsNoTrim( const AData : String; AStrings : TStrings; const ADelim : String)
Procedure SplitLines( AData : PChar; ADataSize : Integer; AStrings : TStrings)
Procedure SplitString( const AStr, AToken : String; var VLeft, VRight : String)
procedure SQLSyntax1Click(Sender: TObject);
Procedure SRand( Seed : RNG_IntType)
Procedure Start
Procedure StartCount( var Counter : TJclCounter; const Compensate : Boolean)
procedure StartFileFinder3(spath,aext,searchstr: string; arecursiv: boolean; reslist: TStringlist);
Procedure StartTransaction( TransDesc : TTransactionDesc)
Procedure Status( var AStatusList : TStringList)
Procedure StatusBar1DblClick( Sender : TObject)
Procedure StepInto1Click( Sender : TObject)
Procedure StepIt
Procedure StepOut1Click( Sender : TObject)
Procedure Stop
procedure stopmp3;
procedure StartWeb(aurl: string);
Procedure Str(aint: integer; astr: string); //of system
Procedure StrDispose( Str : PChar)
procedure StrDispose(Str: PChar)
Procedure StrReplace(var Str: String; Old, New: String);
Procedure StretchDIBits( DC : THandle; const Dest : TRect; HalftoneStretch : Boolean)
procedure StretchDraw(const Rect: TRect; Graphic: TGraphic);
Procedure StringToBytes( Value : String; Bytes : array of byte)
procedure StrSet(c : Char; I : Integer; var s : String);
Procedure StrSplitP(const Delimiter: Char; Input: string; const Strings: TStrings);
Procedure StructureMount( APath : String)
procedure STYLECHANGED(SENDER:TOBJECT)
Procedure Subselect( Node : TTreeNode; Validate : Boolean)
procedure Succ(X: int64);
Procedure SumsAndSquares( const Data : array of Double; var Sum, SumOfSquares : Extended)
procedure SwapChar(var X,Y: char);  //swapX follows
Procedure SwapFloats( var X, Y : Float)
procedure SwapGrid(grd: TStringGrid);
Procedure SwapOrd( var I, J : Byte);
Procedure SwapOrd( var X, Y : Integer)
Procedure SwapOrd1( var I, J : Shortint);
Procedure SwapOrd2( var I, J : Smallint);
Procedure SwapOrd3( var I, J : Word);
Procedure SwapOrd4( var I, J : Integer);
Procedure SwapOrd5( var I, J : Cardinal);
Procedure SwapOrd6( var I, J : Int64);
Procedure SymetricCompareFiles(const plaintext, replaintext: string)
Procedure Synchronize1( Method : TMethod);
procedure SyntaxCheck1Click(Sender: TObject);
Procedure SysFreeString(const S: WideString); stdcall;
Procedure TakeOver( Other : TLinearBitmap)
Procedure Talkln(const sText: string)  //async voice
procedure tbtn6resClick(Sender: TObject);
Procedure tbtnUseCaseClick( Sender : TObject)
procedure TerminalStyle1Click(Sender: TObject);
Procedure Terminate
Procedure texSyntax1Click( Sender : TObject)
procedure TextOut(X, Y: Integer; Text: string);
Procedure TextRect( Rect : TRect; X, Y : Integer; const Text : string);
procedure TextRect(Rect: TRect; X: Integer; Y: Integer; const Text: string);
```

```
Procedure TextRect1( var Rect : TRect; var Text : string; TextFormat : TTextFormat);
Procedure TextStart
procedure TILE
Procedure TimeStampToBytes( Value : TBcd; Bytes : array of byte)
Procedure TitleClick( Column : TColumn)
Procedure ToDo
procedure toolbtnTutorialClick(Sender: TObject);
Procedure Trace1( AURL : string; const AResponseContent : TStream);
Procedure TransferMode( ATransferMode : TIdFTPTransferMode)
Procedure Truncate
procedure Tutorial101Click(Sender: TObject);
procedure Tutorial10Statistics1Click(Sender: TObject);
procedure Tutorial11Forms1Click(Sender: TObject);
procedure Tutorial12SQL1Click(Sender: TObject);
Procedure tutorial1Click( Sender : TObject)
Procedure tutorial21Click( Sender : TObject)
Procedure tutorial31Click( Sender : TObject)
Procedure tutorial4Click( Sender : TObject)
Procedure Tutorial5Click( Sender : TObject)
procedure Tutorial6Click(Sender: TObject);
procedure Tutorial91Click(Sender: TObject);
Procedure UnhookSignal( RtlSigNum : Integer; OnlyIfHooked : Boolean)
procedure UniqueString(var str: AnsiString)
procedure UnloadLoadPackage(Module: HMODULE)
Procedure Unlock
Procedure UNMERGE( MENU : TMAINMENU)
Procedure UnRegisterChanges( Value : TChangeLink)
Procedure UnregisterConversionFamily( const AFamily : TConvFamily)
Procedure UnregisterConversionType( const AType : TConvType)
Procedure UnRegisterProvider( Prov : TCustomProvider)
Procedure UPDATE
Procedure UpdateBatch( AffectRecords : TAffectRecords)
Procedure UPDATECURSORPOS
Procedure UpdateFile
Procedure UpdateItems( FirstIndex, LastIndex : Integer)
Procedure UpdateResponse( AResponse : TWebResponse)
Procedure UpdateScrollBar
Procedure UpdateView1Click( Sender : TObject)
procedure Val(const s: string; var n, z: Integer)
procedure VarArraySet(c : Variant; I : Integer; var s : Variant);
Procedure VarFMTBcdCreate( var ADest : Variant; const ABcd : TBcd);
Procedure VariantAdd( const src : Variant; var dst : Variant)
Procedure VariantAnd( const src : Variant; var dst : Variant)
Procedure VariantArrayRedim( var V : Variant; High : Integer)
Procedure VariantCast( const src : Variant; var dst : Variant; vt : Integer)
Procedure VariantClear( var V : Variant)
Procedure VariantCpy( const src : Variant; var dst : Variant)
Procedure VariantDiv( const src : Variant; var dst : Variant)
Procedure VariantMod( const src : Variant; var dst : Variant)
Procedure VariantMul( const src : Variant; var dst : Variant)
Procedure VariantOr( const src : Variant; var dst : Variant)
Procedure VariantPutElement( var V : Variant; const data : Variant; i1 : integer);
Procedure VariantPutElement1( var V : Variant; const data : Variant; i1, i2 : integer);
Procedure VariantPutElement2( var V : Variant; const data : Variant; i1, i2, i3 : integer);
Procedure VariantPutElement3( var V : Variant; const data : Variant; i1, i2, i3, i4 : integer);
Procedure VariantPutElement4( var V : Variant; const data : Variant; i1, i2, i3, i4, i5 : integer);
Procedure VariantShl( const src : Variant; var dst : Variant)
Procedure VariantShr( const src : Variant; var dst : Variant)
Procedure VariantSub( const src : Variant; var dst : Variant)
Procedure VariantXor( const src : Variant; var dst : Variant)
Procedure VarCastError;
Procedure VarCastError1( const ASourceType, ADestType : TVarType);
Procedure VarInvalidOp
Procedure VarInvalidNullOp
Procedure VarOverflowError( const ASourceType, ADestType : TVarType)
Procedure VarRangeCheckError( const ASourceType, ADestType : TVarType)
Procedure VarArrayCreateError
Procedure VarResultCheck( AResult : HRESULT);
Procedure VarResultCheck1( AResult : HRESULT; ASourceType, ADestType : TVarType);
Procedure HandleConversionException( const ASourceType, ADestType : TVarType)
Function VarTypeAsText( const AType : TVarType) : string
procedure Voice(const sText: string) //async
procedure Voice2(const sText: string) //sync
Procedure WaitMiliSeconds( AMSec : word)
Procedure WideAppend( var dst : WideString; const src : WideString)
Procedure WideAssign( var dst : WideString; var src : WideString)
Procedure WideDelete( var dst : WideString; index, count : Integer)
Procedure WideFree( var s : WideString)
Procedure WideFromAnsi( var dst : WideString; const src : AnsiString)
Procedure WideFromPChar( var dst : WideString; src : PChar)
Procedure WideInsert( var dst : WideString; const src : WideString; index : Integer)
Procedure WideSetLength( var dst : WideString; len : Integer)
Procedure WideString2Stream( aWideString : WideString; oStream : TStream)
Procedure WideStringToBytes( Value : WideString; Bytes : array of byte)
Procedure WinColorToOpenGLColor( const Color : TColor; out Red, Green, Blue : Float)
Procedure WinInet_HttpGet(const Url: string; Stream:TStream);
Procedure HttpGet(const Url: string; Stream:TStream);
Procedure WordToTwoBytes( AWord : Word; ByteArray : TIdBytes; Index : integer)
Procedure WordWrap1Click( Sender : TObject)
Procedure Write( const AOut : string)
Procedure Write( Socket : TSocket)
procedure Write(S: string);
Procedure WriteBinaryStream( const Section, Name : string; Value : TStream)
Procedure WriteBool( const Section, Ident : string; Value : Boolean)
Procedure WriteBuffer( const ABuffer, AByteCount : Longint; const AWriteNow : Boolean)
procedure WriteBuffer(Buffer:String;Count:LongInt)
Procedure WriteCardinal( AValue : Cardinal; const AConvert : Boolean)
Procedure WriteChar( AValue : Char)
Procedure WriteDate( const Section, Name : string; Value : TDateTime)
Procedure WriteDateTime( const Section, Name : string; Value : TDateTime)
Procedure WriteFloat( const Section, Name : string; Value : Double)
Procedure WriteHeader( AHeader : TStrings)
Procedure WriteInteger( AValue : Integer; const AConvert : Boolean)
Procedure WriteInteger( const Section, Ident : string; Value : Longint)
Procedure WriteLn( const AOut : string)
procedure Writeln(s: string);
Procedure WriteLog( const FileName, LogLine : string)
Procedure WriteRFCReply( AReply : TIdRFCReply)
Procedure WriteRFCStrings( AStrings : TStrings)
Procedure WriteSmallInt( AValue : SmallInt; const AConvert : Boolean)
Procedure WriteStream(AStream:TStream;const AAll:Bool;const AWriteByteCount:Bool;const ASize:Int)
Procedure WriteString( const Section, Ident, Value : String)
Procedure WriteStrings( AValue : TStrings; const AWriteLinesCount : Boolean)
Procedure WriteTime( const Section, Name : string; Value : TDateTime)
Procedure WriteObjectResourceHeader( ObjStream, Output : TStream)
Procedure Write16bitResourceHeader( const AName : TBytes; DataSize : Integer; Output : TStream)
Procedure Write32bitResourceHeader( const AName : TBytes; DataSize : Integer; Output : TStream)
procedure WStrSet(c : AnyString; I : Integer; var s : AnyString);
procedure XMLSyntax1Click(Sender: TObject);
Procedure XOR_Streams2( Dest, Srce : TMemoryStream)
Procedure XOR_Streams3( Dest, SrceA, SrceB : TMemoryStream)
Procedure ZeroFillStream( Stream : TMemoryStream)
procedure XMLSyntax1Click(Sender: TObject);
Procedure ZeroMemory( Ptr : Pointer; Length : Longint)
procedure(Control: TWinControl; Index: Integer; Rect: TRect; State: Byte)
procedure(Control: TWinControl; Index: Integer; var Height: Integer)
procedure(Sender, Source: TObject; X, Y: Integer; State: TDragState; var Accept: Boolean)
procedure(Sender, Source: TObject;X, Y: Integer)
procedure(Sender, Target: TObject; X, Y: Integer)
procedure(Sender: TObject; ASection, AWidth: Integer)
procedure(Sender: TObject; ScrollCode: TScrollCode;var ScrollPos: Integer)
procedure(Sender: TObject; Shift: TShiftState; X, Y: Integer);
procedure(Sender: TObject; var Action: TCloseAction)
procedure(Sender: TObject; var CanClose: Boolean)
procedure(Sender: TObject; var Key: Char);
ProcedureName ProcedureNames ProcedureParametersCursor @

*************Now Constructors constructor *************
 Size is: 1248 1115 996 628 550 544 501 459 (381)
 Attach( VersionInfoData : Pointer; Size : Integer)
 constructor Create( ABuckets : TBucketListSizes)
 Create( ACallBackWnd : HWND)
 Create( AClient : TCustomTaskDialog)
 Create( AClient : TIdTelnet)
 Create( ACollection : TCollection)
 Create( ACollection : TFavoriteLinkItems)
 Create( ACollection : TTaskDialogButtons)
 Create( AConnection : TIdCustomHTTP)
 Create( ACreateSuspended : Boolean)
 Create( ADataSet : TCustomSQLDataSet)
 CREATE( ADATASET : TDATASET)
 Create( Aggregates : TAggregates; ADataSet : TCustomClientDataSet);
 Create( AGrid : TCustomDBGrid)
 Create( AGrid : TStringGrid; AIndex : Longint)
 Create( AHTTP : TIdCustomHTTP)
 Create( AListItems : TListItems)
 Create( AOnBytesRemoved : TIdBufferBytesRemoved)
 Create( AOnBytesRemoved : TIdBufferBytesRemoved)
 Create( AOwner : TCommonCalendar)
 Create( AOwner : TComponent)
 CREATE( AOWNER : TCOMPONENT)
 Create( AOwner : TCustomListView)
 Create( AOwner : TCustomOutline)
 Create( AOwner : TCustomRichEdit)
 Create( AOwner : TCustomRichEdit; AttributeType : TAttributeType)
 Create( AOwner : TCustomTreeView)
 Create( AOwner : TIdUserManager)
 Create( AOwner : TListItems)
Create(AOwner:TObject;Handl:hDBICur;CBTyp:CBType;CBBuf:Ptr;CBBufSiz:Int;CallbckEvent:TBDECallbackEvent;Chain:Bool)
 CREATE( AOWNER : TPERSISTENT)
 Create( AOwner : TPersistent)
 Create( AOwner : TTable)
 Create( AOwner : TTreeNodes)
 Create( AOwner : TWinControl; const ClassName : string)
 Create( AParent : TIdCustomHTTP)
 Create( AParent : TUpdateTree; AResolver : TCustomResolver)
 Create( AProvider : TBaseProvider)
 Create( AProvider : TCustomProvider);
 Create( AProvider : TDataSetProvider)
 Create( ASocket : TCustomWinSocket; TimeOut : Longint)
 Create( ASocket : TSocket)
 Create( AStrings : TWideStrings)
 Create( AToolBar : TToolBar)
 Create( ATreeNodes : TTreeNodes)
 Create( Autofill : boolean)
 Create( AWebPageInfo : TAbstractWebPageInfo)
 Create( AWebRequest : TWebRequest)
 Create( Collection : TCollection)
 Create( Collection : TIdMessageParts; ABody : TStrings)
 Create( Collection : TIdMessageParts; const AFileName : TFileName)
 Create( Column : TColumn)
 Create( const AConvFamily : TConvFamily; const ADescription : string)
 Create( const AConvFamily : TConvFamily; const ADescription : string; const AFactor : Double)
 Create( const AConvFamily : TConvFamily; const ADescription : string; const AToCommonProc, AFromCommonProc : TConversionProc)
 Create( const AInitialState : Boolean; const AManualReset : Boolean)
 Create( const ATabSet : TTabSet)
 Create( const Compensate : Boolean)
 Create( const FileMap : TJclCustomFileMapping; Access, Size : Cardinal; ViewOffset : Int64)
 Create( const FileName : string)
 Create( const FileName : string;FileMode:Cardinal; const Name:string; Protect:Cardinal; const MaximumSize : Int64; const SecAttr : PSecurityAttributes);
 Create( const FileName : string; FileMode : WordfmShareDenyWrite)
 Create( const MaskValue : string)
 Create(const Name:string; Protect:Cardinal;const MaximumSize:Int64;const SecAttr:PSecurityAttributes)
 Create( const Prefix : string)
 Create( const sRegularExpression : string; xFlags : TniRegularExpressionMatchFlags)
 Create( const sRule : string; xFlags : TniRegularExpressionMatchFlags)
 Create( const sRule : string; xFlags : TniRegularExpressionMatchFlags)
 Create( CoolBar : TCoolBar)
 Create( CreateSuspended : Boolean; ASocket : TServerClientWinSocket)
 Create( CreateSuspended : Boolean; ASocket : TServerWinSocket)
 Create( DataSet :TDataSet; const Text:Widestring;Options:TFilterOptions;ParserOptions:TParserOptions;const FieldName : Widestring; DepFields : TBits; FieldMap : TFieldMap)
 Create( DBCtrlGrid : TDBCtrlGrid)
 Create( DSTableProducer : TDSTableProducer)
 Create( DSTableProducer : TDSTableProducer; ColumnClass : THTMLTableColumnClass)
 Create( ErrorCode : DBIResult)
 Create( Field : TBlobField; Mode : TBlobStreamMode)
 Create( Grid : TCustomDBGrid; ColumnClass : TColumnClass)
 Create( HeaderControl : TCustomHeaderControl)
 Create( HTTPRequest : TWebRequest)
 Create( iStart : integer; sText : string)
 Create( iValue : Integer)
 Create( Kind : TMmTimerKind; Notification : TMmNotificationKind)
 Create( MciErrNo : MCIERROR; const Msg : string)
 Create(MemoryStream:TCustomMemStream;FreeStream:Boolean;const AIndexOption:TJclMappedTextReaderIndex);
 Create( Message : string; ErrorCode : DBResult)
 Create( Msg : string)
 Create( NativeError, Context : string; ErrCode, PrevError : Integer; E : Exception)
 Create( NativeError, Context : string; ErrorCode, PreviousError : DBResult)
 Create( oExpression : TniRegularExpression; eType : TniRegularExpressionStateType)
 Create( oOwner : TniRegularExpression; xFlags : TniRegularExpressionMatchFlags)
 Create(oSource:TniRegularExpressState;oDestination:TniRegularExprState;xCharacts:TCharSet;bLambda:bool)
 Create( Owner : EDBEngineError; ErrorCode : DBIResult; NativeError : Longint; Message : PChar)
 Create( Owner : TCustomComboBoxEx)
 CREATE( OWNER : TINDEXDEFS; const NAME, FIELDS: String; OPTIONS: TINDEXOPTIONS)
 Create( Owner : TPersistent)
 Create( Params : TStrings)
 Create( Size : Cardinal)
 Create( Socket : TSocket; ServerWinSocket : TServerWinSocket)
 Create( StatusBar : TCustomStatusBar)
 Create( WebDispatcher : TCustomWebDispatcher; ItemClass : TCollectionItemClass)
 Create( WebResponse : TWebResponse; ItemClass : TCollectionItemClass)
 Create(AHandle:Integer)
 Create(AOwner: TComponent); virtual;
 Create(const AURI : string)
 Create(FileName:String;Mode:Word)
 Create(Instance:THandle;ResName:String;ResType:PChar)
 Create(Stream : TStream)
 Create1( ADataset : TDataset);
 Create1(const FileHandle:THandle;const Name:string;Protect:Cardinal;const MaximumSize:Int64;const SecAttr:PSecurityAttributes);
 Create1( const FileName : string; const AIndexOption : TJclMappedTextReaderIndex);
 Create2( Other : TObject);
 CreateAt(FileMap: TJclCustomFileMapping; Access,Size:Cardinal;ViewOffset:Int64;Address: Pointer)
 CreateError(const anErrCode: Integer;const asReplyMessage: string; const asErrorMessage: string)
 CreateFmt( MciErrNo : MCIERROR; const Msg : string; const Args : array of const)
 CreateFromId(Instance:THandle;ResId:Integer;ResType:PChar)
 CreateLinked( DBCtrlGrid : TDBCtrlGrid)
 CREATENEW(AOWNER:TCOMPONENT; Dummy: Integer)
 CreateRes( Ident : Integer);
 CreateRes( MciErrNo : MCIERROR; Ident : Integer)
 CreateRes( ResStringRec : PResStringRec);
 CreateResHelp( Ident : Integer; AHelpContext : Integer);
```

```
 CreateResHelp( ResStringRec : PResStringRec; AHelpContext : Integer);
 CreateShadow( AOwner : TComponent; ControlSide : TControlSide)
 CreateSize( AWidth, AHeight : Integer)
 Open( const Name : string; const InheritHandle : Boolean; const DesiredAccess : Cardinal)

 ----------------------------------------------------------------------------
unit uPSI_MathMax;
 ----------------------------------------------------------------------------
 CONSTS
  Bernstein: Float = 0.2801694990238691330364364912307;  // Bernstein constant
  Cbrt2: Float     = 1.2599210498948731647672106072782;  // CubeRoot(2)
  Cbrt3: Float     = 1.4422495703074083823216383107801;  // CubeRoot(3)
  Cbrt10: Float    = 2.1544346900318837217592935665194;  // CubeRoot(10)
  Cbrt100: Float   = 4.6415888336127788924100763509194;  // CubeRoot(100)
  CbrtPi: Float    = 1.4645918875615232630201425272638;  // CubeRoot(PI)
  Catalan: Float   = 0.9159655941772190150546035149324;  // Catalan constant
  PiJ:   Float       = 3.1415926535897932384626433832795;  // PI
  PI:  Extended =  3.1415926535897932384626433832795);
  PiOn2: Float     = 1.5707963267948966192313216916398;  // PI / 2
  PiOn3: Float     = 1.0471975511965977461542144610932;  // PI / 3
  PiOn4: Float     = 0.78539816339744830961566084581988; // PI / 4
  Sqrt2: Float     = 1.4142135623730950488016887242097;  // Sqrt(2)
  Sqrt3: Float     = 1.7320508075688772935274463415059;  // Sqrt(3)
  Sqrt5: Float     = 2.2360679774997896964091736687313;  // Sqrt(5)
  Sqrt10: Float    = 3.1622776601683793319988935444327;  // Sqrt(10)
  SqrtPi: Float    = 1.7724538509055160272981674833411;  // Sqrt(PI)
  Sqrt2Pi: Float   = 2.506628274631000502415765284811;   // Sqrt(2 * PI)
  TwoPi: Float     = 6.283185307179586476925286766559;   // 2 * PI
  ThreePi: Float   = 9.4247779607693797153879301498385;  // 3 * PI
  Ln2: Float       = 0.69314718055994530941723212145818; // Ln(2)
  Ln10: Float      = 2.3025850929940456840179914546844;  // Ln(10)
  LnPi: Float      = 1.1447298858494001741434273513531;  // Ln(PI)
  Log2J: Float     = 0.30102999566398119521373889472449; // Log10(2)
  Log3: Float      = 0.47712125471966243729502790325512; // Log10(3)
  LogPi: Float     = 0.4971498726941338543512682882909;  // Log10(PI)
  LogE: Float      = 0.43429448190325182765112891891661; // Log10(E)
  E: Float         = 2.7182818284590452353602874713527;  // Natural constant
  hLn2Pi: Float    = 0.91893853320467274178032973640562; // Ln(2*PI)/2
  inv2Pi: Float    = 0.159154943091895335768883763372514362034459645740456; // 0.5/Pi
  TwoToPower63: Float = 9223372036854775808.0;           // 2^63
  GoldenMean: Float   = 1.618033988749894848204586834365638;  // GoldenMean
  EulerMascheroni: Float = 0.5772156649015328606065120900824;  // Euler GAMMA
  RadCor : Double = 57.29577951308232;    {number of degrees in a radian}
  StDelta         : Extended = 0.00001;  {delta for difference equations}
  StEpsilon       : Extended = 0.00001;  {epsilon for difference equations}
  StMaxIterations : Integer  = 100;      {max attempts for convergence}

procedure SIRegister_StdConvs(CL: TPSPascalCompiler);
begin
  MetersPerInch = 0.0254; // [1]
  MetersPerFoot = MetersPerInch * 12;
  MetersPerYard = MetersPerFoot * 3;
  MetersPerMile = MetersPerFoot * 5280;
  MetersPerNauticalMiles = 1852;
  MetersPerAstronomicalUnit = 1.49598E11; // [4]
  MetersPerLightSecond = 2.99792458E8; // [5]
  MetersPerLightYear = MetersPerLightSecond * 31556925.9747; // [7]
  MetersPerParsec = MetersPerAstronomicalUnit * 206264.806247096; // 60 * 60 * (180 / Pi)
  MetersPerCubit = 0.4572; // [6][7]
  MetersPerFathom = MetersPerFoot * 6;
  MetersPerFurlong = MetersPerYard * 220;
  MetersPerHand = MetersPerInch * 4;
  MetersPerPace = MetersPerInch * 30;
  MetersPerRod = MetersPerFoot * 16.5;
  MetersPerChain = MetersPerRod * 4;
  MetersPerLink = MetersPerChain / 100;
  MetersPerPoint = MetersPerInch * 0.013837; // [7]
  MetersPerPica = MetersPerPoint * 12;

