**FM scripting overview**

**Table of Contents**

[1 GENERAL 1](#_Toc507941906)

[1.1 SCREEN 1](#_Toc507941907)

[1.2 STRINGS 1](#_Toc507941908)

[1.