  SquareMetersPerSquareInch = MetersPerInch * MetersPerInch;
  SquareMetersPerSquareFoot = MetersPerFoot * MetersPerFoot;
  SquareMetersPerSquareYard = MetersPerYard * MetersPerYard;
  SquareMetersPerSquareMile = MetersPerMile * MetersPerMile;
  SquareMetersPerAcre = SquareMetersPerSquareYard * 4840;
  SquareMetersPerSquareRod = MetersPerRod * MetersPerRod;

  CubicMetersPerCubicInch = MetersPerInch * MetersPerInch * MetersPerInch;
  CubicMetersPerCubicFoot = MetersPerFoot * MetersPerFoot * MetersPerFoot;
  CubicMetersPerCubicYard = MetersPerYard * MetersPerYard * MetersPerYard;
  CubicMetersPerCubicMile = MetersPerMile * MetersPerMile * MetersPerMile;
  CubicMetersPerAcreFoot = SquareMetersPerAcre * MetersPerFoot;
  CubicMetersPerAcreInch = SquareMetersPerAcre * MetersPerInch;
  CubicMetersPerCord = CubicMetersPerCubicFoot * 128;
  CubicMetersPerCordFoot = CubicMetersPerCubicFoot * 16;

  CubicMetersPerUSFluidGallon = CubicMetersPerCubicInch * 231; // [2][3][7]
  CubicMetersPerUSFluidQuart = CubicMetersPerUSFluidGallon / 4;
  CubicMetersPerUSFluidPint = CubicMetersPerUSFluidQuart / 2;
  CubicMetersPerUSFluidCup = CubicMetersPerUSFluidPint / 2;
  CubicMetersPerUSFluidGill = CubicMetersPerUSFluidCup / 2;
  CubicMetersPerUSFluidOunce = CubicMetersPerUSFluidCup / 8;
  CubicMetersPerUSFluidTablespoon = CubicMetersPerUSFluidOunce / 2;
  CubicMetersPerUSFluidTeaspoon = CubicMetersPerUSFluidOunce / 6;

  CubicMetersPerUSDryGallon = CubicMetersPerCubicInch * 268.8025; // [7]
  CubicMetersPerUSDryQuart = CubicMetersPerUSDryGallon / 4;
  CubicMetersPerUSDryPint = CubicMetersPerUSDryQuart / 2;
  CubicMetersPerUSDryPeck = CubicMetersPerUSDryGallon * 2;
  CubicMetersPerUSDryBucket = CubicMetersPerUSDryPeck * 2;
  CubicMetersPerUSDryBushel = CubicMetersPerUSDryBucket * 2;

  CubicMetersPerUKGallon = 0.00454609; // [2][7]
  CubicMetersPerUKPottle = CubicMetersPerUKGallon / 2;
  CubicMetersPerUKQuart = CubicMetersPerUKPottle / 2;
  CubicMetersPerUKPint = CubicMetersPerUKQuart / 2;
  CubicMetersPerUKGill = CubicMetersPerUKPint / 4;
  CubicMetersPerUKOunce = CubicMetersPerUKPint / 20;
  CubicMetersPerUKPeck = CubicMetersPerUKGallon * 2;
  CubicMetersPerUKBucket = CubicMetersPerUKPeck * 2;
  CubicMetersPerUKBushel = CubicMetersPerUKBucket * 2;

  GramsPerPound = 453.59237; // [1][7]
  GramsPerDrams = GramsPerPound / 256;
  GramsPerGrains = GramsPerPound / 7000;
  GramsPerTons = GramsPerPound * 2000;
  GramsPerLongTons = GramsPerPound * 2240;
  GramsPerOunces = GramsPerPound / 16;
  GramsPerStones = GramsPerPound * 14;

  MaxAngle 9223372036854775808.0;
  MaxTanH 5678.2617031470719747459655389854);
  MaxFactorial( 1754);
  MaxFloatingPoint(1.189731495357231765085759326628E+4932);
  MinFloatingPoint',(3.3621031431120935062626778173218E-4932);
  MaxTanH( 354.89135644669199842162284618659);
  MaxFactorial'LongInt'( 170);
  MaxFloatingPointD(1.797693134862315907729305190789E+308);
  MinFloatingPointD(2.2250738585072013830902327173324E-308);
  MaxTanH( 44.361419555836499802702855773323);
  MaxFactorial'LongInt'( 33);
  MaxFloatingPointS( 3.4028236692093846346337460743177E+38);
  MinFloatingPointS( 1.1754943508222875079687365372222E-38);
  PiExt( 3.1415926535897932384626433832795);
  RatioDegToRad( PiExt / 180.0);
  RatioGradToRad( PiExt / 200.0);
  RatioDegToGrad( 200.0 / 180.0);
  RatioGradToDeg( 180.0 / 200.0);
  Crc16PolynomCCITT'LongWord $1021);
 Crc16PolynomIBM'LongWord $8005);
 Crc16Bits'LongInt'( 16);
 Crc16Bytes'LongInt'( 2);
 Crc16HighBit'LongWord $8000);
 NotCrc16HighBit','LongWord $7FFF);
  Crc32PolynomIEEE','LongWord $04C11DB7);
 Crc32PolynomCastagnoli','LongWord $1EDC6F41);
 Crc32Koopman','LongWord $741B8CD7);
 Crc32Bits','LongInt'( 32);
 Crc32Bytes','LongInt'( 4);
 Crc32HighBit','LongWord $80000000);
 NotCrc32HighBit','LongWord $7FFFFFFF);

  MinByte       = Low(Byte);
  MaxByte       = High(Byte);
  MinWord       = Low(Word);
  MaxWord       = High(Word);
  MinShortInt   = Low(ShortInt);
  MaxShortInt   = High(ShortInt);
  MinSmallInt   = Low(SmallInt);
  MaxSmallInt   = High(SmallInt);
  MinLongWord   = LongWord(Low(LongWord));
  MaxLongWord   = LongWord(High(LongWord));
  MinLongInt    = LongInt(Low(LongInt));
  MaxLongInt    = LongInt(High(LongInt));
  MinInt64      = Int64(Low(Int64));
  MaxInt64      = Int64(High(Int64));
  MinInteger    = Integer(Low(Integer));
  MaxInteger    = Integer(High(Integer));
  MinCardinal   = Cardinal(Low(Cardinal));
  MaxCardinal   = Cardinal(High(Cardinal));
  MinNativeUInt = NativeUInt(Low(NativeUInt));
  MaxNativeUInt = NativeUInt(High(NativeUInt));
  MinNativeInt  = NativeInt(Low(NativeInt));
  MaxNativeInt  = NativeInt(High(NativeInt));
 Function CosH( const Z : Float) : Float;
 Function SinH( const Z : Float) : Float;
 Function TanH( const Z : Float) : Float;


```

```
//***********from DMath.Dll Lib of types.inc in source\dmath_dll
InvLn2     = 1.44269504088896340736;  { 1/Ln(2) }
InvLn10    = 0.43429448190325182765;  { 1/Ln(10) }
TwoPi      = 6.28318530717958647693;  { 2*Pi }
PiDiv2     = 1.57079632679489661923;  { Pi/2 }
SqrtPi     = 1.77245385090551602730;  { Sqrt(Pi) }
Sqrt2Pi    = 2.50662827463100050242;  { Sqrt(2*Pi) }
InvSqrt2Pi = 0.39894228040143267794;  { 1/Sqrt(2*Pi) }
LnSqrt2Pi  = 0.91893853320467274178;  { Ln(Sqrt(2*Pi)) }
Ln2PiDiv2  = 0.91893853320467274178;  { Ln(2*Pi)/2 }
Sqrt2      = 1.41421356237309504880;  { Sqrt(2) }
Sqrt2Div2  = 0.70710678118654752440;  { Sqrt(2)/2 }
Gold       = 1.61803398874989484821;  { Golden Mean = (1 + Sqrt(5))/2 }
CGold      = 0.38196601125010515179;  { 2 - GOLD }
MachEp = 2.220446049250313E-16;   { 2^(-52) }
MaxNum = 1.797693134862315E+308;  { 2^1024 }
MinNum = 2.225073858507202E-308;  { 2^(-1022) }
MaxLog = 709.7827128933840;
MinLog = -708.3964185322641;
MaxFac = 170;
MaxGam = 171.624376956302;
MaxLgm = 2.556348E+305;
SingleCompareDelta   = 1.0E-34;
DoubleCompareDelta   = 1.0E-280;
{$IFDEF CLR}
ExtendedCompareDelta = DoubleCompareDelta;
{$ELSE}
ExtendedCompareDelta = 1.0E-4400;
{$ENDIF}
Bytes1KB  = 1024;
Bytes1MB  = 1024 * Bytes1KB;
Bytes1GB  = 1024 * Bytes1MB;
Bytes64KB = 64 * Bytes1KB;
Bytes64MB = 64 * Bytes1MB;
Bytes2GB  = 2 * LongWord(Bytes1GB);
  clBlack32', $FF000000 ));
  clDimGray32', $FF3F3F3F ));
  clGray32', $FF7F7F7F ));
  clLightGray32', $FFBFBFBF ));
  clWhite32', $FFFFFFFF ));
  clMaroon32', $FF7F0000 ));
  clGreen32', $FF007F00 ));
  clOlive32', $FF7F7F00 ));
  clNavy32', $FF00007F ));
  clPurple32', $FF7F007F ));
  clTeal32', $FF007F7F ));
  clRed32', $FFFF0000 ));
  clLime32', $FF00FF00 ));
  clYellow32', $FFFFFF00 ));
  clBlue32', $FF0000FF ));
  clFuchsia32', $FFFF00FF ));
  clAqua32', $FF00FFFF ));
  clAliceBlue32', $FFF0F8FF ));
  clAntiqueWhite32', $FFFAEBD7 ));
  clAquamarine32', $FF7FFFD4 ));
  clAzure32', $FFF0FFFF ));
  clBeige32', $FFF5F5DC ));
  clBisque32', $FFFFE4C4 ));
  clBlancheDalmond32', $FFFFEBCD ));
  clBlueViolet32', $FF8A2BE2 ));
  clBrown32', $FFA52A2A ));
  clBurlyWood32', $FFDEB887 ));
  clCadetblue32', $FF5F9EA0 ));
  clChartReuse32', $FF7FFF00 ));
  clChocolate32', $FFD2691E ));
  clCoral32', $FFFF7F50 ));
  clCornFlowerBlue32', $FF6495ED ));
  clCornSilk32', $FFFFF8DC ));
  clCrimson32', $FFDC143C ));
  clDarkBlue32', $FF00008B ));
  clDarkCyan32', $FF008B8B ));
  clDarkGoldenRod32', $FFB8860B ));
  clDarkGray32', $FFA9A9A9 ));
  clDarkGreen32', $FF006400 ));
  clDarkGrey32', $FFA9A9A9 ));
  clDarkKhaki32', $FFBDB76B ));
  clDarkMagenta32', $FF8B008B ));
  clDarkOliveGreen32', $FF556B2F ));
  clDarkOrange32', $FFFF8C00 ));
  clDarkOrchid32', $FF9932CC ));
  clDarkRed32', $FF8B0000 ));
  clDarkSalmon32', $FFE9967A ));
  clDarkSeaGreen32', $FF8FBC8F ));
  clDarkSlateBlue32', $FF483D8B ));
  clDarkSlateGray32', $FF2F4F4F ));
  clDarkSlateGrey32', $FF2F4F4F ));
  clDarkTurquoise32', $FF00CED1 ));
  clDarkViolet32', $FF9400D3 ));
  clDeepPink32', $FFFF1493 ));
```

```
    clDeepSkyBlue32', $FF00BFFF ));
    clDodgerBlue32', $FF1E90FF ));
    clFireBrick32', $FFB22222 ));
    clFloralWhite32', $FFFFFAF0 ));
    clGainsBoro32', $FFDCDCDC ));
    clGhostWhite32', $FFF8F8FF ));
    clGold32', $FFFFD700 ));
    clGoldenRod32', $FFDAA520 ));
    clGreenYellow32', $FFADFF2F ));
    clGrey32', $FF808080 ));
    clHoneyDew32', $FFF0FFF0 ));
    clHotPink32', $FFFF69B4 ));
    clIndianRed32', $FFCD5C5C ));
    clIndigo32', $FF4B0082 ));
    clIvory32', $FFFFFFF0 ));
    clKhaki32', $FFF0E68C ));
    clLavender32', $FFE6E6FA ));
    clLavenderBlush32', $FFFFF0F5 ));
    clLawnGreen32', $FF7CFC00 ));
    clLemonChiffon32', $FFFFFACD ));
    clLightBlue32', $FFADD8E6 ));
    clLightCoral32', $FFF08080 ));
    clLightCyan32', $FFE0FFFF ));
    clLightGoldenRodYellow32', $FFFAFAD2 ));
    clLightGreen32', $FF90EE90 ));
    clLightGrey32', $FFD3D3D3 ));
    clLightPink32', $FFFFB6C1 ));
    clLightSalmon32', $FFFFA07A ));
    clLightSeagreen32', $FF20B2AA ));
    clLightSkyblue32', $FF87CEFA ));
    clLightSlategray32', $FF778899 ));
    clLightSlategrey32', $FF778899 ));
    clLightSteelblue32', $FFB0C4DE ));
    clLightYellow32', $FFFFFFE0 ));
    clLtGray32', $FFC0C0C0 ));
    clMedGray32', $FFA0A0A4 ));
    clDkGray32', $FF808080 ));
    clMoneyGreen32', $FFC0DCC0 ));
    clLegacySkyBlue32', $FFA6CAF0 ));
    clCream32', $FFFFFBF0 ));
    clLimeGreen32', $FF32CD32 ));
    clLinen32', $FFFAF0E6 ));
    clMediumAquamarine32', $FF66CDAA ));
    clMediumBlue32', $FF0000CD ));
    clMediumOrchid32', $FFBA55D3 ));
    clMediumPurple32', $FF9370DB ));
    clMediumSeaGreen32', $FF3CB371 ));
    clMediumSlateBlue32', $FF7B68EE ));
    clMediumSpringGreen32', $FF00FA9A ));
    clMediumTurquoise32', $FF48D1CC ));
    clMediumVioletRed32', $FFC71585 ));
    clMidnightBlue32', $FF191970 ));
    clMintCream32', $FFF5FFFA ));
    clMistyRose32', $FFFFE4E1 ));
    clMoccasin32', $FFFFE4B5 ));
    clNavajoWhite32', $FFFFDEAD ));
    clOldLace32', $FFFDF5E6 ));
    clOliveDrab32', $FF6B8E23 ));
    clOrange32', $FFFFA500 ));
    clOrangeRed32', $FFFF4500 ));
    clOrchid32', $FFDA70D6 ));
    clPaleGoldenRod32', $FFEEE8AA ));
    clPaleGreen32', $FF98FB98 ));
    clPaleTurquoise32', $FFAFEEEE ));
    clPaleVioletred32', $FFDB7093 ));
    clPapayaWhip32', $FFFFEFD5 ));
    clPeachPuff32', $FFFFDAB9 ));
    clPeru32', $FFCD853F ));
    clPlum32', $FFDDA0DD ));
    clPowderBlue32', $FFB0E0E6 ));
    clRosyBrown32', $FFBC8F8F ));
    clRoyalBlue32', $FF4169E1 ));
    clSaddleBrown32', $FF8B4513 ));
    clSalmon32', $FFFA8072 ));
    clSandyBrown32', $FFF4A460 ));
    clSeaGreen32', $FF2E8B57 ));
    clSeaShell32', $FFFFF5EE ));
    clSienna32', $FFA0522D ));
    clSilver32', $FFC0C0C0 ));
    clSkyblue32', $FF87CEEB ));
    clSlateBlue32', $FF6A5ACD ));
    clSlateGray32', $FF708090 ));
    clSlateGrey32', $FF708090 ));
    clSnow32', $FFFFFAFA ));
    clSpringgreen32', $FF00FF7F ));
    clSteelblue32', $FF4682B4 ));
    clTan32', $FFD2B48C ));
    clThistle32', $FFD8BFD8 ));
    clTomato32', $FFFF6347 ));
    clTurquoise32', $FF40E0D0 ));
    clViolet32', $FFEE82EE ));
    clWheat32', $FFF5DEB3 ));
    clWhitesmoke32', $FFF5F5F5 ));
    clYellowgreen32', $FF9ACD32 ));
    clTrWhite32', $7FFFFFFF ));
    clTrBlack32', $7F000000 ));
    clTrRed32', $7FFF0000 ));
    clTrGreen32', $7F00FF00 ));
    clTrBlue32', $7F0000FF ));
  // Fixed point math constants
  FixedOne = $10000;  FixedHalf = $7FFF;
  FixedPI  = Round(PI * FixedOne);
  FixedToFloat = 1/FixedOne;

 Special Types
*****************************************************
  type Complex = record
    X, Y : Float;
  end;
  type TVector     = array of Float;
  TIntVector  = array of Integer;
  TCompVector = array of Complex;
  TBoolVector = array of Boolean;
  TStrVector  = array of String;
  TMatrix     = array of TVector;
  TIntMatrix  = array of TIntVector;
  TCompMatrix = array of TCompVector;
  TBoolMatrix = array of TBoolVector;
  TStrMatrix  = array of TStrVector;
  TByteArray = array[0..32767] of byte; !
  THexArray = array [0..15] of Char; // = '0123456789ABCDEF';
  TBitmapStyle = (bsNormal, bsCentered, bsStretched);
  T2StringArray = array of array of string;
  T2IntegerArray = array of array of integer;
  AddTypeS('INT_PTR', 'Integer
  AddTypeS('LONG_PTR', 'Integer
  AddTypeS('UINT_PTR', 'Cardinal
  AddTypeS('ULONG_PTR', 'Cardinal
  AddTypeS('DWORD_PTR', 'ULONG_PTR
  TIntegerDynArray', 'array of Integer
  TCardinalDynArray', 'array of Cardinal
  TWordDynArray', 'array of Word
  TSmallIntDynArray', 'array of SmallInt
  TByteDynArray', 'array of Byte
  TShortIntDynArray', 'array of ShortInt
  TInt64DynArray', 'array of Int64
  TLongWordDynArray', 'array of LongWord
  TSingleDynArray', 'array of Single
  TDoubleDynArray', 'array of Double
  TBooleanDynArray', 'array of Boolean
  TStringDynArray', 'array of string
  TWideStringDynArray', 'array of WideString
  TDynByteArray      = array of Byte;
  TDynShortintArray = array of Shortint;
  TDynSmallintArray = array of Smallint;
  TDynWordArray      = array of Word;
  TDynIntegerArray  = array of Integer;
  TDynLongintArray  = array of Longint;
  TDynCardinalArray = array of Cardinal;
  TDynInt64Array     = array of Int64;
  TDynExtendedArray = array of Extended;
  TDynDoubleArray    = array of Double;
  TDynSingleArray    = array of Single;
  TDynFloatArray     = array of Float;
  TDynPointerArray  = array of Pointer;
  TDynStringArray    = array of string;
  TSynSearchOption = (ssoMatchCase, ssoWholeWord, ssoBackwards,
    ssoEntireScope, ssoSelectedOnly, ssoReplace, ssoReplaceAll, ssoPrompt);
  TSynSearchOptions = set of TSynSearchOption;



//* Project  : Base Include RunTime Lib for maXbox *Name: pas_includebox.inc
-----------------------------------------------------------------------------
procedure drawPolygon(vPoints: TXYVector; cFrm: TForm);
procedure drawPlot(vPoints: TXYVector; cFrm: TForm; vcolor: integer);
procedure SaveCanvas(vCanvas: TCanvas; FileName: string);
procedure SaveCanvas2(vCanvas: TCanvas; FileName: string);
function CheckStringSum(vstring: string): integer;
function HexToInt(HexNum: string): LongInt;
function IntToBin(Int: Integer): String;
function BinToInt(Binary: String): Integer;
function HexToBin(HexNum: string): string; external2
function BinToHex(Binary: String): string;
function IntToFloat(i: Integer): double;
function AddThousandSeparator(S: string; myChr: Char): string;
function Max3(const X,Y,Z: Integer): Integer;
procedure Swap(var X,Y: char); // faster without inline
procedure ReverseString(var S: String);
function CharToHexStr(Value: Char): string;
function CharToUniCode(Value: Char): string;
function Hex2Dec(Value: Str002): Byte;
function HexStrCodeToStr(Value: string): string;
function HexToStr(i: integer; value: string): string;
function UniCodeToStr(Value: string): string;
function CRC16(statement: string): string;
function SearchForSubstrings(aStrList: TStrings; aSearchStr1, aSearchStr2: string): string;
procedure SearchAndReplace(aStrList: TStrings; aSearchStr, aNewStr: string);
procedure SearchAndCopy(aStrList: TStrings; aSearchStr, aNewStr: string; offset: integer);
Procedure ExecuteCommand(executeFile, paramstring: string);
Procedure ShellExecuteAndWait(executeFile, paramstring: string);
function ExecConsoleApp(const AppName, Parameters: String; AppOutput: TStrings): DWORD;
procedure SearchAndOpenDoc(vfilenamepath: string);
procedure ShowInterfaces(myFile: string);
function Fact2(av: integer): extended;
Function BoolToStr(B: Boolean): string;
Function GCD(x, y : LongInt) : LongInt;
function LCM(m,n: longint): longint;
function GetASCII: string;
function GetItemHeight(Font: TFont): Integer;
function myPlaySound(s: pchar; flag,syncflag: integer): boolean;
function myGetWindowsDirectory(lpBuffer: PChar; uSize: longword): longword;
function getHINSTANCE: longword;
function getHMODULE: longword;
function GetASCII: string;
function ByteIsOk(const AByte: string; var VB: Byte): boolean;
function WordIsOk(const AWord: string; var VW: Word): boolean;
function TwentyFourBitValueIsOk(const AValue: string; var VI: Integer): boolean;
function LongIsOk(const ALong: string; var VC: Cardinal): boolean;
function SafeStr(const s: string): string;
function ExtractUrlPath(const FileName: string): string;
function ExtractUrlName(const FileName: string): string;
function IsInternet: boolean;
function RotateLeft1Bit_u32( Value: uint32): uint32;
procedure LinearRegression(const KnownY:array of Double;const KnownX: array of Double;NData:Int;var
LF:TStLinEst; ErrorStats : Boolean);
procedure getEnvironmentInfo;
procedure AntiFreeze;
function GetCPUSpeed: Double;
function IsVirtualPcGuest : Boolean;
function IsVmWareGuest : Boolean;
procedure StartSerialDialog;
function IsWoW64: boolean;
function IsWow64String(var s: string): Boolean;
procedure StartThreadDemo;
Function RGB(R,G,B: Byte): TColor;
Function Sendln(amess: string): boolean;
Procedure maXbox;
Function AspectRatio(aWidth, aHeight: Integer): String;
function wget(aURL, afile: string): boolean;
procedure PrintList(Value: TStringList);
procedure PrintImage(aValue: TBitmap; Style: TBitmapStyle);
procedure getEnvironmentInfo;
procedure AntiFreeze;
function getBitmap(apath: string): TBitmap;
procedure ShowMessageBig(const aText : string);
function YesNoDialog(const ACaption, AMsg: string): boolean;
procedure SetArrayLength2String(arr: T2StringArray; asize1, asize2: integer);
procedure SetArrayLength2Integer(arr: T2IntegerArray; asize1, asize2: integer);
//function myStrToBytes(const Value: String): TBytes;
//function myBytesToStr(const Value: TBytes): String;
function SaveAsExcelFile(AGrid: TStringGrid; ASheetName, AFileName: string; open: boolean): Boolean;
function getBitmap(apath: string): TBitmap;
procedure ShowMessageBig(const aText : string);
Function StrToBytes(const Value: String): TBytes;
Function BytesToStr(const Value: TBytes): String;
function SaveAsExcelFile(AGrid: TStringGrid; ASheetName, AFileName: string; open: boolean): Boolean;
function ReverseDNSLookup(const IPAdrs:String;const DNSServer:String;Timeout,Retries:Int;var
HostName:String):Bool;
function FindInPaths(const fileName, paths : String) : String;
procedure initHexArray(var hexn: THexArray);
function josephusG(n,k: integer; var graphout: string): integer;
function isPowerof2(num: int64): boolean;
function powerOf2(exponent: integer): int64;
function getBigPI: string;
procedure MakeSound(Frequency{Hz}, Duration{mSec}: Integer; Volume: TVolumeLevel; savefilePath: string);
 function GetASCIILine: string;
procedure MakeComplexSound(N:integer{stream # to use}; freqlist:TStrings; Duration{mSec}: Integer;
                           pinknoise: boolean; shape: integer; Volume: TVolumeLevel);
procedure SetComplexSoundElements(freqedt,Phaseedt,AmpEdt,WaveGrp:integer);
procedure AddComplexSoundObjectToList(newf,newp,newa,news:integer; freqlist: TStrings);
function mapfunc(ax, in_min, in_max, out_min, out_max: integer): integer;
function mapmax(ax, in_min, in_max, out_min, out_max: integer): integer;
function isKeypressed: boolean;
function Keypress: boolean;
procedure StrSplitP(const Delimiter: Char; Input: string; const Strings: TStrings);
function ReadReg(Base: HKEY; KeyName, ValueName: string): string;
function ReadRegistry(Base: HKEY; KeyName, ValueName: string): string;
function GetOSName: string;
function GetOSVersion: string;
function GetOSNumber: string;
function getEnvironmentString: string;
procedure StrReplace(var Str: String; Old, New: String);
procedure SendEmail(mFrom,  mTo,  mSubject,  mBody,  mAttachment: variant);
function getTeamViewerID: string;
Procedure RecurseDirectory(Dir:String; IncludeSubs:boolean; callback:TFileCallbackProcedure);
Procedure RecurseDirectory2(Dir : String; IncludeSubs : boolean);
procedure WinInet_HttpGet(const Url: string; Stream:TStream);
procedure GetQrCode2(Width,Height: Word;Correct_Level:string;const Data:string;apath:string);
function StartSocketService: Boolean;
procedure StartSocketServiceForm;
function GetFileList(FileList: TStringlist; apath: string): TStringlist;
function GetFileList1(apath: string): TStringlist;
procedure LetFileList(FileList: TStringlist; apath: string);
procedure StartWeb(aurl: string);
function GetTodayFiles(startdir, amask: string): TStringlist;
function PortTCPIsOpen(dwPort : Word; ipAddressStr: String): boolean;
function JavahashCode(val: string): Integer;
procedure PostKeyEx32(key: Word; const shift: TShiftState; specialkey: Boolean);
procedure SaveBytesToFile2(const Data: Sysutils.TBytes; const FileName: string);
Procedure HideWindowForSeconds(secs: integer);   {//3 seconds}
Procedure HideWindowForSeconds2(secs: integer; apphandle, aself: TForm);   {//3 seconds}
Procedure ConvertToGray(Cnv: TCanvas);
function GetFileDate(aFile:string; aWithTime:Boolean):string;
procedure ShowMemory;
function ShowMemory2: string;
function getHostIP: string;
procedure ShowBitmap(bmap: TBitmap);
function GetOsVersionInfo: TOSVersionInfo;      //thx to wischnewski
function CreateDBGridForm(dblist: TStringList): TListbox;


// News of 3.9.8 up
Halt-Stop Program in Menu, WebServer2, Stop Event Recompile,
Conversion Routines, Prebuild Forms, more RCData, DebugOutString
CodeSearchEngine to search code patterns in /examples <Ctrl F3>
JvChart - TJvChart Component - 2009 Public
MemoryLeakReport in ini-file (MEMORYREPORT=Y)
PerlRegEx PCRE obj lib included, Perl & Python Syntax Editor, bitbox3 logic example
TAdoQuery.SQL.Add() fixed, ShLwAPI extensions, Indy HTTPHeader Extensions
DMath DLL included incl. Demos
Interface Navigator menu/View/Intf Navigator
Unit Explorer menu/Debug/Units Explorer
EKON 16 Slides ..\maxbox3\docs\utils Excel Export maXcel
Tutorial 19 WinCOM with Arduino Tutorial 20 RegEx Coding
Script History to 9 Files WebServer light /Options/Addons/WebServer
Full Text Finder, JVSimLogic Simulator Package
Halt-Stop Program in Menu, WebServer2, Stop Event ,
Conversion Routines, Prebuild Forms, CodeSearch
Halt-Stop Program in Menu, WebServer2, Stop Event Recompile,
Conversion Routines, Prebuild Forms, more RCData, DebugOutString
CodeSearchEngine to search code patterns in /examples <Ctrl F3>
JvChart - TJvChart Component - 2009 Public, mXGames, JvgXMLSerializer, TJvPaintFX
Compress-Decompress Zip, Services Tutorial22, Synopse framework, PFDLib
SynEdit API, Macro, Macro Recorder, DLL Spy, Configuration Tutorial
IDE Reflection API, Session Service Shell S3
additional SynEdit API, isKeyPressed Routine, Bookmarks, OpenToolsAPI Catalog (OTAC)
Class TMonitor, Configuration Tutorial maxbox_starter25.pdf, Chess.dll Game
arduino map() function, PMRandom Generator
StBarCode Lib, StreamReaderClass, BarCode Package, Astro Package
more ShellAPI, add 32 more units, Simulated Annealing, GenAlgo
REST Test Lib, Multilang Component, Forth Interpreter
New Macros, Sendmail (instant email), DevCUnits, Tetris Addon
DCOM, MDAC, MIDI, TLS support, Posmarks, Utils Addon
Routines for LaTeX/PS, Utils Addon, Indy Package3, TAR Archive, @Callbacks
Routines for LaTeX/PS, Utils Addon, Indy Package3, TAR Archive, @Callbacks
First LCL of Lazarus, CmdLine API, ToDo List, 36 more Units preCompiled
QRCode Service, add more CFunctions like CDateTime of Synapse
Gamma Functions, IndyPackage4, HotLog Threadable, FormTemplateLibrary FTL
Nonlinear regression, ADO Workbench Addon, Assign fixing, IntfNavigator fixing, Applet
30 more Units preCompiled, QRCode Indy Service, more CFunctions like CFill or SRand
RestartDialog, RTF, SQL Scanner, RichEdit, 15 more Units
Tool Section, SOAP Tester, Hot Log Logger2, TCPPortScan, 28 more Units
BOLD Package, Indy Package5, maTRIx. MATHEMAX
SPS Utils WDOS, Plc BitBus (PetriNet), 40 more units
emax layers: system-package-component-unit-class-function-block
HighPrecision Timers,  Indy Package6, AutoDetect, UltraForms
Reversi, GOL, bugfixing, 8 more units, Tutorial 24 Clean Code
Tutorial 18_3 RGB LED, OpenGL Geometry, maxpix, statictext
OpenGL Game Demo: ..Options/Add Ons/Reversi
IBUtils Refactor, InterBase Package, DotNet Routines (JvExControls)
add 31 units, mX4 Introduction Paper, more Socket&Streams, ShortString Routines
7% performance gain (hot spot profiling)
PEP -Pascal Education Program , GSM Module, CGI, PHP Runner
add 42 + 22  (64 units), memcached database, autobookmark, Alcinoe PAC, IPC Lib
Orpheus PAC, AsyncFree Library advapi32 samples, FirebirdExp+MySQL units
```

```
FBX Lib, psAPI, SMS Cell Module, OpenGL, Borland Tools

add routines in 3.9.7.5
097: procedure RIRegister_BarCodeScaner_Routines(S: TPSExec);
996: procedure RIRegister_DBCtrls_Routines(S: TPSExec);
069: procedure RIRegister_IdStrings_Routines(S: TPSExec);
516: procedure RIRegister_JclMultimedia_Routines(S: TPSExec);
215: procedure RIRegister_PNGLoader_Routines(S: TPSExec);
374: procedure RIRegister_SerDlgs_Routines(S: TPSExec);
777: procedure RIRegister_LinarBitmap_Routines(S: TPSExec);

//////////////////////// TestUnits ////////////////////////
  SelftestPEM;
  SelfTestCFundamentUtils;
  SelfTestCFileUtils;
  SelfTestCDateTime;
  SelfTestCTimer;
  SelfTestCRandom;
  Test with e.g.: Assert(PathHasDriveLetter('A:'), 'PathHasDriveLetter
            Assert(WinPathToUnixPath('\c\d.f') = '/c/d.f', 'WinPathToUnixPath

  // Note: There's no need for installing a client certificate in the
  //       webbrowser. The server asks the webbrowser to send a certificate but
  //       if nothing is installed the software will work because the server
  //       doesn't check to see if a client certificate was supplied. If you want you can install:
  //
  //       file: c_cacert.p12
  //       password: c_cakey

  TGraphicControl = class(TControl)
  private
    FCanvas: TCanvas;
    procedure WMPaint(var Message: TWMPaint); message WM_PAINT;
  protected
    procedure Paint; virtual;
    property Canvas: TCanvas read FCanvas;
  public
    constructor Create(AOwner: TComponent); override;
    destructor Destroy; override;
  end;

  TCustomControl = class(TWinControl)
  private
    FCanvas: TCanvas;
    procedure WMPaint(var Message: TWMPaint); message WM_PAINT;
  protected
    procedure Paint; virtual;
    procedure PaintWindow(DC: HDC); override;
    property Canvas: TCanvas read FCanvas;
  public
    constructor Create(AOwner: TComponent); override;
    destructor Destroy; override;
  end;
    RegisterPublishedProperties;
    ('ONCHANGE', 'TNotifyEvent', iptrw);
    ('ONCLICK', 'TNotifyEvent', iptrw);
    ('ONDBLCLICK', 'TNotifyEvent', iptrw);
    ('ONENTER', 'TNotifyEvent', iptrw);
    ('ONEXIT', 'TNotifyEvent', iptrw);
    ('ONKEYDOWN', 'TKeyEvent', iptrw);
    ('ONKEYPRESS', 'TKeyPressEvent', iptrw);
    ('ONMOUSEDOWN', 'TMouseEvent', iptrw);
    ('ONMOUSEMOVE', 'TMouseMoveEvent', iptrw);
    ('ONMOUSEUP', 'TMouseEvent', iptrw);
//**************************************************************************
  // To stop the while loop, click on Options/Show Include (boolean switch)!
  Control a loop in a script with a form event:
  IncludeON;   //control the while loop
  while maxform1.ShowInclude1.checked do begin  //menu event Options/Show Include

//------------------------------------------------------------------------------
//*************mX4 ini-file Configuration*************************************
//-------------------------------------------------------------------------------
  using config file maxboxdef.ini       menu/Help/Config File

//*** Definitions for maXbox mX3 ***
[FORM]
LAST_FILE=E:\maXbox\maxbox3\examples\140_drive_typedemo.txt //history up to 10 files
FONTSIZE=14
EXTENSION=txt
SCREENX=1386
SCREENY=1077
MEMHEIGHT=350
PRINTFONT=Courier New //GUI Settings
LINENUMBERS=Y   //line numbers at gutter in editor at left side
EXCEPTIONLOG=Y  //store excepts and success in 2 log files see below! – menu Debug/Show Last Exceptions
EXECUTESHELL=Y  //prevents execution of ExecuteShell() or ExecuteCommand()
BOOTSCRIPT=Y    //enabling load a boot script
MEMORYREPORT=Y  //shows memory report on closing maXbox
```

```
MACRO=Y         //expand macros (see below) in code e.g. #path:E:\maxbox\maxbox3\docs\
NAVIGATOR=N     //shows function list at the right side of editor
NAVWIDTH=350    //width of the right side interface list <CTRL L>
AUTOBOOKMARK=Y  //sets on all functions a bookmark to jump
[WEB]
IPPORT=8080     //for internal webserver – menu /Options/Add Ons/WebServer
IPHOST=192.168.1.53
ROOTCERT=filepathY
SCERT=filepathY
RSAKEY=filepathY
VERSIONCHECK=Y

using Logfile: maxboxlog.log  , Exceptionlogfile: maxboxerrorlog.txt

Also possible to set report memory in script to override ini setting
procedure Set_ReportMemoryLeaksOnShutdown(abo: boolean)


After Change the ini file you can reload the file with ../Help/Config Update

//---------------------------------------------------------------------------
//**************mX4 maildef.ini ini-file Configuration*********************
//-----------------------------------------------------------------------------
//*** Definitions for maXMail ***
//sendemail, HOST=mail.hover.com, PORT=465 (SSL)
[MAXMAIL]
HOST=getmail.softwareschule.ch
USER=mailusername
PASS=password
PORT=110
SSL=Y
BODY=Y
LAST=5

ADO Connection String:
Provider=MSDASQL;DSN=mx3base;Uid=sa;Pwd=admin

OpenSSL Lib:  unit ssl_openssl_lib;
{$IFDEF CIL}
const
  {$IFDEF LINUX}
  DLLSSLName = 'libssl.so';
  DLLUtilName = 'libcrypto.so';
  {$ELSE}
  DLLSSLName = 'ssleay32.dll';
  DLLUtilName = 'libeay32.dll';
  {$ENDIF}
{$ELSE}
var
  {$IFNDEF MSWINDOWS}
    {$IFDEF DARWIN}
    DLLSSLName: string = 'libssl.dylib';
    DLLUtilName: string = 'libcrypto.dylib';
    {$ELSE}
    DLLSSLName: string = 'libssl.so';
    DLLUtilName: string = 'libcrypto.so';
    {$ENDIF}
  {$ELSE}
  DLLSSLName: string = 'ssleay32.dll';
  DLLSSLName2: string = 'libssl32.dll';
  DLLUtilName: string = 'libeay32.dll';
  {$ENDIF}
{$ENDIF}



//---------------------------------------------------------------------------
//**************mX4 Macro Tags  ******************************************************
//--------------------------------------------------------- --------------------

  asm #name #hostmAPSN2APSN211le, #head,max: APSN21: 04.01.2014 19:05:50 E:\maxbox\maxbox3\docs\maxbox_extract_funclist399.txt end

//Tag Macros

  asm #name, #date, #host, #path, #file, #head, #sign #tech #net end

//Tag Macros
10188: SearchAndCopy(memo1.lines, '#name', getUserNameWin, 11);
10189: SearchAndCopy(memo1.lines, '#date', datetimetoStr(now), 11);
10190: SearchAndCopy(memo1.lines, '#host', getComputernameWin, 11);
10191: SearchAndCopy(memo1.lines, '#path', fpath, 11);
10192: SearchAndCopy(memo1.lines, '#file', fname, 11);
10199  SearchAndCopy(memo1.lines, '#fils', fname +' '+SHA1(Act_Filename), 11);
10193: SearchAndCopy(memo1.lines, '#locs', intToStr(getCodeEnd), 11);
10194: SearchAndCopy(memo1.lines, '#perf', perftime, 11);
10195: SearchAndCopy(memo1.lines, '#sign', Format('%s: %s: %s',
       [getUserNameWin, getComputernameWin, datetimetoStr(now),
10196: SearchAndCopy(memo1.lines, '#head',Format('%s: %s: %s %s ',
10197: [getUserNameWin, getComputernameWin, datetimetoStr(now), Act_Filename]),11);
       [getUserNameWin, getComputernameWin, datetimetoStr(now), Act_Filename]),11);
10198: SearchAndCopy(memo1.lines, '#tech',Format('perf: %s threads: %d %s %s',
       [perftime, numprocessthreads, getIPAddress(getComputerNameWin), timetoStr(time), mbversion]), 11);
10298:  SearchAndCopy(memo1.lines, '#net',Format('DNS: %s; local IPs: %s; local IP: %s',
       [getDNS, GetLocalIPs, getIPAddress(getComputerNameWin)]), 10);

//#tech!perf: 0:0:29.297 threads: 3 192.168.174.1 19:26:30

//Replace Macros
  SearchAndCopy(memo1.lines, '<TIME>', timetoStr(time), 6);
  SearchAndCopy(memo1.lines, '<DATE>', datetoStr(date), 6);
  SearchAndCopy(memo1.lines, '<PATH>', fpath, 6);
  SearchAndCopy(memo1.lines, '<EXEPATH>', EXEPath, 9);
  SearchAndCopy(memo1.lines, '<FILE>', fname, 6);
  SearchAndCopy(memo1.lines, '<SOURCE>', ExePath+'Source', 8);

10198: SearchAndCopy(memo1.lines, '#tech'perf:  threads: 2 192.168.1.53 19:05:50 3.9.9.84
       [perftime,numprocessthreads,getIPAddress(getComputerNameWin),timetoStr(time),mbversion]), 11);
//#tech!84perf:  threads: 2 192.168.1.53 19:05:50 3.9.9.84
  SearchAndCopy(memo1.lines, 'maxbox_extract_funclist399.txt

//---------------------------------------------------------------------------
//*************mX4 ToDo List Tags  ../Help/ToDo List*************************
//------------------------------------------------------- --------------------

    while I < sl.Count do begin
//      if MatchesMask(sl[I], '*/? TODO ([a-z0-9_]*#[1-9]#)*:*') then
      if MatchesMask(sl[I], '*/? TODO (?*#?#)*:*') then
        BreakupToDo(Filename, sl, I, 'TODO', True, True) // full info TODO
      else if MatchesMask(sl[I], '*/? DONE (?*#?#)*:*') then
        BreakupToDo(Filename, sl, I, 'DONE', True, True) // full info DONE
      else if MatchesMask(sl[I], '*/? TODO (#?#)*:*') then
        BreakupToDo(Filename, sl, I, 'TODO', False, True) // only priority info TODO
      else if MatchesMask(sl[I], '*/? DONE (#?#)*:*') then
        BreakupToDo(Filename, sl, I, 'DONE', False, True) // only priority info DONE
      else if MatchesMask(sl[I], '*/?*TODO*:*') then
        BreakupToDo(Filename, sl, I, 'TODO', False, False) // custom TODO
      else if MatchesMask(sl[I], '*/?*DONE*:*') then
        BreakupToDo(Filename, sl, I, 'DONE', False, False); // custom DONE
      Inc(I);
    end;


//---------------------------------------------------------------------------
//*************mX4 Public  Tools API *****************************************
//-----------------------------------------------------------------------------
  file : unit uPSI_fMain.pas;               {$OTAP} Open Tools API Catalog
 // Those functions concern the editor and preprocessor, all of the IDE
 Example: Call it with maxform1.Info1Click(self)
 Note: Call all Methods with maxForm1., e.g.:
                       maxForm1.ShellStyle1Click(self);

procedure SIRegister_fMain(CL: TPSPascalCompiler);
begin
 Const('BYTECODE','String 'bytecode.txt'
 Const('PSTEXT','String PS Scriptfiles (*.txt)|*.TXT)'
 Const('PSMODEL','String PS Modelfiles (*.uc)|*.UC
 Const('PSPASCAL','String PS Pascalfiles (*.pas)|*.PAS
 Const('PSINC','String PS Includes (*.inc)|*.INC
 Const('DEFFILENAME','String 'firstdemo.txt
 Const('DEFINIFILE','String 'maxboxdef.ini
 Const('EXCEPTLOGFILE','String 'maxboxerrorlog.txt
 Const('ALLFUNCTIONSLIST','String 'upsi_allfunctionslist.txt
 Const('ALLFUNCTIONSLISTPDF','String 'maxbox_functions_all.pdf
 Const('ALLOBJECTSLIST','String 'docs\VCL.pdf
 Const('ALLRESOURCELIST','String 'docs\upsi_allresourcelist.txt
 Const('ALLUNITLIST','String 'docs\maxbox3_9.xml');
 Const('INCLUDEBOX','String 'pas_includebox.inc
 Const('BOOTSCRIPT','String 'maxbootscript.txt
 Const('MBVERSION','String '3.9.9.94
 Const('VERSION','String'3.9.9.94
 Const('MBVER','String '399
 Const('MBVERI','Integer'(399);
 Const('MBVERIALL','Integer'(39994);
 Const('EXENAME','String 'maXbox3.exe
 Const('MXSITE','String 'http://www.softwareschule.ch/maxbox.htm
 Const('MXVERSIONFILE','String 'http://www.softwareschule.ch/maxvfile.txt
 Const('MXINTERNETCHECK','String 'www.ask.com
 Const('MXMAIL','String 'max@kleiner.com
 Const('TAB','Char #$09);
 Const('CODECOMPLETION','String 'bds_delphi.dci
 SIRegister_TMaxForm1(CL);
end;

  with FindClass('TForm'),'TMaxForm1') do begin
    memo2', 'TMemo', iptrw);
    memo1', 'TSynMemo', iptrw);
    CB1SCList', 'TComboBox', iptrw);
    mxNavigator', 'TComboBox', iptrw);
    IPHost', 'string', iptrw);
    IPPort', 'integer', iptrw);
    COMPort', 'integer', iptrw);      //3.9.6.4
    Splitter1', 'TSplitter', iptrw);
    PSScript', 'TPSScript', iptrw);
    PS3DllPlugin', 'TPSDllPlugin', iptrw);
    MainMenu1', 'TMainMenu', iptrw);
    Program1', 'TMenuItem', iptrw);
    Compile1', 'TMenuItem', iptrw);
    Files1', 'TMenuItem', iptrw);
    open1', 'TMenuItem', iptrw);
    Save2', 'TMenuItem', iptrw);
    Options1', 'TMenuItem', iptrw);
    Savebefore1', 'TMenuItem', iptrw);
    Largefont1', 'TMenuItem', iptrw);
    sBytecode1', 'TMenuItem', iptrw);
    Saveas3', 'TMenuItem', iptrw);
    Clear1', 'TMenuItem', iptrw);
    Slinenumbers1', 'TMenuItem', iptrw);
    About1', 'TMenuItem', iptrw);
    Search1', 'TMenuItem', iptrw);
    SynPasSyn1', 'TSynPasSyn', iptrw);
    memo1', 'TSynMemo', iptrw);
    SynEditSearch1', 'TSynEditSearch', iptrw);
    WordWrap1', 'TMenuItem', iptrw);
    XPManifest1', 'TXPManifest', iptrw);
    SearchNext1', 'TMenuItem', iptrw);
    Replace1', 'TMenuItem', iptrw);
    PSImport_Controls1', 'TPSImport_Controls', iptrw);
    PSImport_Classes1', 'TPSImport_Classes', iptrw);
    ShowInclude1', 'TMenuItem', iptrw);
    SynEditPrint1', 'TSynEditPrint', iptrw);
    Printout1', 'TMenuItem', iptrw);
    mnPrintColors1', 'TMenuItem', iptrw);
    dlgFilePrint', 'TPrintDialog', iptrw);
    dlgPrintFont1', 'TFontDialog', iptrw);
    mnuPrintFont1', 'TMenuItem', iptrw);
    Include1', 'TMenuItem', iptrw);
    CodeCompletionList1', 'TMenuItem', iptrw);
    IncludeList1', 'TMenuItem', iptrw);
    ImageList1', 'TImageList', iptrw);
    ImageList2', 'TImageList', iptrw);
    CoolBar1', 'TCoolBar', iptrw);
    ToolBar1', 'TToolBar', iptrw);
    tbtnLoad', 'TToolButton', iptrw);
    ToolButton2', 'TToolButton', iptrw);
    tbtnFind', 'TToolButton', iptrw);
    tbtnCompile', 'TToolButton', iptrw);
    tbtnTrans', 'TToolButton', iptrw);
    tbtnUseCase', 'TToolButton', iptrw);   //3.8
    toolbtnTutorial', 'TToolButton', iptrw);
    tbtn6res', 'TToolButton', iptrw);
    ToolButton5', 'TToolButton', iptrw);
    ToolButton1', 'TToolButton', iptrw);
    ToolButton3', 'TToolButton', iptrw);
    statusBar1', 'TStatusBar', iptrw);
    SaveOutput1', 'TMenuItem', iptrw);
    ExportClipboard1', 'TMenuItem', iptrw);
    Close1', 'TMenuItem', iptrw);
    Manual1', 'TMenuItem', iptrw);
    About2', 'TMenuItem', iptrw);
    loadLastfile1', 'TMenuItem', iptrw);
    imglogo', 'TImage', iptrw);
    cedebug', 'TPSScriptDebugger', iptrw);
    debugPopupMenu1', 'TPopupMenu', iptrw);
    BreakPointMenu', 'TMenuItem', iptrw);
    Decompile1', 'TMenuItem', iptrw);
    StepInto1', 'TMenuItem', iptrw);
    StepOut1', 'TMenuItem', iptrw);
    Reset1', 'TMenuItem', iptrw);
    DebugRun1', 'TMenuItem', iptrw);
    PSImport_ComObj1', 'TPSImport_ComObj', iptrw);
    PSImport_StdCtrls1', 'TPSImport_StdCtrls', iptrw);
    PSImport_Forms1', 'TPSImport_Forms', iptrw);
    PSImport_DateUtils1', 'TPSImport_DateUtils', iptrw);
    tutorial4', 'TMenuItem', iptrw);
    ExporttoClipboard1', 'TMenuItem', iptrw);
    ImportfromClipboard1', 'TMenuItem', iptrw);
    N4', 'TMenuItem', iptrw);
    N5', 'TMenuItem', iptrw);
    N6', 'TMenuItem', iptrw);
    ImportfromClipboard2', 'TMenuItem', iptrw);
    tutorial1', 'TMenuItem', iptrw);
    N7', 'TMenuItem', iptrw);
    ShowSpecChars1', 'TMenuItem', iptrw);
    OpenDirectory1', 'TMenuItem', iptrw);
    procMess', 'TMenuItem', iptrw);
    tbtnUseCase', 'TToolButton', iptrw);
    ToolButton7', 'TToolButton', iptrw);
    EditFont1', 'TMenuItem', iptrw);
    UseCase1', 'TMenuItem', iptrw);
    tutorial21', 'TMenuItem', iptrw);
    OpenUseCase1', 'TMenuItem', iptrw);
    PSImport_DB1', 'TPSImport_DB', iptrw);
    tutorial31', 'TMenuItem', iptrw);
    SynHTMLSyn1', 'TSynHTMLSyn', iptrw);
    HTMLSyntax1', 'TMenuItem', iptrw);
    ShowInterfaces1', 'TMenuItem', iptrw);
    Tutorial5', 'TMenuItem', iptrw);
    AllFunctionsList1', 'TMenuItem', iptrw);
    ShowLastException1', 'TMenuItem', iptrw);
    PlayMP31', 'TMenuItem', iptrw);
    SynTeXSyn1', 'TSynTeXSyn', iptrw);
    texSyntax1', 'TMenuItem', iptrw);
    N8', 'TMenuItem', iptrw);
    GetEMails1', 'TMenuItem', iptrw);
    SynCppSyn1', 'TSynCppSyn', iptrw);
    CSyntax1', 'TMenuItem', iptrw);
    Tutorial6', 'TMenuItem', iptrw);
    New1', 'TMenuItem', iptrw);
    AllObjectsList1', 'TMenuItem', iptrw);
    LoadBytecode1', 'TMenuItem', iptrw);
    CipherFile1', 'TMenuItem', iptrw);
    N9', 'TMenuItem', iptrw);
    N10', 'TMenuItem', iptrw);
    Tutorial11', 'TMenuItem', iptrw);
    Tutorial71', 'TMenuItem', iptrw);
    UpdateService1', 'TMenuItem', iptrw);
    PascalSchool1', 'TMenuItem', iptrw);
    Tutorial81', 'TMenuItem', iptrw);
    DelphiSite1', 'TMenuItem', iptrw);
    Output1', 'TMenuItem', iptrw);
    TerminalStyle1', 'TMenuItem', iptrw);
    ReadOnly1', 'TMenuItem', iptrw);
    ShellStyle1', 'TMenuItem', iptrw);
    BigScreen1', 'TMenuItem', iptrw);
    Tutorial91', 'TMenuItem', iptrw);
    SaveOutput2', 'TMenuItem', iptrw);
    N11', 'TMenuItem', iptrw);
    SaveScreenshot', 'TMenuItem', iptrw);
    Tutorial101', 'TMenuItem', iptrw);
    SQLSyntax1', 'TMenuItem', iptrw);
    SynSQLSyn1', 'TSynSQLSyn', iptrw);
    Console1', 'TMenuItem', iptrw);
    SynXMLSyn1', 'TSynXMLSyn', iptrw);
    XMLSyntax1', 'TMenuItem', iptrw);
    ComponentCount1', 'TMenuItem', iptrw);
    NewInstance1', 'TMenuItem', iptrw);
    toolbtnTutorial', 'TToolButton', iptrw);
    Memory1', 'TMenuItem', iptrw);
    SynJavaSyn1', 'TSynJavaSyn', iptrw);
    JavaSyntax1', 'TMenuItem', iptrw);
    SyntaxCheck1', 'TMenuItem', iptrw);
    Tutorial10Statistics1', 'TMenuItem', iptrw);
    ScriptExplorer1', 'TMenuItem', iptrw);
    FormOutput1', 'TMenuItem', iptrw);
    ArduinoDump1', 'TMenuItem', iptrw);
    AndroidDump1', 'TMenuItem', iptrw);
    GotoEnd1', 'TMenuItem', iptrw);
    AllResourceList1', 'TMenuItem', iptrw);
    ToolButton4', 'TToolButton', iptrw);
    tbtn6res', 'TToolButton', iptrw);
    Tutorial11Forms1', 'TMenuItem', iptrw);
    Tutorial12SQL1', 'TMenuItem', iptrw);
    ResourceExplore1', 'TMenuItem', iptrw);
    Info1', 'TMenuItem', iptrw);
    N12', 'TMenuItem', iptrw);
    CryptoBox1', 'TMenuItem', iptrw);
    Tutorial13Ciphering1', 'TMenuItem', iptrw);
    CipherFile2', 'TMenuItem', iptrw);
    N13', 'TMenuItem', iptrw);
    ModulesCount1', 'TMenuItem', iptrw);
    AddOns2', 'TMenuItem', iptrw);
    N4GewinntGame1', 'TMenuItem', iptrw);
    DocuforAddOns1', 'TMenuItem', iptrw);
    Tutorial14Async1', 'TMenuItem', iptrw);
    Lessons15Review1', 'TMenuItem', iptrw);
    SynPHPSyn1', 'TSynPHPSyn', iptrw);
    PHPSyntax1', 'TMenuItem', iptrw);
    Breakpoint1', 'TMenuItem', iptrw);
    SerialRS2321', 'TMenuItem', iptrw);
    N14', 'TMenuItem', iptrw);
    SynCSSyn1', 'TSynCSSyn', iptrw);
    CSyntax2', 'TMenuItem', iptrw);
    Calculator1', 'TMenuItem', iptrw);
    tbtnSerial', 'TToolButton', iptrw);
    ToolButton8', 'TToolButton', iptrw);
```

```
Tutorial151', 'TMenuItem', iptrw);
N15', 'TMenuItem', iptrw);
N16', 'TMenuItem', iptrw);
ControlBar1', 'TControlBar', iptrw);
ToolBar2', 'TToolBar', iptrw);
BtnOpen', 'TToolButton', iptrw);
BtnSave', 'TToolButton', iptrw);
BtnPrint', 'TToolButton', iptrw);
BtnColors', 'TToolButton', iptrw);
btnClassReport', 'TToolButton', iptrw);
BtnRotateRight', 'TToolButton', iptrw);
BtnFullSize', 'TToolButton', iptrw);
BtnFitToWindowSize', 'TToolButton', iptrw);
BtnZoomMinus', 'TToolButton', iptrw);
BtnZoomPlus', 'TToolButton', iptrw);
Panel1', 'TPanel', iptrw);
LabelBrettgroesse', 'TLabel', iptrw);
CB1SCList', 'TComboBox', iptrw);
ImageListNormal', 'TImageList', iptrw);
spbtnexplore', 'TSpeedButton', iptrw);
spbtnexample', 'TSpeedButton', iptrw);
spbsaveas', 'TSpeedButton', iptrw);
imglogobox', 'TImage', iptrw);
EnlargeFont1', 'TMenuItem', iptrw);
EnlargeFont2', 'TMenuItem', iptrw);
ShrinkFont1', 'TMenuItem', iptrw);
ThreadDemo1', 'TMenuItem', iptrw);
HEXEditor1', 'TMenuItem', iptrw);
HEXView1', 'TMenuItem', iptrw);
HEXInspect1', 'TMenuItem', iptrw);
SynExporterHTML1', 'TSynExporterHTML', iptrw);
ExporttoHTML1', 'TMenuItem', iptrw);
ClassCount1', 'TMenuItem', iptrw);
HTMLOutput1', 'TMenuItem', iptrw);
HEXEditor2', 'TMenuItem', iptrw);
Minesweeper1', 'TMenuItem', iptrw);
N17', 'TMenuItem', iptrw);
PicturePuzzle1', 'TMenuItem', iptrw);
sbvclhelp', 'TSpeedButton', iptrw);
DependencyWalker1', 'TMenuItem', iptrw);
WebScanner1', 'TMenuItem', iptrw);
View1', 'TMenuItem', iptrw);
mnToolbar1', 'TMenuItem', iptrw);
mnStatusbar2', 'TMenuItem', iptrw);
mnConsole2', 'TMenuItem', iptrw);
mnCoolbar2', 'TMenuItem', iptrw);
mnSplitter2', 'TMenuItem', iptrw);
WebServer1', 'TMenuItem', iptrw);
Tutorial17Server1', 'TMenuItem', iptrw);
Tutorial18Arduino1', 'TMenuItem', iptrw);
SynPerlSyn1', 'TSynPerlSyn', iptrw);
PerlSyntax1', 'TMenuItem', iptrw);
SynPythonSyn1', 'TSynPythonSyn', iptrw);
PythonSyntax1', 'TMenuItem', iptrw);
DMathLibrary1', 'TMenuItem', iptrw);
IntfNavigator1', 'TMenuItem', iptrw);
EnlargeFontConsole1', 'TMenuItem', iptrw);
ShrinkFontConsole1', 'TMenuItem', iptrw);
SetInterfaceList1', 'TMenuItem', iptrw);
popintfList', 'TPopupMenu', iptrw);
intfAdd1', 'TMenuItem', iptrw);
intfDelete1', 'TMenuItem', iptrw);
intfRefactor1', 'TMenuItem', iptrw);
Defactor1', 'TMenuItem', iptrw);
Tutorial19COMArduino1', 'TMenuItem', iptrw);
Tutorial20Regex', 'TMenuItem', iptrw);
N18', 'TMenuItem', iptrw);
ManualE1', 'TMenuItem', iptrw);
FullTextFinder1', 'TMenuItem', iptrw);
Move1', 'TMenuItem', iptrw);
FractalDemo1', 'TMenuItem', iptrw);
Tutorial21Android1', 'TMenuItem', iptrw);
Tutorial0Function1', 'TMenuItem', iptrw);
SimuLogBox1', 'TMenuItem', iptrw);
OpenExamples1', 'TMenuItem', iptrw);
SynJScriptSyn1', 'TSynJScriptSyn', iptrw);
JavaScriptSyntax1', 'TMenuItem', iptrw);
Halt1', 'TMenuItem', iptrw);
CodeSearch1', 'TMenuItem', iptrw);
SynRubySyn1', 'TSynRubySyn', iptrw);
RubySyntax1', 'TMenuItem', iptrw);
Undo1', 'TMenuItem', iptrw);
SynUNIXShellScriptSyn1', 'TSynUNIXShellScriptSyn', iptrw);
LinuxShellScript1', 'TMenuItem', iptrw);
Rename1', 'TMenuItem', iptrw);
spdcodesearch', 'TSpeedButton', iptrw);
Preview1', 'TMenuItem', iptrw);
Tutorial22Services1', 'TMenuItem', iptrw);
Tutorial23RealTime1', 'TMenuItem', iptrw);
```

```
    Configuration1', 'TMenuItem', iptrw);
    MP3Player1', 'TMenuItem', iptrw);
    DLLSpy1', 'TMenuItem', iptrw);
    SynURIOpener1', 'TSynURIOpener', iptrw);
    SynURISyn1', 'TSynURISyn', iptrw);
    URILinksClicks1', 'TMenuItem', iptrw);
    EditReplace1', 'TMenuItem', iptrw);
    GotoLine1', 'TMenuItem', iptrw);
    ActiveLineColor1', 'TMenuItem', iptrw);
    ConfigFile1', 'TMenuItem', iptrw);
    Sort1Intflist', 'TMenuItem', iptrw);
    Redo1', 'TMenuItem', iptrw);
    Tutorial24CleanCode1', 'TMenuItem', iptrw);
    Tutorial25Configuration1', 'TMenuItem', iptrw);
    IndentSelection1', 'TMenuItem', iptrw);
    UnindentSection1', 'TMenuItem', iptrw);
    SkyStyle1', 'TMenuItem', iptrw);
    N19', 'TMenuItem', iptrw);
    CountWords1', 'TMenuItem', iptrw);
    imbookmarkimages', 'TImageList', iptrw);
    Bookmark11', 'TMenuItem', iptrw);
    N20', 'TMenuItem', iptrw);
    Bookmark21', 'TMenuItem', iptrw);
    Bookmark31', 'TMenuItem', iptrw);
    Bookmark41', 'TMenuItem', iptrw);
    SynMultiSyn1', 'TSynMultiSyn', iptrw);

    Procedure IFPS3ClassesPlugin1CompImport( Sender : TObject; x : TPSPascalCompiler)
    Procedure IFPS3ClassesPlugin1ExecImport(Sender: TObject; Exec: TPSExec; x:TPSRuntimeClassImporter);
    Procedure PSScriptCompile( Sender : TPSScript)
    Procedure Compile1Click( Sender : TObject)
    Procedure PSScriptExecute( Sender : TPSScript)
    Procedure open1Click( Sender : TObject)
    Procedure Save2Click( Sender : TObject)
    Procedure Savebefore1Click( Sender : TObject)
    Procedure Largefont1Click( Sender : TObject)
    Procedure FormActivate( Sender : TObject)
    Procedure SBytecode1Click( Sender : TObject)
    Procedure FormKeyPress( Sender : TObject; var Key : Char)
    Procedure Saveas3Click( Sender : TObject)
    Procedure Clear1Click( Sender : TObject)
    Procedure Slinenumbers1Click( Sender : TObject)
    Procedure About1Click( Sender : TObject)
    Procedure Search1Click( Sender : TObject)
    Procedure FormCreate( Sender : TObject)
    Procedure Memo1ReplaceText(Sender:TObject;const ASearch,AReplace:String;Line,Column:Integer;
                                                var Action : TSynReplaceAction)
    Procedure Memo1StatusChange( Sender : TObject; Changes : TSynStatusChanges)
    Procedure WordWrap1Click( Sender : TObject)
    Procedure SearchNext1Click( Sender : TObject)
    Procedure Replace1Click( Sender : TObject)
    Function PSScriptNeedFile(Sender:TObject;const OrginFileName:String;var FName,Output:String):Bool;
    Procedure ShowInclude1Click( Sender : TObject)
    Procedure Printout1Click( Sender : TObject)
    Procedure mnuPrintFont1Click( Sender : TObject)
    Procedure Include1Click( Sender : TObject)
    Procedure FormDestroy( Sender : TObject)
    Procedure FormClose( Sender : TObject; var Action : TCloseAction)
    Procedure UpdateView1Click( Sender : TObject)
    Procedure CodeCompletionList1Click( Sender : TObject)
    Procedure SaveOutput1Click( Sender : TObject)
    Procedure ExportClipboard1Click( Sender : TObject)
    Procedure Close1Click( Sender : TObject)
    Procedure Manual1Click( Sender : TObject)
    Procedure LoadLastFile1Click( Sender : TObject)
    Procedure Memo1Change( Sender : TObject)
    Procedure Decompile1Click( Sender : TObject)
    Procedure StepInto1Click( Sender : TObject)
    Procedure StepOut1Click( Sender : TObject)
    Procedure Reset1Click( Sender : TObject)
    Procedure cedebugAfterExecute( Sender : TPSScript)
    Procedure cedebugBreakpoint(Sender:TObject; const FileName:String; Position,Row, Col: Cardinal)
    Procedure cedebugCompile( Sender : TPSScript)
    Procedure cedebugExecute( Sender : TPSScript)
    Procedure cedebugIdle( Sender : TObject)
    Procedure cedebugLineInfo( Sender:TObject;const FileName:String; Position, Row, Col : Cardinal)
    Procedure Memo1SpecialLineColors(Sender: TObject; Line:Int; var Special:Boolean;var FG,BG:TColor);
    Procedure BreakPointMenuClick( Sender : TObject)
    Procedure DebugRun1Click( Sender : TObject)
    Procedure tutorial4Click( Sender : TObject)
    Procedure ImportfromClipboard1Click( Sender : TObject)
    Procedure ImportfromClipboard2Click( Sender : TObject)
    Procedure tutorial1Click( Sender : TObject)
    Procedure ShowSpecChars1Click( Sender : TObject)
    Procedure StatusBar1DblClick( Sender : TObject)
    Procedure PSScriptLine( Sender : TObject)
    Procedure OpenDirectory1Click( Sender : TObject)
    Procedure procMessClick( Sender : TObject)
    Procedure tbtnUseCaseClick( Sender : TObject)
    Procedure EditFont1Click( Sender : TObject)
    Procedure tutorial21Click( Sender : TObject)
    Procedure tutorial31Click( Sender : TObject)
    Procedure HTMLSyntax1Click( Sender : TObject)
    Procedure ShowInterfaces1Click( Sender : TObject)
    Procedure Tutorial5Click( Sender : TObject)
    Procedure ShowLastException1Click( Sender : TObject)
    Procedure PlayMP31Click( Sender : TObject)
    Procedure AllFunctionsList1Click( Sender : TObject)
    Procedure texSyntax1Click( Sender : TObject)
    Procedure GetEMails1Click( Sender : TObject)
    procedure DelphiSite1Click(Sender: TObject);
    procedure TerminalStyle1Click(Sender: TObject);
    procedure ReadOnly1Click(Sender: TObject);
    procedure ShellStyle1Click(Sender: TObject);
    procedure Console1Click(Sender: TObject);     //3.2
    procedure BigScreen1Click(Sender: TObject);
    procedure Tutorial91Click(Sender: TObject);
    procedure SaveScreenshotClick(Sender: TObject);
    procedure Tutorial101Click(Sender: TObject);
    procedure SQLSyntax1Click(Sender: TObject);
    procedure XMLSyntax1Click(Sender: TObject);
    procedure ComponentCount1Click(Sender: TObject);
    procedure NewInstance1Click(Sender: TObject);
    procedure CSyntax1Click(Sender: TObject);
    procedure Tutorial6Click(Sender: TObject);
    procedure New1Click(Sender: TObject);
    procedure AllObjectsList1Click(Sender: TObject);
    procedure LoadBytecode1Click(Sender: TObject);
    procedure CipherFile1Click(Sender: TObject);  //V3.5
    procedure NewInstance1Click(Sender: TObject);
    procedure toolbtnTutorialClick(Sender: TObject);
    procedure Memory1Click(Sender: TObject);
    procedure JavaSyntax1Click(Sender: TObject);
    procedure SyntaxCheck1Click(Sender: TObject);
    procedure ScriptExplorer1Click(Sender: TObject);
    procedure FormOutput1Click(Sender: TObject);  //V3.6
    procedure GotoEnd1Click(Sender: TObject);
    procedure AllResourceList1Click(Sender: TObject);
    procedure tbtn6resClick(Sender: TObject);   //V3.7
    procedure Info1Click(Sender: TObject);
    procedure Tutorial10Statistics1Click(Sender: TObject);
    procedure Tutorial11Forms1Click(Sender: TObject);
    procedure Tutorial12SQL1Click(Sender: TObject);  //V3.8
    procedure ResourceExplore1Click(Sender: TObject);
    procedure Info1Click(Sender: TObject);
    procedure CryptoBox1Click(Sender: TObject);
    procedure ModulesCount1Click(Sender: TObject);
    procedure N4GewinntGame1Click(Sender: TObject);
    procedure PHPSyntax1Click(Sender: TObject);
    procedure SerialRS2321Click(Sender: TObject);
    procedure CSyntax2Click(Sender: TObject);
    procedure Calculator1Click(Sender: TObject);
    procedure Tutorial13Ciphering1Click(Sender: TObject);
    procedure Tutorial14Async1Click(Sender: TObject);
    procedure PHPSyntax1Click(Sender: TObject);
    procedure BtnZoomPlusClick(Sender: TObject);
    procedure BtnZoomMinusClick(Sender: TObject);
    procedure btnClassReportClick(Sender: TObject);
    procedure ThreadDemo1Click(Sender: TObject);
    procedure HEXView1Click(Sender: TObject);
    procedure ExporttoHTML1Click(Sender: TObject);
    procedure Minesweeper1Click(Sender: TObject);
    procedure PicturePuzzle1Click(Sender: TObject); //V3.9
    procedure sbvclhelpClick(Sender: TObject);
    procedure DependencyWalker1Click(Sender: TObject);
    procedure CB1SCListDrawItem(Control:TWinControl;Index:Int;aRect:TRect;State:TOwnerDrawState);
    procedure WebScanner1Click(Sender: TObject);
    procedure mnToolbar1Click(Sender: TObject);
    procedure mnStatusbar2Click(Sender: TObject);
    procedure mnConsole2Click(Sender: TObject);
    procedure mnCoolbar2Click(Sender: TObject);
    procedure mnSplitter2Click(Sender: TObject);
    procedure WebServer1Click(Sender: TObject);
    procedure PerlSyntax1Click(Sender: TObject);
    procedure PythonSyntax1Click(Sender: TObject);
    procedure DMathLibrary1Click(Sender: TObject);
    procedure IntfNavigator1Click(Sender: TObject);
    procedure FullTextFinder1Click(Sender: TObject);
    function AppName: string;
    function ScriptName: string;
    function LastName: string;
    procedure FractalDemo1Click(Sender: TObject);
    procedure SimuLogBox1Click(Sender: TObject);
    procedure OpenExamples1Click(Sender: TObject);
    procedure Halt1Click(Sender: TObject);
    procedure Stop;
    procedure CodeSearch1Click(Sender: TObject);
    procedure RubySyntax1Click(Sender: TObject);
    procedure Undo1Click(Sender: TObject);
    procedure LinuxShellScript1Click(Sender: TObject);
    procedure WebScannerDirect(urls: string);
    procedure WebScanner(urls: string);
    procedure LoadInterfaceList2;
    procedure DLLSpy1Click(Sender: TObject);
    procedure Memo1DblClick(Sender: TObject);
    procedure URILinksClicks1Click(Sender: TObject);
    procedure GotoLine1Click(Sender: TObject);
    procedure ConfigFile1Click(Sender: TObject);
    Procedure Sort1IntflistClick( Sender : TObject)
    Procedure Redo1Click( Sender : TObject)
    Procedure Tutorial24CleanCode1Click( Sender : TObject)
    Procedure IndentSelection1Click( Sender : TObject)
    Procedure UnindentSection1Click( Sender : TObject)
    Procedure SkyStyle1Click( Sender : TObject)
    Procedure CountWords1Click( Sender : TObject)
    Procedure Memo1PlaceBookmark( Sender : TObject; var Mark : TSynEditMark)
    Procedure Memo1GutterClick(Sender:TObject;Button:TMouseButton;X,Y,Line:Integer;Mark:TSynEditMark);
    Procedure Bookmark11Click( Sender : TObject)
    Procedure Bookmark21Click( Sender : TObject)
    Procedure Bookmark31Click( Sender : TObject)
    Procedure Bookmark41Click( Sender : TObject)
    Procedure SynMultiSyn1CustomRange(Sender:TSynMultiSyn;Operation:TRangeOperation;var Range:Pointer);
    'STATMemoryReport', 'boolean', iptrw);
    'IPPort', 'integer', iptrw);
    'COMPort', 'integer', iptrw);
    'lbintflist', 'TListBox', iptrw);
    Function GetStatChange : boolean
    Procedure SetStatChange( vstat : boolean)
    Function GetActFileName : string
    Procedure SetActFileName( vname : string)
    Function GetLastFileName : string
    Procedure SetLastFileName( vname : string)
    Procedure WebScannerDirect( urls : string)
    Procedure LoadInterfaceList2
    Function GetStatExecuteShell : boolean
    Procedure DoEditorExecuteCommand( EditorCommand : word)
    function GetActiveLineColor: TColor
    procedure SetActiveLineColor(acolor: TColor)
    procedure ScriptListbox1Click(Sender: TObject);
    procedure Memo2KeyPress(Sender: TObject; var Key: Char);
    procedure EnlargeGutter1Click(Sender: TObject);
    procedure Tetris1Click(Sender: TObject);
    procedure ToDoList1Click(Sender: TObject);
    procedure ProcessList1Click(Sender: TObject);
    procedure MetricReport1Click(Sender: TObject);
    procedure ProcessList1Click(Sender: TObject);
    procedure TCPSockets1Click(Sender: TObject);
    procedure ConfigUpdate1Click(Sender: TObject);
    procedure ADOWorkbench1Click(Sender: TObject);
    procedure SocketServer1Click(Sender: TObject);
    procedure FormDemo1Click(Sender: TObject);
    procedure Richedit1Click(Sender: TObject);
    procedure SimpleBrowser1Click(Sender: TObject);
    procedure DOSShell1Click(Sender: TObject);
    procedure SynExport1Click(Sender: TObject);
    procedure ExporttoRTF1Click(Sender: TObject);
    procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);
    procedure SOAPTester1Click(Sender: TObject);
    procedure Sniffer1Click(Sender: TObject);
    procedure AutoDetectSyntax1Click(Sender: TObject);
    procedure FPlot1Click(Sender: TObject);
    procedure PasStyle1Click(Sender: TObject);
    procedure Tutorial183RGBLED1Click(Sender: TObject);
    procedure Reversi1Click(Sender: TObject);
    procedure ManualmaXbox1Click(Sender: TObject);
    procedure BlaisePascalMagazine1Click(Sender: TObject);
    procedure AddToDo1Click(Sender: TObject);
    procedure CreateGUID1Click(Sender: TObject);
    procedure Tutorial27XML1Click(Sender: TObject);
    procedure CreateDLLStub1Click(Sender: TObject);
    procedure Tutorial28DLL1Click(Sender: TObject);');
    procedure ResetKeyPressed;');
    procedure FileChanges1Click(Sender: TObject);');
    procedure OpenGLTry1Click(Sender: TObject);');
    procedure AllUnitList1Click(Sender: TObject);');
    procedure Tutorial29UMLClick(Sender: TObject);
    procedure CreateHeader1Click(Sender: TObject);

//----------------------------------------------------------------------------
//*************mX4 Editor SynEdit Tools API **********************************
//------------------------------------------------------------------------------
procedure SIRegister_TCustomSynEdit(CL: TPSPascalCompiler);
begin
  //with RegClassS(CL,'TCustomControl', 'TCustomSynEdit') do
  with FindClass('TCustomControl'),'TCustomSynEdit') do begin
    Constructor Create( AOwner : TComponent)
    SelStart', 'Integer', iptrw);
    SelEnd', 'Integer', iptrw); AlwaysShowCaret', 'Boolean', iptrw);
    Procedure UpdateCaret
    Procedure AddKey(Command: TSynEditorCommand;Key1:word;SS1:TShiftState; Key2:word;SS2:TShiftState);
    Procedure AddKey(Command: TSynEditorCommand;Key1:word;SS1:TShiftState; Key2: word;SS2:TShiftState);
    Procedure BeginUndoBlock
    Procedure BeginUpdate
    Function CaretInView : Boolean
    Function CharIndexToRowCol( Index : integer) : TBufferCoord
    Procedure Clear
    Procedure ClearAll
    Procedure ClearBookMark( BookMark : Integer)
    Procedure ClearSelection
    Procedure CommandProcessor( Command : TSynEditorCommand; AChar : char; Data : pointer)
    Procedure ClearUndo
    Procedure CopyToClipboard
    Procedure CutToClipboard
    Procedure DoCopyToClipboard( const SText : string)
    Procedure EndUndoBlock
    Procedure EndUpdate
    Procedure EnsureCursorPosVisible
    Procedure EnsureCursorPosVisibleEx( ForceToMiddle : Boolean)
    Procedure FindMatchingBracket
    Function GetMatchingBracket : TBufferCoord
    Function GetMatchingBracketEx( const APoint : TBufferCoord) : TBufferCoord
    Procedure ExecuteCommand( Command : TSynEditorCommand; AChar : char; Data : pointer)
    Function GetBookMark( BookMark : integer; var X, Y : integer) : boolean
    Function GetHighlighterAttriAtRowCol( const XY: TBufferCoord; var Token : string; var Attri
              : TSynHighlighterAttributes) : boolean
    Function GetHighlighterAttriAtRowColEx( const XY : TBufferCoord; var Token : string;
              var TokenType, Start : Integer; var Attri:TSynHighlighterAttributes):boolean
    Function GetPositionOfMouse( out aPos : TBufferCoord) : Boolean
    Function GetWordAtRowCol( const XY : TBufferCoord) : string
    Procedure GotoBookMark( BookMark : Integer)
    Procedure GotoLineAndCenter( ALine : Integer)
    Function IdentChars : TSynIdentChars
    Procedure InvalidateGutter
    Procedure InvalidateGutterLine( aLine : integer)
    Procedure InvalidateGutterLines( FirstLine, LastLine : integer)
    Procedure InvalidateLine( Line : integer)
    Procedure InvalidateLines( FirstLine, LastLine : integer)
    Procedure InvalidateSelection
    Function IsBookmark( BookMark : integer) : boolean
    Function IsPointInSelection( const Value : TBufferCoord) : boolean
    Procedure LockUndo
    Function BufferToDisplayPos( const p : TBufferCoord) : TDisplayCoord
    Function DisplayToBufferPos( const p : TDisplayCoord) : TBufferCoord
    Function LineToRow( aLine : integer) : integer
    Function RowToLine( aRow : integer) : integer
    Function NextWordPos : TBufferCoord
    Function NextWordPosEx( const XY : TBufferCoord) : TBufferCoord
    Procedure PasteFromClipboard
    Function WordStart : TBufferCoord
    Function WordStartEx( const XY : TBufferCoord) : TBufferCoord
    Function WordEnd : TBufferCoord
    Function WordEndEx( const XY : TBufferCoord) : TBufferCoord
    Function PrevWordPos : TBufferCoord
    Function PrevWordPosEx( const XY : TBufferCoord) : TBufferCoord
    Function PixelsToRowColumn( aX, aY : integer) : TDisplayCoord
    Function PixelsToNearestRowColumn( aX, aY : integer) : TDisplayCoord
    Procedure Redo
    Procedure RegisterCommandHandler(const AHandlerProc:THookedCommandEvent;AHandlerData:pointer));
    Function RowColumnToPixels( const RowCol : TDisplayCoord) : TPoint
    Function RowColToCharIndex( RowCol : TBufferCoord) : integer
    Function SearchReplace( const ASearch, AReplace:string; AOptions:TSynSearchOptions): integer
    Procedure SelectAll
    Procedure SetBookMark( BookMark : Integer; X : Integer; Y : Integer)
    Procedure SetCaretAndSelection( const ptCaret, ptBefore, ptAfter : TBufferCoord)
    Procedure SetDefaultKeystrokes
    Procedure SetSelWord
    Procedure SetWordBlock( Value : TBufferCoord)
    Procedure Undo
    Procedure UnlockUndo
    Procedure UnregisterCommandHandler( AHandlerProc : THookedCommandEvent)
    Procedure AddKeyUpHandler( aHandler : TKeyEvent)
    Procedure RemoveKeyUpHandler( aHandler : TKeyEvent)
    Procedure AddKeyDownHandler( aHandler : TKeyEvent)
    Procedure RemoveKeyDownHandler( aHandler : TKeyEvent)
    Procedure AddKeyPressHandler( aHandler : TKeyPressEvent)
    Procedure RemoveKeyPressHandler( aHandler : TKeyPressEvent)
    Procedure AddFocusControl( aControl : TWinControl)
    Procedure RemoveFocusControl( aControl : TWinControl)
    Procedure AddMouseDownHandler( aHandler : TMouseEvent)
    Procedure RemoveMouseDownHandler( aHandler : TMouseEvent)
    Procedure AddMouseUpHandler( aHandler : TMouseEvent)
    Procedure RemoveMouseUpHandler( aHandler : TMouseEvent)
    Procedure AddMouseCursorHandler( aHandler : TMouseCursorEvent)
    Procedure RemoveMouseCursorHandler( aHandler : TMouseCursorEvent)
    Procedure SetLinesPointer( ASynEdit : TCustomSynEdit)
    Procedure RemoveLinesPointer
    Procedure HookTextBuffer( aBuffer : TSynEditStringList; aUndo, aRedo : TSynEditUndoList)
    Procedure UnHookTextBuffer
    BlockBegin', 'TBufferCoord', iptrw);
    BlockEnd', 'TBufferCoord', iptrw);
    CanPaste', 'Boolean', iptr);
    CanRedo', 'boolean', iptr);
    CanUndo', 'boolean', iptr);
    CaretX', 'Integer', iptrw);
    CaretY', 'Integer', iptrw);
    CaretXY', 'TBufferCoord', iptrw);
    ActiveLineColor', 'TColor', iptrw);
    DisplayX', 'Integer', iptr);
    DisplayY', 'Integer', iptr);
    DisplayXY', 'TDisplayCoord', iptr);
    DisplayLineCount', 'integer', iptr);
    CharsInWindow', 'Integer', iptr);
    CharWidth', 'integer', iptr);
    Font', 'TFont', iptrw);
    GutterWidth', 'Integer', iptr);
    Highlighter', 'TSynCustomHighlighter', iptrw);
    LeftChar', 'Integer', iptrw);
    LineHeight', 'integer', iptr);
    LinesInWindow', 'Integer', iptr);
    LineText', 'string', iptrw);
    Lines', 'TStrings', iptrw);
    Marks', 'TSynEditMarkList', iptr);
    MaxScrollWidth', 'integer', iptrw);
    Modified', 'Boolean', iptrw);
    PaintLock', 'Integer', iptr);
    ReadOnly', 'Boolean', iptrw);
    SearchEngine', 'TSynEditSearchCustom', iptrw);
    SelAvail', 'Boolean', iptr); SelLength', 'integer', iptrw);
    SelTabBlock', 'Boolean', iptr);
    SelTabLine', 'Boolean', iptr);
    SelText', 'string', iptrw);
    StateFlags', 'TSynStateFlags', iptr);
    Text', 'string', iptrw);
    TopLine', 'Integer', iptrw);
    WordAtCursor', 'string', iptr);
    WordAtMouse', 'string', iptr);
    UndoList', 'TSynEditUndoList', iptr);
    RedoList', 'TSynEditUndoList', iptr);
    OnProcessCommand', 'TProcessCommandEvent', iptrw);
    BookMarkOptions', 'TSynBookMarkOpt', iptrw);
    BorderStyle', 'TSynBorderStyle', iptrw);
    ExtraLineSpacing', 'integer', iptrw);
    Gutter', 'TSynGutter', iptrw);
    HideSelection', 'boolean', iptrw);
    InsertCaret', 'TSynEditCaretType', iptrw);
    InsertMode', 'boolean', iptrw);
    IsScrolling', 'Boolean', iptr);
    Keystrokes', 'TSynEditKeyStrokes', iptrw);
    MaxUndo', 'Integer', iptrw);
    Options', 'TSynEditorOptions', iptrw);
    OverwriteCaret', 'TSynEditCaretType', iptrw);
    RightEdge', 'Integer', iptrw); RightEdgeColor', 'TColor', iptrw);
    ScrollHintColor', 'TColor', iptrw);
    ScrollHintFormat', 'TScrollHintFormat', iptrw);
    ScrollBars', 'TScrollStyle', iptrw);
    SelectedColor', 'TSynSelectedColor', iptrw);
    SelectionMode', 'TSynSelectionMode', iptrw);
    ActiveSelectionMode', 'TSynSelectionMode', iptrw);
    TabWidth', 'integer', iptrw);  WantReturns', 'boolean', iptrw);
    WantTabs', 'boolean', iptrw);  WordWrap', 'boolean', iptrw);
    WordWrapGlyph', 'TSynGlyph', iptrw);
    OnChange', 'TNotifyEvent', iptrw);
    OnClearBookmark', 'TPlaceMarkEvent', iptrw);
    OnCommandProcessed', 'TProcessCommandEvent', iptrw);
    OnContextHelp', 'TContextHelpEvent', iptrw);
    OnDropFiles', 'TDropFilesEvent', iptrw);
    OnGutterClick', 'TGutterClickEvent', iptrw);
    OnGutterGetText', 'TGutterGetTextEvent', iptrw);
    OnGutterPaint', 'TGutterPaintEvent', iptrw);
    OnMouseCursor', 'TMouseCursorEvent', iptrw);
    OnPaint', 'TPaintEvent', iptrw);
    OnPlaceBookmark', 'TPlaceMarkEvent', iptrw);
    OnProcessUserCommand', 'TProcessCommandEvent', iptrw);
    OnReplaceText', 'TReplaceTextEvent', iptrw);
    OnSpecialLineColors', 'TSpecialLineColorsEvent', iptrw);
    OnStatusChange', 'TStatusChangeEvent', iptrw);
    OnPaintTransient', 'TPaintTransient', iptrw);
    OnScroll', 'TScrollEvent', iptrw);
   end;
 Procedure RegisterPlaceableHighlighter(highlighter : TSynCustomHighlighterClass)
 Function GetPlaceableHighlighters : TSynHighlighterList
 Function EditorCommandToDescrString( Cmd : TSynEditorCommand) : string
 Function EditorCommandToCodeString( Cmd : TSynEditorCommand) : string
 Procedure GetEditorCommandValues( Proc : TGetStrProc)
 Procedure GetEditorCommandExtended( Proc : TGetStrProc)
 Function IdentToEditorCommand( const Ident : string; var Cmd : longint) : boolean
 Function EditorCommandToIdent( Cmd : longint; var Ident : string) : boolean
 Function ConvertCodeStringToExtended( AString : String) : String
 Function ConvertExtendedToCodeString( AString : String) : String
 Function ConvertExtendedToCommand( AString : String) : TSynEditorCommand
 Function ConvertCodeStringToCommand( AString : String) : TSynEditorCommand
 Function IndexToEditorCommand( const AIndex : Integer) : Integer

  TSynEditorOption = (
    eoAltSetsColumnMode,       //Holding down the Alt Key will put the selection mode into columnar format
    eoAutoIndent,              //Will indent caret on newlines with same amount of leading whitespace as
                               // preceding line
    eoAutoSizeMaxScrollWidth,  //Automatically resizes the MaxScrollWidth property when inserting text
    eoDisableScrollArrows,     //Disables the scroll bar arrow buttons when you can't scroll in that
                               //direction any more
    eoDragDropEditing,         //Allows to select a block of text and drag it within document to another
                               // location
    eoDropFiles,               //Allows the editor accept OLE file drops
    eoEnhanceHomeKey,          //enhances home key positioning, similar to visual studio
    eoEnhanceEndKey,           //enhances End key positioning, similar to JDeveloper
    eoGroupUndo,               //When undoing/redoing actions, handle all continous changes the same kind
                               // in one call
                               //instead undoing/redoing each command separately
    eoHalfPageScroll,          //When scrolling with page-up and page-down commands, only scroll a half
                               //page at a time
    eoHideShowScrollbars,      //if enabled, then scrollbars will only show if necessary.
    If you have ScrollPastEOL, then it the horizontal bar will always be there (it uses MaxLength instead)
    eoKeepCaretX,              //When moving through lines w/o cursor Past EOL, keeps X position of cursor
    eoNoCaret,                 //Makes it so the caret is never visible
    eoNoSelection,             //Disables selecting text
    eoRightMouseMovesCursor,   //When clicking with right mouse for popup menu, moves cursor to location
    eoScrollByOneLess,         //Forces scrolling to be one less
    eoScrollHintFollows,       //The scroll hint follows the mouse when scrolling vertically
    eoScrollPastEof,           //Allows the cursor to go past the end of file marker
    eoScrollPastEol,           //Allows cursor to go past last character into white space at end of a line
    eoShowScrollHint,          //Shows a hint of the visible line numbers when scrolling vertically
    eoShowSpecialChars,        //Shows the special Characters
    eoSmartTabDelete,          //similar to Smart Tabs, but when you delete characters
    eoSmartTabs,               //When tabbing, cursor will go to non-white space character of previous line
    eoSpecialLineDefaultFg,    //disables the foreground text color override using OnSpecialLineColor event
    eoTabIndent,               //If active <Tab>and<Shift><Tab> act block indent,unindent when text select
    eoTabsToSpaces,            //Converts a tab character to a specified number of space characters
    eoTrimTrailingSpaces       //Spaces at the end of lines will be trimmed and not saved

    *********************************Important Editor Short Cuts*****************************);
    Double click to select a word and count words with highlighting.
    Triple click to select a line.
    CTRL+SHIFT+click to extend a selection.
    Drag with the ALT key down to select columns of text !!!
    Drag and drop is supported.
    Type CTRL+Z to undo and SHIFT+CTRL+Z to redo.
    Type CTRL+A to select all.
    Type CTRL+N to set a new line.
    Type CTRL+T to delete a line or token. //Tokenizer
    Type CTRL+C to copy to clipboard. Type CTRL+V to paste from clipboard.
    Type CTRL+Shift+T to add ToDo in line and list.
    Type CTRL+Shift+[0..9] to set bookmarks.   //Bookmark
    Type CTRL[0..9] to jump or get to bookmarks.
    Type Home to position cursor at beginning of current line and End to position it at end of line.
    Type CTRL+Home to position cursor at start of doc and CTRL+End to position it at end of document.
    Page Up and Page Down work as expected.
    CTRL+Page Up sends cursor to top of viewed portion and CTRL+Page Down sends it to bottom.
    using http://pp4s.co.uk/main/tu-form2-help-demo-laz.html

{$ Short Key Positions Ctrl<A-Z>: }
def
   <A> Select All
   <B> Count Words
   <C> Copy
   <D> Internet Start
   <E> Script List
   <F> Find
   <G> Goto
   <H> Mark Line
   <I> Interface List
   <J> Code Completion
   <K> Console
   <L> Interface List Box
   <M> Font Larger -
   <N> New Line
   <O> Open File
   <P> Font Smaller +
   <Q> Quit
   <R> Replace
   <S> Save!
   <T> Delete Line
   <U> Use Case Editor
   <V> Paste
   <W> URI Links
   <X> Reserved for coding use internal
   <Y> Delete Line
   <Z> Undo

ref
   F1 Help
   F2 Syntax Check
   F3 Search Next
   F4 New Instance
   F5 Line Mark /Breakpoint
   F6 Goto End
   F7 Debug Step Into
   F8 Debug Step Out
   F9 Compile
   F10 Menu
   F11 Word Count Highlight
   F12 Reserved for coding use internal

 def ReservedWords: array[0..82] of string =
    ('and', 'array', 'as', 'asm', 'at', 'begin', 'case', 'class', 'const',
     'constructor', 'default', 'destructor', 'dispinterface', 'div', 'do',
     'downto', 'else', 'end', 'except', 'exports', 'file', 'finalization',
     'finally', 'for', 'function', 'goto', 'if', 'implementation', 'in',
     'inherited', 'initialization', 'inline', 'interface', 'is', 'label',
     'library', 'message', 'mod', 'nil', 'not', 'object', 'of', 'on', 'or',
     'out', 'packed', 'procedure', 'program', 'property', 'raise', 'read',
     'record', 'repeat', 'resourcestring', 'set', 'shl', 'shr', 'string',
     'stored', 'then', 'threadvar', 'to', 'try', 'type', 'unit', 'until',
     'uses', 'var', 'while', 'with', 'write', 'xor', 'private', 'protected',
     'public', 'published','def','ref','using','typedef','memo1','memo2','doc','maxform1';
  AllowedChars: array[0..5] of string = ('(',')', '[', ']',' ',' t,t1,t2,t3: boolean;

//-----------------------------------------------------------------------------
//*************End of mX4 Public  Tools API **********************************
//-----------------------------------------------------------------------------

Amount of Functions: 12264
Amount of Procedures: 7632
Amount of Constructors: 1253
Totals of Calls: 21149
SHA1: Win 3.9.9.94 746418954E790A1713A93FF786ABCB50097FCB59

*************************************************************
 Doc Short Manual with 50 Tips!
*************************************************************
- Install: just save your maxboxdef.ini before and then extract the zip file!
- Toolbar: Click on the red maXbox Sign (right on top) opens your work directory or jump to <Help>
- Menu: With <F2> you check syntax with <F8> you debug and <F9> you compile!
- Menu: With <Crtl><F3> you can search for code on examples
- Menu: Open in menu Output a new instance <F4> of the box to compare or prepare your scripts
- Menu: Set Interface Naviagator in menu /View/Intf Navigator
- Menu: Switch or toogle between the last 2 scripts in menu File/LoadLast (History is set to 9 files)

- Inifile: Set memory report in ini: MEMORYREPORT=Y :report on memory leaks on shutdown by dialog
- Inifile: Refresh (reload) the inifile after edit with ../Help/Config Update
- Context Menu: You can printout your scripts as a pdf-file or html-export
- Context: You do have a context menu with the right mouse click

- Menu: With the UseCase Editor you can convert graphic formats too.
- Menu: On menu Options you find Addons as compiled scripts
- IDE: You don't need a mouse to handle maXbox, use shortcuts
- Menu: Check Options/ProcessMessages! if something is wrong or you can't see graphics in a time
- IDE: Dragndrop your scripts in box or the model in use case editor (Cut,Copy,Paste always available)
- Editor: You can get templates as code completion with <ctrl j> in editor like classp or iinterface
         or ttimer (you type classp and then CTRL J),or you type tstringlist and <Ctrl><J>

- Menu: In menu output (console) you can set output menu in edit mode by unchecking <read only output>
- Editor: After the end. you can write or copy notes or descriptions concerning the app or code
- Code: If you code a loop till key-pressed use function: isKeyPressed;
- Code: Macro set the macros #name, #date, #host, #path, #file, #head #sign, Tutorial maxbox_starter25.pdf
- Code: change Syntax in autoboot macro 'maxbootscript.txt'
- Editor: - <F11> Click on Word in Editor search amount of words with highlighting, Dbl Click on Bookmarks
         to delete and Click and mark to drag a bookmark
- Menu: To start handling from CD-ROM (read only mode) uncheck in Menu /Options/Save before Compile
- IDE: A file info with system and script information you find in menu Program/Information
- IDE: After change the config file in help you can update changes in menu Help/Config Update
- IDE: Make a screenshot of the content and environment in menu Output/Save Screenshot
- IDE: Use a boot loader script 'maxbootscript.txt' (as auto start) to change box each time you start it.
- IDE: With escape or <Ctrl> Q you can also leave the box or stop a script in menu program - stop program
- Editor: Set Bookmarks to check your work in app or code
- Editor: With <Ctrl H> you set {$Active Line Color} and F11 you get Word Count Statistic on Output too
- Editor: With {//TODO: some description} or DONE you set code entries for ToDo List in ../Help/ToDo List
- Editor: With <Ctrl W> you set active URL links in your code to test availability in Context Menu

- IDE with  menu /Options/ADO SQL Workbench you can manage your Database
- Context Menu: You can write your docus with RichEdit RTF printout /Editor Form Options/Richedit
- Menu: Set Interface Naviagator also with  toogle <Ctrl L> or /View/Intf Navigator
- Toolbar: In menu /View switch Off Toolbar and Coolbar to get more coding space
- Code: Put some resources in your code /Help/Resource Explorer like bitbtn, forms, dialogs;
- Code Editor: Compile with <F9> but also Alt C in case <F9> isnt available;
- IDE set bookmarks with  <Ctrl Shift> (0..9) and jump with <Ctrl> (0..9)
- IDE menu /Help/Tools/ write with RTF Editor or open a DOS Shell or browse

- Add on write your Requirements in RTF Docu with <CTRL ALT R> in context menu
- Add on when no browser is available start /Options/Add ons/Easy Browser
- Add on SOAP Tester with SOP POST File
- Add on IP Protocol Sniffer with List View
- Add on OpenGL mX Robot Demo for android

- Menu: Help/Tools as a Tool Section with DOS Opener
- Menu Editor: export the code as RTF File
- Menu: Help/Tools/Syn Export your code as available in HTML or RTF
- Menu: Help/Tools/ with <Alt H-S-D> you start the DOS Shell
- Context: Auto Detect of Syntax depending on file extension
- Code: some Windows API function start with w in the name like wGetAtomName();
- IDE Close - if you cant close the box then reset it <Ctrl F2> in menu /Debug
- IDE File Check with menu ..View/File Changes/...

- using DLL example in maXbox: //function: {**********************************************}
   Function GetProcessMemoryInfo(Process: THandle; var MemoryCounters: TProcessMemoryCounters;
                                          cb: DWORD): BOOL; //stdcall;;
     External 'GetProcessMemoryInfo@psapi.dll stdcall';
   Function OpenProcess(dwDesiredAccess:DWORD; bInheritHandle:BOOL; dwProcessId: DWORD):THandle;
     External  'OpenProcess@kernel32.dll stdcall';


PCT Precompile Technology , mX4 ScriptStudio
Indy, JCL, Jedi, VCL, Systools, TurboPower, Fundamentals, ExtendedRTL, Synedit
DMath, devC, Graphics32, ExtPascal, mX4, LCL, CLX, FCL, CPort and more
emax layers: system-package-component-unit-class-function-block
new keywords def ref using maXCalcF
UML: use case act class state seq pac comp dep - lib lab
FBX Lib, psAPI, SMS Cell Module, OpenGL, Borland Tools
Tutorials, 30 Units add, VCL constructors, controls plus, unit list
2 Tutorials, 36 Units add, Synapse V40, LDAP, OpenSSL, AVScan
1 Report, 15 Units add, DBCtrls, Stream+, IRadio, Wininet
1 DLL Report, 24 Units add, DRTable, Remote+, Cindy functions!
DLL Report, UML Tutor, 32 Units add, DRTable, Remote+, Cindy functions!
https://unibe-ch.academia.edu/MaxKleiner
www.slideshare.net/maxkleiner1
http://www.scribd.com/max_kleiner



************************************************************
 unit List asm internal end
************************************************************
01 unit RIRegister_StrUtils_Routines(exec);    //Delphi
02 unit SIRegister_IdStrings                   //Indy Sockets
03 unit RIRegister_niSTRING_Routines(Exec);    //from RegEx
04 unit uPSI_fMain Functions;                  //maXbox Open Tools API
05 unit IFSI_WinForm1puzzle;                   //maXbox
06 unit RIRegister_LinarBitmap_Routines(Exec); //ImageFileLibBCB
07 unit RegisterDateTimeLibrary_R(exec);       //Delphi
08 unit RIRegister_MathMax_Routines(exec);     //Jedi & Delphi
09 unit RIRegister_IdGlobal_Routines(exec);    //Indy Sockets
10 unit RIRegister_SysUtils_Routines(Exec);    //Delphi
11 unit uPSI_IdTCPConnection;                  //Indy some functions
12 unit uPSCompiler.pas;                       //PS kernel functions
13 unit uPSI_DBCommon;                         //DB Common_Routines and Types
14 unit uPSI_Printers.pas                      //Delphi VCL
15 unit uPSI_MPlayer.pas                       //Delphi VCL
16 unit uPSC_comobj;                           //COM Functions
17 unit uPSI_Clipbrd;                          //Delphi VCL
18 unit Filectrl in IFSI_SysUtils_max;         //VCL Runtime
19 unit uPSI_SqlExpr;                          //DBX3
20 unit uPSI_ADODB;                            //ADODB
21 unit uPSI_StrHlpr;                          //String Helper Routines
22 unit uPSI_DateUtils;                        //Expansion to DateTimeLib
23 unit uPSI_FileUtils;                        //Expansion to Sys/File Utils
24 unit JUtils / gsUtils;                      //Jedi / Metabase
25 unit JvFunctions_max;                       //Jedi Functions
26 unit HTTPParser;                            //Delphi VCL
27 unit HTTPUtil;                              //Delphi VCL
28 unit uPSI_XMLUtil;                          //Delphi VCL
29 unit uPSI_SOAPHTTPClient;                   //Delphi VCL SOAP WebService V3.5
30 unit uPSI_Contnrs;                          //Delphi RTL Container of Classes
31 unit uPSI_MaskUtils;                        //RTL Edit and Mask functions
32 unit uPSI_MyBigInt;                         //big integer class with Math
33 unit uPSI_ConvUtils;                        //Delphi VCL Conversions engine
34 unit Types_Variants;                        //Delphi\Win32\rtl\sys
35 unit uPSI_IdHashSHA1;                       //Indy Crypto Lib
36 unit uPSI_IdHashMessageDigest               //Indy Crypto;
37 unit uPSI_IdASN1Util;                       //Indy ASN1Utility Routines;
38 unit uPSI_IdLogFile;                        //Indy Logger from LogBase
39 unit uPSI_IdIcmpClient;                     //Indy Ping ICMP
40 unit uPSI_IdHashMessageDigest_max           //Indy Crypto &OpenSSL;
41 unit uPSI_FileCtrl;                         //Delphi RTL
```

```
42 unit uPSI_Outline;                         //Delphi VCL
43 unit uPSI_ScktComp;                        //Delphi RTL
44 unit uPSI_Calendar;                        //Delphi VCL
45 unit uPSI_VListView                        //VListView;
46 unit uPSI_DBGrids;                         //Delphi VCL
47 unit uPSI_DBCtrls;                         //Delphi VCL
48 unit ide_debugoutput;                      //maXbox
49 unit uPSI_ComCtrls;                        //Delphi VCL
50 unit uPSC_stdctrls+;                       //Delphi VCL
51 unit uPSI_Dialogs;                         //Delphi VCL
52 unit uPSI_StdConvs;                        //Delphi RTL
53 unit uPSI_DBClient;                        //Delphi RTL
54 unit uPSI_DBPlatform;                      //Delphi RTL
55 unit uPSI_Provider;                        //Delphi RTL
56 unit uPSI_FMTBcd;                          //Delphi RTL
57 unit uPSI_DBCGrids;                        //Delphi VCL
58 unit uPSI_CDSUtil;                         //MIDAS
59 unit uPSI_VarHlpr;                         //Delphi RTL
60 unit uPSI_ExtDlgs;                         //Delphi VCL
61 unit sdpStopwatch;                         //maXbox
62 unit uPSI_JclStatistics;                   //JCL
63 unit uPSI_JclLogic;                        //JCL
64 unit uPSI_JclMiscel;                       //JCL
65 unit uPSI_JclMath_max;                     //JCL RTL
66 unit uPSI_uTPLb_StreamUtils;               //LockBox 3
67 unit uPSI_MathUtils;                       //BCB
68 unit uPSI_JclMultimedia;                   //JCL
69 unit uPSI_WideStrUtils;                    //Delphi API/RTL
70 unit uPSI_GraphUtil;                       //Delphi RTL
71 unit uPSI_TypeTrans;                       //Delphi RTL
72 unit uPSI_HTTPApp;                         //Delphi VCL
73 unit uPSI_DBWeb;                           //Delphi VCL
74 unit uPSI_DBBdeWeb;                        //Delphi VCL
75 unit uPSI_DBXpressWeb;                     //Delphi VCL
76 unit uPSI_ShadowWnd;                       //Delphi VCL
77 unit uPSI_ToolWin;                         //Delphi VCL
78 unit uPSI_Tabs;                            //Delphi VCL
79 unit uPSI_JclGraphUtils;                   //JCL
80 unit uPSI_JclCounter;                      //JCL
81 unit uPSI_JclSysInfo;                      //JCL
82 unit uPSI_JclSecurity;                     //JCL
83 unit uPSI_JclFileUtils;                    //JCL
84 unit uPSI_IdUserAccounts;                  //Indy
85 unit uPSI_IdAuthentication;                //Indy
86 unit uPSI_uTPLb_AES;                       //LockBox 3
87 unit uPSI_IdHashSHA1;                      //LockBox 3
88 unit uTPLb_BlockCipher;                    //LockBox 3
89 unit uPSI_ValEdit.pas;                     //Delphi VCL
90 unit uPSI_JvVCLUtils;                      //JCL
91 unit uPSI_JvDBUtil;                        //JCL
92 unit uPSI_JvDBUtils;                       //JCL
93 unit uPSI_JvAppUtils;                      //JCL
94 unit uPSI_JvCtrlUtils;                     //JCL
95 unit uPSI_JvFormToHtml;                    //JCL
96 unit uPSI_JvParsing;                       //JCL
97 unit uPSI_SerDlgs;                         //Toolbox
98 unit uPSI_Serial;                          //Toolbox
99 unit uPSI_JvComponent;                     //JCL
100 unit uPSI_JvCalc;                         //JCL
101 unit uPSI_JvBdeUtils;                     //JCL
102 unit uPSI_JvDateUtil;                     //JCL
103 unit uPSI_JvGenetic;                      //JCL
104 unit uPSI_JclBase;                        //JCL
105 unit uPSI_JvUtils;                        //JCL
106 unit uPSI_JvStrUtil;                      //JCL
107 unit uPSI_JvStrUtils;                     //JCL
108 unit uPSI_JvFileUtil;                     //JCL
109 unit uPSI_JvMemoryInfos;                  //JCL
110 unit uPSI_JvComputerInfo;                 //JCL
111 unit uPSI_JvgCommClasses;                 //JCL
112 unit uPSI_JvgLogics;                      //JCL
113 unit uPSI_JvLED;                          //JCL
114 unit uPSI_JvTurtle;                       //JCL
115 unit uPSI_SortThds; unit uPSI_ThSort;     //maXbox
116 unit uPSI_JvgUtils;                       //JCL
117 unit uPSI_JvExprParser;                   //JCL
118 unit uPSI_HexDump;                        //Borland
119 unit uPSI_DBLogDlg;                       //VCL
120 unit uPSI_SqlTimSt;                       //RTL
121 unit uPSI_JvHtmlParser;                   //JCL
122 unit uPSI_JvgXMLSerializer;               //JCL
123 unit uPSI_JvJCLUtils;                     //JCL
124 unit uPSI_JvStrings;                      //JCL
125 unit uPSI_uTPLb_IntegerUtils;             //TurboPower
126 unit uPSI_uTPLb_HugeCardinal;             //TurboPower
127 unit uPSI_uTPLb_HugeCardinalUtils;        //TurboPower
128 unit uPSI_SynRegExpr;                     //SynEdit
129 unit uPSI_StUtils;                        //SysTools4
130 unit uPSI_StToHTML;                       //SysTools4
```

```
131 unit uPSI_StStrms;                         //SysTools4
132 unit uPSI_StFIN;                           //SysTools4
133 unit uPSI_StAstroP;                        //SysTools4
134 unit uPSI_StStat;                          //SysTools4
135 unit uPSI_StNetCon;                        //SysTools4
136 unit uPSI_StDecMth;                        //SysTools4
137 unit uPSI_StOStr;                          //SysTools4
138 unit uPSI_StPtrns;                         //SysTools4
139 unit uPSI_StNetMsg;                        //SysTools4
140 unit uPSI_StMath;                          //SysTools4
141 unit uPSI_StExpEng;                        //SysTools4
142 unit uPSI_StCRC;                           //SysTools4
143 unit uPSI_StExport,                        //SysTools4
144 unit uPSI_StExpLog,                        //SysTools4
145 unit uPSI_ActnList;                        //Delphi VCL
146 unit uPSI_jpeg;                            //Borland
147 unit uPSI_StRandom;                        //SysTools4
148 unit uPSI_StDict;                          //SysTools4
149 unit uPSI_StBCD;                           //SysTools4
150 unit uPSI_StTxtDat;                        //SysTools4
151 unit uPSI_StRegEx;                         //SysTools4
152 unit uPSI_IMouse;                          //VCL
153 unit uPSI_SyncObjs;                        //VCL
154 unit uPSI_AsyncCalls;                      //Hausladen
155 unit uPSI_ParallelJobs;                    //Saraiva
156 unit uPSI_Variants;                        //VCL
157 unit uPSI_VarCmplx;                        //VCL Wolfram
158 unit uPSI_DTDSchema;                       //VCL
159 unit uPSI_ShLwApi;                         //Brakel
160 unit uPSI_IBUtils;                         //VCL
161 unit uPSI_CheckLst;                        //VCL
162 unit uPSI_JvSimpleXml;                     //JCL
163 unit uPSI_JclSimpleXml;                    //JCL
164 unit uPSI_JvXmlDatabase;                   //JCL
165 unit uPSI_JvMaxPixel;                      //JCL
166 unit uPSI_JvItemsSearchs;                  //JCL
167 unit uPSI_StExpEng2;                       //SysTools4
168 unit uPSI_StGenLog;                        //SysTools4
169 unit uPSI_JvLogFile;                       //Jcl
170 unit uPSI_CPort;                           //ComPort Lib v4.11
171 unit uPSI_CPortCtl;                        //ComPort
172 unit uPSI_CPortEsc;                        //ComPort
173 unit BarCodeScaner;                        //ComPort
174 unit uPSI_JvGraph;                         //JCL
175 unit uPSI_JvComCtrls;                      //JCL
176 unit uPSI_GUITesting;                      //D Unit
177 unit uPSI_JvFindFiles;                     //JCL
178 unit uPSI_StSystem;                        //SysTools4
179 unit uPSI_JvKeyboardStates;                //JCL
180 unit uPSI_JvMail;                          //JCL
181 unit uPSI_JclConsole;                      //JCL
182 unit uPSI_JclLANMan;                       //JCL
183 unit uPSI_IdCustomHTTPServer;              //Indy
184 unit IdHTTPServer                          //Indy
185 unit uPSI_IdTCPServer;                     //Indy
186 unit uPSI_IdSocketHandle;                  //Indy
187 unit uPSI_IdIOHandlerSocket;               //Indy
188 unit IdIOHandler;                          //Indy
189 unit uPSI_cutils;                          //Bloodshed
190 unit uPSI_BoldUtils;                       //boldsoft
191 unit uPSI_IdSimpleServer;                  //Indy
192 unit uPSI_IdSSLOpenSSL;                    //Indy
193 unit uPSI_IdMultipartFormData;             //Indy
194 unit uPSI_SynURIOpener;                    //SynEdit
195 unit uPSI_PerlRegEx;                       //PCRE
196 unit uPSI_IdHeaderList;                    //Indy
197 unit uPSI_StFirst;                         //SysTools4
198 unit uPSI_JvCtrls;                         //JCL
199 unit uPSI_IdTrivialFTPBase;                //Indy
200 unit uPSI_IdTrivialFTP;                    //Indy
201 unit uPSI_IdUDPBase;                       //Indy
202 unit uPSI_IdUDPClient;                     //Indy
203 unit uPSI_utypes;                          //for DMath.DLL
204 unit uPSI_ShellAPI;                        //Borland
205 unit uPSI_IdRemoteCMDClient;               //Indy
206 unit uPSI_IdRemoteCMDServer;               //Indy
207 unit IdRexecServer;                        //Indy
208 unit IdRexec; (unit uPSI_IdRexec;)         //Indy
209 unit IdUDPServer;                          //Indy
210 unit IdTimeUDPServer;                      //Indy
211 unit IdTimeServer;                         //Indy
212 unit IdTimeUDP; (unit uPSI_IdUDPServer;)    //Indy
213 unit uPSI_IdIPWatch;                       //Indy
214 unit uPSI_IdIrcServer;                     //Indy
215 unit uPSI_IdMessageCollection;             //Indy
216 unit uPSI_cPEM;                            //Fundamentals 4
217 unit uPSI_cFundamentUtils;                 //Fundamentals 4
218 unit uPSI_uwinplot;                        //DMath
219 unit uPSI_xrtl_util_CPUUtils;              //ExtentedRTL
```

```
220 unit uPSI_GR32_System;                         //Graphics32
221 unit uPSI_cFileUtils;                          //Fundamentals 4
222 unit uPSI_cDateTime; (timemachine)             //Fundamentals 4
223 unit uPSI_cTimers; (high precision timer)      //Fundamentals 4
224 unit uPSI_cRandom;                             //Fundamentals 4
225 unit uPSI_ueval;                               //DMath
226 unit uPSI_xrtl_net_URIUtils;                   //ExtendedRTL
227 unit xrtl_net_URIUtils;                        //ExtendedRTL
228 unit uPSI_ufft;  (FFT)                         //DMath
229 unit uPSI_DBXChannel;                          //Delphi
230 unit uPSI_DBXIndyChannel;                      //Delphi Indy
231 unit uPSI_xrtl_util_COMCat;                    //ExtendedRTL
232 unit uPSI_xrtl_util_StrUtils;                  //ExtendedRTL
233 unit uPSI_xrtl_util_VariantUtils;              //ExtendedRTL
234 unit uPSI_xrtl_util_FileUtils;                 //ExtendedRTL
235 unit xrtl_util_Compat;                         //ExtendedRTL
236 unit uPSI_OleAuto;                             //Borland
237 unit uPSI_xrtl_util_COMUtils;                  //ExtendedRTL
238 unit uPSI_CmAdmCtl;                            //Borland
239 unit uPSI_ValEdit2;                            //VCL
240 unit uPSI_GR32;  //Graphics32                  //Graphics32
241 unit uPSI_GR32_Image;                          //Graphics32
242 unit uPSI_xrtl_util_TimeUtils;                 //ExtendedRTL
243 unit uPSI_xrtl_util_TimeZone;                  //ExtendedRTL
244 unit uPSI_xrtl_util_TimeStamp;                 //ExtendedRTL
245 unit uPSI_xrtl_util_Map;                       //ExtendedRTL
246 unit uPSI_xrtl_util_Set;                       //ExtendedRTL
247 unit uPSI_CPortMonitor;                        //ComPort
248 unit uPSI_StIniStm;                            //SysTools4
249 unit uPSI_GR32_ExtImage;                       //Graphics32
250 unit uPSI_GR32_OrdinalMaps;                    //Graphics32
251 unit uPSI_GR32_Rasterizers;                    //Graphics32
252 unit uPSI_xrtl_util_Exception;                 //ExtendedRTL
253 unit uPSI_xrtl_util_Value;                     //ExtendedRTL
254 unit uPSI_xrtl_util_Compare;                   //ExtendedRTL
255 unit uPSI_FlatSB;                              //VCL
256 unit uPSI_JvAnalogClock;                       //JCL
257 unit uPSI_JvAlarms;                            //JCL
258 unit uPSI_JvSQLS;                              //JCL
259 unit uPSI_JvDBSecur;                           //JCL
260 unit uPSI_JvDBQBE;                             //JCL
261 unit uPSI_JvStarfield;                         //JCL
262 unit uPSI_JVCLMiscal;                          //JCL
263 unit uPSI_JvProfiler32;                        //JCL
264 unit uPSI_JvDirectories,                       //JCL
265 unit uPSI_JclSchedule,                         //JCL
266 unit uPSI_JclSvcCtrl,                          //JCL
267 unit uPSI_JvSoundControl,                      //JCL
268 unit uPSI_JvBDESQLScript,                      //JCL
269 unit uPSI_JvgDigits,                           //JCL>
270 unit uPSI_ImgList;                             //TCustomImageList
271 unit uPSI_JclMIDI;                             //JCL>
272 unit uPSI_JclWinMidi;                          //JCL>
273 unit uPSI_JclNTFS;                             //JCL>
274 unit uPSI_JclAppInst;                          //JCL>
275 unit uPSI_JvRle;                               //JCL>
276 unit uPSI_JvRas32;                             //JCL>
277 unit uPSI_JvImageDrawThread,                   //JCL>
278 unit uPSI_JvImageWindow,                       //JCL>
279 unit uPSI_JvTransparentForm;                   //JCL>
280 unit uPSI_JvWinDialogs;                        //JCL>
281 unit uPSI_JvSimLogic,                          //JCL>
282 unit uPSI_JvSimIndicator,                      //JCL>
283 unit uPSI_JvSimPID,                            //JCL>
284 unit uPSI_JvSimPIDLinker,                      //JCL>
285 unit uPSI_IdRFCReply;                          //Indy
286 unit uPSI_IdIdent;                             //Indy
287 unit uPSI_IdIdentServer;                       //Indy
288 unit uPSI_JvPatchFile;                         //JCL
289 unit uPSI_StNetPfm;                            //SysTools4
290 unit uPSI_StNet;                               //SysTools4
291 unit uPSI_JclPeImage;                          //JCL
292 unit uPSI_JclPrint;                            //JCL
293 unit uPSI_JclMime;                             //JCL
294 unit uPSI_JvRichEdit;                          //JCL
295 unit uPSI_JvDBRichEd;                          //JCL
296 unit uPSI_JvDice;                              //JCL
297 unit uPSI_JvFloatEdit;                         //JCL 3.9.8
298 unit uPSI_JvDirFrm;                            //JCL
299 unit uPSI_JvDualList;                          //JCL
300 unit uPSI_JvSwitch;                            ////JCL
301 unit uPSI_JvTimerLst;                          ////JCL
302 unit uPSI_JvMemTable;                          //JCL
303 unit uPSI_JvObjStr;                            //JCL
304 unit uPSI_StLArr;                              //SysTools4
305 unit uPSI_StWmDCpy;                            //SysTools4
306 unit uPSI_StText;                              //SysTools4
307 unit uPSI_StNTLog;                             //SysTools4
308 unit uPSI_xrtl_math_Integer;                   //ExtendedRTL
```

```
309 unit uPSI_JvImagPrvw;                      //JCL
310 unit uPSI_JvFormPatch;                     //JCL
311 unit uPSI_JvPicClip;                       //JCL
312 unit uPSI_JvDataConv;                      //JCL
313 unit uPSI_JvCpuUsage;                      //JCL
314 unit uPSI_JclUnitConv_mX2;                 //JCL
315 unit JvDualListForm;                       //JCL
316 unit uPSI_JvCpuUsage2;                     //JCL
317 unit uPSI_JvParserForm;                    //JCL
318 unit uPSI_JvJanTreeView;                   //JCL
319 unit uPSI_JvTransLED;                      //JCL
320 unit uPSI_JvPlaylist;                      //JCL
321 unit uPSI_JvFormAutoSize;                  //JCL
322 unit uPSI_JvYearGridEditForm;              //JCL
323 unit uPSI_JvMarkupCommon;                  //JCL
324 unit uPSI_JvChart;                         //JCL
325 unit uPSI_JvXPCore;                        //JCL
326 unit uPSI_JvXPCoreUtils;                   //JCL
327 unit uPSI_StatsClasses;                    //mX4
328 unit uPSI_ExtCtrls2;                       //VCL
329 unit uPSI_JvUrlGrabbers;                   //JCL
330 unit uPSI_JvXmlTree;                       //JCL
331 unit uPSI_JvWavePlayer;                    //JCL
332 unit uPSI_JvUnicodeCanvas;                 //JCL
333 unit uPSI_JvTFUtils;                       //JCL
334 unit uPSI_IdServerIOHandler;               //Indy
335 unit uPSI_IdServerIOHandlerSocket;         //Indy
336 unit uPSI_IdMessageCoder;                  //Indy
337 unit uPSI_IdMessageCoderMIME;              //Indy
338 unit uPSI_IdMIMETypes;                     //Indy
339 unit uPSI_JvConverter;                     //JCL
340 unit uPSI_JvCsvParse;                      //JCL
341 unit uPSI_umath;  unit uPSI_ugamma;        //DMath
342 unit uPSI_ExcelExport;(Nat:TJsExcelExport) //JCL
343 unit uPSI_JvDBGridExport;                  //JCL
344 unit uPSI_JvgExport;                       //JCL
345 unit uPSI_JvSerialMaker;                   //JCL
346 unit uPSI_JvWin32;                         //JCL
347 unit uPSI_JvPaintFX;                       //JCL
348 unit uPSI_JvOracleDataSet; (beta)          //JCL
349 unit uPSI_JvValidators; (preview)          //JCL
350 unit uPSI_JvNTEventLog;                    //JCL
351 unit uPSI_ShellZipTool;                    //mX4
352 unit uPSI_JvJoystick;                      //JCL
353 unit uPSI_JvMailSlots;                     //JCL
354 unit uPSI_JclComplex;                      //JCL
355 unit uPSI_SynPdf;                          //Synopse
356 unit uPSI_Registry;                        //VCL
357 unit uPSI_TlHelp32;                        //VCL
358 unit uPSI_JclRegistry;                     //JCL
359 unit uPSI_JvAirBrush;                      //JCL
360 unit uPSI_mORMotReport;                    //Synopse
361 unit uPSI_JclLocales;                      //JCL
362 unit uPSI_SynEdit;                         //SynEdit
363 unit uPSI_SynEditTypes;                    //SynEdit
364 unit uPSI_SynMacroRecorder;                //SynEdit
365 unit uPSI_LongIntList;                     //SynEdit
366 unit uPSI_devcutils;                       //DevC
367 unit uPSI_SynEditMiscClasses;              //SynEdit
368 unit uPSI_SynEditRegexSearch;              //SynEdit
369 unit uPSI_SynEditHighlighter;              //SynEdit
370 unit uPSI_SynHighlighterPas;               //SynEdit
371 unit uPSI_JvSearchFiles;                   //JCL
372 unit uPSI_SynHighlighterAny;               //Lazarus
373 unit uPSI_SynEditKeyCmds;                  //SynEdit
374 unit uPSI_SynEditMiscProcs,                //SynEdit
375 unit uPSI_SynEditKbdHandler                //SynEdit
376 unit uPSI_JvAppInst,                       //JCL
377 unit uPSI_JvAppEvent;                      //JCL
378 unit uPSI_JvAppCommand;                    //JCL
379 unit uPSI_JvAnimTitle;                     //JCL
380 unit uPSI_JvAnimatedImage;                 //JCL
381 unit uPSI_SynEditExport;                   //SynEdit
382 unit uPSI_SynExportHTML;                   //SynEdit
383 unit uPSI_SynExportRTF;                    //SynEdit
384 unit uPSI_SynEditSearch;                   //SynEdit
385 unit uPSI_fMain_back                       //maXbox;
386 unit uPSI_JvZoom;                          //JCL
387 unit uPSI_PMrand;                          //PM
388 unit uPSI_JvSticker;                       //JCL
389 unit uPSI_XmlVerySimple;                   //mX4
390 unit uPSI_Services;                        //ExtPascal
391 unit uPSI_ExtPascalUtils;                  //ExtPascal
392 unit uPSI_SocketsDelphi;                   //ExtPascal
393 unit uPSI_StBarC;                          //SysTools
394 unit uPSI_StDbBarC;                        //SysTools
395 unit uPSI_StBarPN;                         //SysTools
396 unit uPSI_StDbPNBC;                        //SysTools
397 unit uPSI_StDb2DBC;                        //SysTools
```

```
398 unit uPSI_StMoney;                         //SysTools
399 unit uPSI_JvForth;                         //JCL
400 unit uPSI_RestRequest;                     //mX4
401 unit uPSI_HttpRESTConnectionIndy;          //mX4
402 unit uPSI_JvXmlDatabase; //update          //JCL
403 unit uPSI_StAstro;                         //SysTools
404 unit uPSI_StSort;                          //SysTools
405 unit uPSI_StDate;                          //SysTools
406 unit uPSI_StDateSt;                        //SysTools
407 unit uPSI_StBase;                          //SysTools
408 unit uPSI_StVInfo;                         //SysTools
409 unit uPSI_JvBrowseFolder;                  //JCL
410 unit uPSI_JvBoxProcs;                      //JCL
411 unit uPSI_urandom; (unit uranuvag;)        //DMath
412 unit uPSI_usimann; (unit ugenalg;)         //DMath
413 unit uPSI_JvHighlighter;                   //JCL
414 unit uPSI_Diff;                            //mX4
415 unit uPSI_SpringWinAPI;                    //DSpring
416 unit uPSI_StBits;                          //SysTools
417 unit uPSI_TomDBQue;                        //mX4
418 unit uPSI_MultilangTranslator;             //mX4
419 unit uPSI_HyperLabel;                      //mX4
420 unit uPSI_Starter;                         //mX4
421 unit uPSI_FileAssocs;                      //devC
422 unit uPSI_devFileMonitorX;                 //devC
423 unit uPSI_devrun;                          //devC
424 unit uPSI_devExec;                         //devC
425 unit uPSI_oysUtils;                        //devC
426 unit uPSI_DosCommand;                      //devC
427 unit uPSI_CppTokenizer;                    //devC
428 unit uPSI_JvHLParser;                      //devC
429 unit uPSI_JclMapi;                         //JCL
430 unit uPSI_JclShell;                        //JCL
431 unit uPSI_JclCOM;                          //JCL
432 unit uPSI_GR32_Math;                       //Graphics32
433 unit uPSI_GR32_LowLevel;                   //Graphics32
434 unit uPSI_SimpleHl;                        //mX4
435 unit uPSI_GR32_Filters,                    //Graphics32
436 unit uPSI_GR32_VectorMaps;                 //Graphics32
437 unit uPSI_cXMLFunctions;                   //Fundamentals 4
438 unit uPSI_JvTimer;                         //JCL
439 unit uPSI_cHTTPUtils;                      //Fundamentals 4
440 unit uPSI_cTLSUtils;                       //Fundamentals 4
441 unit uPSI_JclGraphics;                     //JCL
442 unit uPSI_JclSynch;                        //JCL
443 unit uPSI_IdTelnet;                        //Indy
444 unit uPSI_IdTelnetServer,                  //Indy
445 unit uPSI_IdEcho,                          //Indy
446 unit uPSI_IdEchoServer,                    //Indy
447 unit uPSI_IdEchoUDP,                       //Indy
448 unit uPSI_IdEchoUDPServer,                 //Indy
449 unit uPSI_IdSocks,                         //Indy
450 unit uPSI_IdAntiFreezeBase;                //Indy
451 unit uPSI_IdHostnameServer;                //Indy
452 unit uPSI_IdTunnelCommon,                  //Indy
453 unit uPSI_IdTunnelMaster,                  //Indy
454 unit uPSI_IdTunnelSlave,                   //Indy
455 unit uPSI_IdRSH,                           //Indy
456 unit uPSI_IdRSHServer,                     //Indy
457 unit uPSI_Spring_Cryptography_Utils;       //Spring4Delphi
458 unit uPSI_MapReader,                       //devC
459 unit uPSI_LibTar,                          //devC
460 unit uPSI_IdStack;                         //Indy
461 unit uPSI_IdBlockCipherIntercept;          //Indy
462 unit uPSI_IdChargenServer;                 //Indy
463 unit uPSI_IdFTPServer,                     //Indy
464 unit uPSI_IdException,                     //Indy
465 unit uPSI_utexplot;                        //DMath
466 unit uPSI_uwinstr;                         //DMath
467 unit uPSI_VarRecUtils;                     //devC
468 unit uPSI_JvStringListToHtml,              //JCL
469 unit uPSI_JvStringHolder,                  //JCL
470 unit uPSI_IdCoder;                         //Indy
471 unit uPSI_SynHighlighterDfm;               //Synedit
472 unit uHighlighterProcs; in 471             //Synedit
473 unit uPSI_LazFileUtils,                    //LCL
474 unit uPSI_IDECmdLine;                      //LCL
475 unit uPSI_lazMasks;                        //LCL
476 unit uPSI_ip_misc;                         //mX4
477 unit uPSI_Barcode;                         //LCL
478 unit uPSI_SimpleXML;                       //LCL
479 unit uPSI_JclIniFiles;                     //JCL
480 unit uPSI_D2XXUnit;  {$X-}                 //FTDI
481 unit uPSI_JclDateTime;                     //JCL
482 unit uPSI_JclEDI;                          //JCL
483 unit uPSI_JclMiscel2;                      //JCL
484 unit uPSI_JclValidation;                   //JCL
485 unit uPSI_JclAnsiStrings; {-PString}       //JCL
486 unit uPSI_SynEditMiscProcs2;               //Synedit
```

```
487 unit uPSI_JclStreams;                     //JCL
488 unit uPSI_QRCode;                         //mX4
489 unit uPSI_BlockSocket;                    //ExtPascal
490 unit uPSI_Masks,Utils                     //VCL
491 unit uPSI_synautil;                       //Synapse!
492 unit uPSI_JclMath_Class;                  //JCL RTL
493 unit ugamdist; //Gamma function           //DMath
494 unit uibeta, ucorrel; //IBeta             //DMath
495 unit uPSI_SRMgr;                          //mX4
496 unit uPSI_HotLog;                         //mX4
497 unit uPSI_DebugBox;                       //mX4
498 unit uPSI_ustrings;                       //DMath
499 unit uPSI_uregtest;                       //DMath
500 unit uPSI_usimplex;                       //DMath
501 unit uPSI_uhyper;                         //DMath
502 unit uPSI_IdHL7;                          //Indy
503 unit uPSI_IdIPMCastBase,                  //Indy
504 unit uPSI_IdIPMCastServer;                //Indy
505 unit uPSI_IdIPMCastClient;                //Indy
506 unit uPSI_unlfit; //nlregression          //DMath
507 unit uPSI_IdRawHeaders;                   //Indy
508 unit uPSI_IdRawClient;                    //Indy
509 unit uPSI_IdRawFunctions;                 //Indy
510 unit uPSI_IdTCPStream;                    //Indy
511 unit uPSI_IdSNPP;                         //Indy
512 unit uPSI_St2DBarC;                       //SysTools
513 unit uPSI_ImageWin;  //FTL                //VCL
514 unit uPSI_CustomDrawTreeView; //FTL       //VCL
515 unit uPSI_GraphWin;  //FTL                //VCL
516 unit uPSI_actionMain;  //FTL              //VCL
517 unit uPSI_StSpawn;                        //SysTools
518 unit uPSI_CtlPanel;                       //VCL
519 unit uPSI_IdLPR;                          //Indy
520 unit uPSI_SockRequestInterpreter;         //Indy
521 unit uPSI_ulambert;                       //DMath
522 unit uPSI_ucholesk;                       //DMath
523 unit uPSI_SimpleDS;                       //VCL
524 unit uPSI_DBXSqlScanner;                  //VCL
525 unit uPSI_DBXMetaDataUtil;                //VCL
526 unit uPSI_Chart;                          //TEE
527 unit uPSI_TeeProcs;                       //TEE
528 unit mXBDEUtils;                          //mX4
529 unit uPSI_MDIEdit;                        //VCL
530 unit uPSI_CopyPrsr;                       //VCL
531 unit uPSI_SockApp;                        //VCL
532 unit uPSI_AppEvnts;                       //VCL
533 unit uPSI_ExtActns;                       //VCL
534 unit uPSI_TeEngine;                       //TEE
535 unit uPSI_CoolMain; //browser             //VCL
536 unit uPSI_StCRC;                          //SysTools
537 unit uPSI_StDecMth2;                      //SysTools
538 unit uPSI_frmExportMain;                  //Synedit
539 unit uPSI_SynDBEdit;                      //Synedit
540 unit uPSI_SynEditWildcardSearch;          //Synedit
541 unit uPSI_BoldComUtils;                   //BOLD
542 unit uPSI_BoldIsoDateTime;                //BOLD
543 unit uPSI_BoldGUIDUtils; //inCOMUtils     //BOLD
544 unit uPSI_BoldXMLRequests;                //BOLD
545 unit uPSI_BoldStringList;                 //BOLD
546 unit uPSI_BoldFileHandler;                //BOLD
547 unit uPSI_BoldContainers;                 //BOLD
548 unit uPSI_BoldQueryUserDlg;               //BOLD
549 unit uPSI_BoldWinINet;                    //BOLD
550 unit uPSI_BoldQueue;                      //BOLD
551 unit uPSI_JvPcx;                          //JCL
552 unit uPSI_IdWhois;                        //Indy
553 unit uPSI_IdWhoIsServer;                  //Indy
554 unit uPSI_IdGopher;                       //Indy
555 unit uPSI_IdDateTimeStamp;                //Indy
556 unit uPSI_IdDayTimeServer;                //Indy
557 unit uPSI_IdDayTimeUDP;                   //Indy
558 unit uPSI_IdDayTimeUDPServer;             //Indy
559 unit uPSI_IdDICTServer;                   //Indy
560 unit uPSI_IdDiscardServer;                //Indy
561 unit uPSI_IdDiscardUDPServer;             //Indy
562 unit uPSI_IdMappedFTP;                    //Indy
563 unit uPSI_IdMappedPortTCP;                //Indy
564 unit uPSI_IdGopherServer;                 //Indy
565 unit uPSI_IdQotdServer;                   //Indy
566 unit uPSI_JvRgbToHtml;                    //JCL
567 unit uPSI_JvRemLog,                       //JCL
568 unit uPSI_JvSysComp;                      //JCL
569 unit uPSI_JvTMTL;                         //JCL
570 unit uPSI_JvWinampAPI;                    //JCL
571 unit uPSI_MSysUtils;                      //mX4
572 unit uPSI_ESBMaths;                       //ESB
573 unit uPSI_ESBMaths2;                      //ESB
574 unit uPSI_uLkJSON;                        //Lk
575 unit uPSI_ZURL;  //Zeos                   //Zeos
```

```
576 unit uPSI_ZSysUtils;                       //Zeos
577 unit unaUtils internals                    //UNA
578 unit uPSI_ZMatchPattern;                   //Zeos
579 unit uPSI_ZClasses;                        //Zeos
580 unit uPSI_ZCollections;                    //Zeos
581 unit uPSI_ZEncoding;                       //Zeos
582 unit uPSI_IdRawBase;                       //Indy
583 unit uPSI_IdNTLM;                          //Indy
584 unit uPSI_IdNNTP;                          //Indy
585 unit uPSI_usniffer; //PortScanForm         //mX4
586 unit uPSI_IdCoderMIME;                     //Indy
587 unit uPSI_IdCoderUUE;                      //Indy
588 unit uPSI_IdCoderXXE;                      //Indy
589 unit uPSI_IdCoder3to4;                     //Indy
590 unit uPSI_IdCookie;                        //Indy
591 unit uPSI_IdCookieManager;                 //Indy
592 unit uPSI_WDosSocketUtils;                 //WDos
593 unit uPSI_WDosPlcUtils;                    //WDos
594 unit uPSI_WDosPorts;                       //WDos
595 unit uPSI_WDosResolvers;                   //WDos
596 unit uPSI_WDosTimers;                      //WDos
597 unit uPSI_WDosPlcs;                        //WDos
598 unit uPSI_WDosPneumatics;                  //WDos
599 unit uPSI_IdFingerServer;                  //Indy
600 unit uPSI_IdDNSResolver;                   //Indy
601 unit uPSI_IdHTTPWebBrokerBridge;           //Indy
602 unit uPSI_IdIntercept;                     //Indy
603 unit uPSI_IdIPMCastBase;                   //Indy
604 unit uPSI_IdLogBase;                       //Indy
605 unit uPSI_IdIOHandlerStream;               //Indy
606 unit uPSI_IdMappedPortUDP;                 //Indy
607 unit uPSI_IdQOTDUDPServer;                 //Indy
608 unit uPSI_IdQOTDUDP;                       //Indy
609 unit uPSI_IdSysLog;                        //Indy
610 unit uPSI_IdSysLogServer;                  //Indy
611 unit uPSI_IdSysLogMessage;                 //Indy
612 unit uPSI_IdTimeServer;                    //Indy
613 unit uPSI_IdTimeUDP;                       //Indy
614 unit uPSI_IdTimeUDPServer;                 //Indy
615 unit uPSI_IdUserAccounts;                  //Indy
616 unit uPSI_TextUtils;                       //mX4
617 unit uPSI_MandelbrotEngine;                //mX4
618 unit uPSI_delphi_arduino_Unit1;            //mX4
619 unit uPSI_DTDSchema2;                      //mX4
620 unit uPSI_fplotMain;                       //DMath
621 unit uPSI_FindFileIter;                    //mX4
622 unit uPSI_PppState;  (JclStrHashMap)       //PPP
623 unit uPSI_PppParser;                       //PPP
624 unit uPSI_PppLexer;                        //PPP
625 unit uPSI_PCharUtils;                      //PPP
626 unit uPSI_uJSON;                           //WU
627 unit uPSI_JclStrHashMap;                   //JCL
628 unit uPSI_JclHookExcept;                   //JCL
629 unit uPSI_EncdDecd;                        //VCL
630 unit uPSI_SockAppReg;                      //VCL
631 unit uPSI_PJFileHandle;                    //PJ
632 unit uPSI_PJEnvVars;                       //PJ
633 unit uPSI_PJPipe;                          //PJ
634 unit uPSI_PJPipeFilters;                   //PJ
635 unit uPSI_PJConsoleApp;                    //PJ
636 unit uPSI_UConsoleAppEx;                   //PJ
637 unit uPSI_DbxSocketChannelNative,          //VCL
638 unit uPSI_DbxDataGenerator,                //VCL
639 unit uPSI_DBXClient;                       //VCL
640 unit uPSI_IdLogEvent;                      //Indy
641 unit uPSI_Reversi;                         //mX4
642 unit uPSI_Geometry;                        //mX4
643 unit uPSI_IdSMTPServer;                    //Indy
644 unit uPSI_Textures;                        //mX4
645 unit uPSI_IBX;                             //VCL
646 unit uPSI_IWDBCommon;                      //VCL
647 unit uPSI_SortGrid;                        //mX4
648 unit uPSI_IB;                              //VCL
649 unit uPSI_IBScript;                        //VCL
650 unit uPSI_JvCSVBaseControls;               //JCL
651 unit uPSI_Jvg3DColors;                     //JCL
652 unit uPSI_JvHLEditor;  //beat              //JCL
653 unit uPSI_JvShellHook;                     //JCL
654 unit uPSI_DBCommon2                        //VCL
655 unit uPSI_JvSHFileOperation;               //JCL
656 unit uPSI_uFilexport;                      //mX4
657 unit uPSI_JvDialogs;                       //JCL
658 unit uPSI_JvDBTreeView;                    //JCL
659 unit uPSI_JvDBUltimGrid;                   //JCL
660 unit uPSI_JvDBQueryParamsForm;             //JCL
661 unit uPSI_JvExControls;                    //JCL
662 unit uPSI_JvBDEMemTable;                   //JCL
663 unit uPSI_JvCommStatus;                    //JCL
664 unit uPSI_JvMailSlots2;                    //JCL
```

```
665 unit uPSI_JvgWinMask;                      //JCL
666 unit uPSI_StEclpse;                        //SysTools
667 unit uPSI_StMime;                          //SysTools
668 unit uPSI_StList;                          //SysTools
669 unit uPSI_StMerge;                         //SysTools
670 unit uPSI_StStrS;                          //SysTools
671 unit uPSI_StTree,                          //SysTools
672 unit uPSI_StVArr;                          //SysTools
673 unit uPSI_StRegIni;                        //SysTools
674 unit uPSI_urkf;                            //DMath
675 unit uPSI_usvd;                            //DMath
676 unit uPSI_DepWalkUtils;                    //JCL
677 unit uPSI_OptionsFrm;                      //JCL
678 unit yuvconverts;                          //mX4
679 uPSI_JvPropAutoSave;                       //JCL
680 uPSI_AclAPI;                               //alcinoe
681 uPSI_AviCap;                               //alcinoe
682 uPSI_ALAVLBinaryTree;                      //alcinoe
683 uPSI_ALFcnMisc;                            //alcinoe
684 uPSI_ALStringList;                         //alcinoe
685 uPSI_ALQuickSortList;                      //alcinoe
686 uPSI_ALStaticText;                         //alcinoe
687 uPSI_ALJSONDoc;                            //alcinoe
688 uPSI_ALGSMComm;                            //alcinoe
689 uPSI_ALWindows;                            //alcinoe
690 uPSI_ALMultiPartFormDataParser;            //alcinoe
691 uPSI_ALHttpCommon;                         //alcinoe
692 uPSI_ALWebSpider,                          //alcinoe
693 uPSI_ALHttpClient;                         //alcinoe
694 uPSI_ALFcnHTML;                            //alcinoe
695 uPSI_ALFTPClient;                          //alcinoe
696 uPSI_ALInternetMessageCommon;              //alcinoe
697 uPSI_ALWininetHttpClient;                  //alcinoe
698 uPSI_ALWinInetFTPClient;                   //alcinoe
699 uPSI_ALWinHttpWrapper;                     //alcinoe
700 uPSI_ALWinHttpClient;                      //alcinoe
701 uPSI_ALFcnWinSock;                         //alcinoe
702 uPSI_ALFcnSQL;                             //alcinoe
703 uPSI_ALFcnCGI;                             //alcinoe
704 uPSI_ALFcnExecute;                         //alcinoe
705 uPSI_ALFcnFile;                            //alcinoe
706 uPSI_ALFcnMime;                            //alcinoe
707 uPSI_ALPhpRunner;                          //alcinoe
708 uPSI_ALGraphic;                            //alcinoe
709 uPSI_ALIniFiles;                           //alcinoe
710 uPSI_ALMemCachedClient;                    //alcinoe
711 unit uPSI_MyGrids;                         //mX4
712 uPSI_ALMultiPartMixedParser                //alcinoe
713 uPSI_ALSMTPClient                          //alcinoe
714 uPSI_ALNNTPClient;                         //alcinoe
715 uPSI_ALHintBalloon;                        //alcinoe
716 unit uPSI_ALXmlDoc;                        //alcinoe
717 unit uPSI_IPCThrd;                         //VCL
718 unit uPSI_MonForm;                         //VCL
719 unit uPSI_TeCanvas;                        //Orpheus
720 unit uPSI_Ovcmisc;                         //Orpheus
721 unit uPSI_ovcfiler;                        //Orpheus
722 unit uPSI_ovcstate;                        //Orpheus
723 unit uPSI_ovccoco;                         //Orpheus
724 unit uPSI_ovcrvexp;                        //Orpheus
725 unit uPSI_OvcFormatSettings;               //Orpheus
726 unit uPSI_OvcUtils;                        //Orpheus
727 unit uPSI_ovcstore;                        //Orpheus
728 unit uPSI_ovcstr;                          //Orpheus
729 unit uPSI_ovcmru;                          //Orpheus
730 unit uPSI_ovccmd;                          //Orpheus
731 unit uPSI_ovctimer;                        //Orpheus
732 unit uPSI_ovcintl;                         //Orpheus
733 uPSI_AfCircularBuffer;                     //AsyncFree
734 uPSI_AfUtils;                              //AsyncFree
735 uPSI_AfSafeSync;                           //AsyncFree
736 uPSI_AfComPortCore;                        //AsyncFree
737 uPSI_AfComPort;                            //AsyncFree
738 uPSI_AfPortControls;                       //AsyncFree
739 uPSI_AfDataDispatcher;                     //AsyncFree
740 uPSI_AfViewers;                            //AsyncFree
741 uPSI_AfDataTerminal;                       //AsyncFree
742 uPSI_SimplePortMain;                       //AsyncFree
743 unit uPSI_ovcclock;                        //Orpheus
744 unit uPSI_o32intlst;                       //Orpheus
745 unit uPSI_o32ledlabel;                     //Orpheus
746 unit uPSI_AlMySqlClient;                   //alcinoe
747 unit uPSI_ALFBXClient;                     //alcinoe
748 unit uPSI_ALFcnSQL;                        //alcinoe
749 unit uPSI_AsyncTimer;                      //mX4
750 unit uPSI_ApplicationFileIO;               //mX4
751 unit uPSI_PsAPI;                           //VCLé
752 uPSI_ovcuser;                              //Orpheus
753 uPSI_ovcurl;                               //Orpheus
```

```
754 uPSI_ovcvlb;                               //Orpheus
755 uPSI_ovccolor;                             //Orpheus
756 uPSI_ALFBXLib,                             //alcinoe
757 uPSI_ovcmeter;                             //Orpheus
758 uPSI_ovcpeakm;                             //Orpheus
759 uPSI_O32BGSty;                             //Orpheus
760 uPSI_ovcBidi;                              //Orpheus
761 uPSI_ovctcary;                             //Orpheus
762 uPSI_DXPUtils;                             //mX4
763 uPSI_ALMultiPartBaseParser;                //alcinoe
764 uPSI_ALMultiPartAlternativeParser;         //alcinoe
765 uPSI_ALPOP3Client;                         //alcinoe
766 uPSI_SmallUtils;                           //mX4
767 uPSI_MakeApp;                              //mX4
768 uPSI_O32MouseMon;                          //Orpheus
769 uPSI_OvcCache;                             //Orpheus
770 uPSI_ovccalc;                              //Orpheus
771 uPSI_Joystick,                             //OpenGL
772 uPSI_ScreenSaver;                          //OpenGL
773 uPSI_XCollection,                          //OpenGL
774 uPSI_Polynomials,                          //OpenGL
775 uPSI_PersistentClasses, //9.86             //OpenGL
776 uPSI_VectorLists;                          //OpenGL
777 uPSI_XOpenGL,                              //OpenGL
778 uPSI_MeshUtils;                            //OpenGL
779 unit uPSI_JclSysUtils;                     //JCL
780 unit uPSI_JclBorlandTools;                 //JCL
781 unit JclFileUtils_max;                     //JCL
782 uPSI_AfDataControls,                       //AsyncFree
783 uPSI_GLSilhouette;                         //OpenGL
784 uPSI_JclSysUtils_class;                    //JCL
785 uPSI_JclFileUtils_class;                   //JCL
786 uPSI_FileUtil;                             //JCL
787 uPSI_changefind;                           //mX4
788 uPSI_cmdIntf;                              //mX4
789 uPSI_fservice;                             //mX4
790 uPSI_Keyboard;                             //OpenGL
791 uPSI_VRMLParser,                           //OpenGL
792 uPSI_GLFileVRML,                           //OpenGL
793 uPSI_Octree;                               //OpenGL
794 uPSI_GLPolyhedron,                         //OpenGL
795 uPSI_GLCrossPlatform;                      //OpenGL
796 uPSI_GLParticles;                          //OpenGL
797 uPSI_GLNavigator;                          //OpenGL
798 uPSI_GLStarRecord;                         //OpenGL
799 uPSI_GLTextureCombiners;                   //OpenGL
800 uPSI_GLCanvas;                             //OpenGL
801 uPSI_GeometryBB;                           //OpenGL
802 uPSI_GeometryCoordinates;                  //OpenGL
803 uPSI_VectorGeometry;                       //OpenGL
804 uPSI_BumpMapping;                          //OpenGL
805 uPSI_TGA;                                  //OpenGL
806 uPSI_GLVectorFileObjects;                  //OpenGL
807 uPSI_IMM;                                  //VCL
808 uPSI_CategoryButtons;                      //VCL
809 uPSI_ButtonGroup;                          //VCL
810 uPSI_DbExcept;                             //VCL
811 uPSI_AxCtrls;                              //VCL
812 uPSI_GL_actorUnit1;                        //OpenGL
813 uPSI_StdVCL;                               //VCL
814 unit CurvesAndSurfaces;                    //OpenGL
815 uPSI_DataAwareMain;                        //AsyncFree
816 uPSI_TabNotBk;                             //VCL
817 uPSI_udwsfiler;                            //mX4
818 uPSI_synaip;                               //Synapse!
819 uPSI_synacode;                             //Synapse
820 uPSI_synachar;                             //Synapse
821 uPSI_synamisc;                             //Synapse
822 uPSI_synaser;                              //Synapse
823 uPSI_synaicnv;                             //Synapse
824 uPSI_tlntsend;                             //Synapse
825 uPSI_pingsend;                             //Synapse
826 uPSI_blcksock;                             //Synapse
827 uPSI_asn1util;                             //Synapse
828 uPSI_dnssend;                              //Synapse
829 uPSI_clamsend;                             //Synapse
830 uPSI_ldapsend;                             //Synapse
831 uPSI_mimemess;                             //Synapse
832 uPSI_slogsend;                             //Synapse
833 uPSI_mimepart;                             //Synapse
834 uPSI_mimeinln;                             //Synapse
835 uPSI_ftpsend,                              //Synapse
836 uPSI_ftptsend;                             //Synapse
837 uPSI_httpsend;                             //Synapse
838 uPSI_sntpsend;                             //Synapse
839 uPSI_smtpsend;                             //Synapse
840 uPSI_snmpsend;                             //Synapse
841 uPSI_imapsend;                             //Synapse
842 uPSI_pop3send;                             //Synapse
```

```
843 uPSI_nntpsend;                             //Synapse
844 uPSI_ssl_cryptlib;                         //Synapse
845 uPSI_ssl_openssl;                          //Synapse
846 uPSI_synhttp_daemon;                       //Synapse
847 uPSI_NetWork;                              //mX4
848 uPSI_PingThread;                           //Synapse
849 uPSI_JvThreadTimer;                        //JCL
850 unit uPSI_wwSystem;                        //InfoPower
851 unit uPSI_IdComponent;                     //Indy
852 unit uPSI_IdIOHandlerThrottle;             //Indy
853 unit uPSI_Themes;                          //VCL
854 unit uPSI_StdStyleActnCtrls;               //VCL
855 unit uPSI_UDDIHelper;                      //VCL
856 unit uPSI_IdIMAP4Server;                   //Indy
857 uPSI_VariantSymbolTable,                   //VCL //3.9.9.92
858 uPSI_udf_glob,                             //mX4
859 uPSI_TabGrid,                              //VCL
860 uPSI_JsDBTreeView,                         //mX4
861 uPSI_JsSendMail,                           //mX4
862 uPSI_dbTvRecordList,                       //mX4
863 uPSI_TreeVwEx,                             //mX4
864 uPSI_ECDataLink,                           //mX4
865 uPSI_dbTree,                               //mX4
866 uPSI_dbTreeCBox,                           //mX4
867 unit uPSI_Debug; //TfrmDebug               //mX4
868 uPSI_TimeFncs;                             //mX4
869 uPSI_FileIntf,                             //VCL
870 uPSI_SockTransport,                        //RTL
871 unit uPSI_WinInet;                         //RTL
872 unit uPSI_Wwstr;                           //mX4
873 uPSI_DBLookup,                             //VCL
874 uPSI_Hotspot,                              //mX4
875 uPSI_HList; //History List                 //mX4
876 unit uPSI_DrTable;                         //VCL
877 uPSI_TConnect,                             //VCL
978 uPSI_DataBkr,                              //VCL
979 uPSI_HTTPIntr;                             //VCL
980 unit uPSI_Mathbox;                         //mX4
881 uPSI_cyIndy,                               //cY
882 uPSI_cySysUtils,                           //cY
883 uPSI_cyWinUtils,                           //cY
884 uPSI_cyStrUtils,                           //cY
885 uPSI_cyObjUtils,                           //cY
886 uPSI_cyDateUtils,                          //cY
887 uPSI_cyBDE,                                //cY
888 uPSI_cyClasses,                            //cY
889 uPSI_cyGraphics,  //3.9.9.94_2             //cY
890 unit uPSI_cyTypes;                         //cY
891 uPSI_JvDateTimePicker,                     //JCL
892 uPSI_JvCreateProcess,                      //JCL
893 uPSI_JvEasterEgg,                          //JCL
894 uPSI_WinSvc,  //3.9.9.94_3                 //VCL
895 uPSI_SvcMgr                                //VCL
896 unit uPSI_JvPickDate;                      //JCL
897 unit uPSI_JvNotify;                        //JCL
898 uPSI_JvStrHlder                            //JCL
899 unit uPSI_JclNTFS2;                        //JCL
900 uPSI_Jcl8087 //math coprocessor            //JCL
901 uPSI_JvAddPrinter                          //JCL
902 uPSI_JvCabFile                             //JCL
903 uPSI_JvDataEmbedded;                       //JCL



////////////////////////////////////////////////////////////////////////////////
//Form Template Library FTL
////////////////////////////////////////////////////////////////////////////////

26 FTL For Form Building out of the Script, eg. 399_form_templates.txt

045 unit uPSI_VListView                        TFormListView;
263 unit uPSI_JvProfiler32;                    TProfReport
270 unit uPSI_ImgList;                         TCustomImageList
278 unit uPSI_JvImageWindow;                   TJvImageWindow
317 unit uPSI_JvParserForm;                    TJvHTMLParserForm
497 unit uPSI_DebugBox;                        TDebugBox
513 unit uPSI_ImageWin;                        TImageForm, TImageForm2
514 unit uPSI_CustomDrawTreeView;              TCustomDrawForm
515 unit uPSI_GraphWin;                        TGraphWinForm
516 unit uPSI_actionMain;                      TActionForm
518 unit uPSI_CtlPanel;                        TAppletApplication
529 unit uPSI_MDIEdit;                         TEditForm
535 unit uPSI_CoolMain; {browser}              TWebMainForm
538 unit uPSI_frmExportMain;                   TSynexportForm
585 unit uPSI_usniffer; {//PortScanForm}       TSniffForm
600 unit uPSI_ThreadForm;                      TThreadSortForm;
618 unit uPSI_delphi_arduino_Unit1;            TLEDForm
620 unit uPSI_fplotMain;                       TfplotForm1
660 unit uPSI_JvDBQueryParamsForm;             TJvQueryParamsDialog
```

```
677 unit uPSI_OptionsFrm;                          TfrmOptions;
718 unit uPSI_MonForm;                             TMonitorForm
742 unit uPSI_SimplePortMain;                      TPortForm1
770 unit uPSI_ovccalc;                             TOvcCalculator  //widget
810 unit uPSI_DbExcept;                            TDbEngineErrorDlg
812 unit uPSI_GL_actorUnit1;                       TglActorForm1  //OpenGL Robot
846 unit uPSI_synhttp_daemon;                      TTCPHttpDaemon, TTCPHttpThrd, TPingThread
867 unit uPSI_Debug;                               TfrmDebug


ex.:with TEditForm.create(self) do begin
      caption:= 'Template Form Tester';
      FormStyle:= fsStayOnTop;
      with editor do begin
        Lines.LoadFromFile(Exepath+'\docs\Readme_rus_mX2.rtf
        SelStart:= 0;
        Modified:= False;
      end;
    end;
  with TWebMainForm.create(self) do begin
    URLs.Text:= 'http://www.kleiner.ch';
    URLsClick(self); Show;
  end;
  with TSynexportForm.create(self) do begin
    Caption:= 'Synexport HTML RTF tester';
    Show;
  end;
  with TThreadSortForm.create(self) do begin
     showmodal; free;
  end;

  with TCustomDrawForm.create(self) do begin
       width:=820; height:=820;
       image1.height:= 600; //add properties
       image1.picture.bitmap:= image2.picture.bitmap;
       //SelectionBackground1Click(self) CustomDraw1Click(self);
       Background1.click;
       bitmap1.click;
       Tile1.click;
       Showmodal;
       Free;
     end;

   with TfplotForm1.Create(self) do begin
     BtnPlotClick(self);
     Showmodal; Free;
   end;

 with TOvcCalculator.create(self) do begin
    parent:= aForm;
  //free;
    setbounds(550,510,200,150);
    displaystr:= 'maXcalc';
  end;


//////////////////////////////////////////////////////////////////////////////
All maXbox Tutorials Table of Content 2014
///////////////////////////////////////////////////////////////////////////////
    Tutorial 00 Function-Coding  (Blix the Programmer)
    Tutorial 01 Procedural-Coding
    Tutorial 02 OO-Programming
    Tutorial 03 Modular Coding
    Tutorial 04 UML Use Case Coding
    Tutorial 05 Internet Coding
    Tutorial 06 Network Coding
    Tutorial 07 Game Graphics Coding
    Tutorial 08 Operating System Coding
    Tutorial 09 Database Coding
    Tutorial 10 Statistic Coding
    Tutorial 11 Forms Coding
    Tutorial 12 SQL DB Coding
    Tutorial 13 Crypto Coding
    Tutorial 14 Parallel Coding
    Tutorial 15 Serial RS232 Coding
    Tutorial 16 Event Driven Coding
    Tutorial 17 Web Server Coding
    Tutorial 18 Arduino System Coding
    Tutorial 18_3 RGB LED System Coding
    Tutorial 19 WinCOM /Arduino Coding
    Tutorial 20 Regular Expressions RegEx
    Tutorial 21 Android Coding (coming 2013)
    Tutorial 22 Services Programming
    Tutorial 23 Real Time Systems
    Tutorial 24 Clean Code
    Tutorial 25 maXbox Configuation I+II
    Tutorial 26 Socket Programming with TCP
    Tutorial 27 XML & TreeView
    Tutorial 28 DLL Coding (available)
    Tutorial 29 UML Scripting (2014)
    Tutorial 30 Web of Things (coming 2014)
    Tutorial 31 Closures (coming 2014)
    Tutorial 32 SQL Firebird (coming 2014)


Doc ref Docu for all Type Class and Const in maXbox_types.pdf
using Docu for this file is maxbox_functions_all.pdf
PEP - Pascal Education Program  Lib Lab

http://stackoverflow.com/tags/pascalscript/hot
http://www.jrsoftware.org/ishelp/index.php?topic=scriptfunctions
http://sourceforge.net/projects/maXbox   #locs:15162
http://sourceforge.net/apps/mediawiki/maXbox
http://www.blaisepascal.eu/
https://github.com/maxkleiner/maXbox3.git
http://www.heise.de/download/maxbox-1176464.html
http://www.softpedia.com/get/Programming/Other-Programming-Files/maXbox.shtml
https://www.facebook.com/pages/Programming-maXbox/166844836691703

  ---- bigbitbox code_cleared_checked----
```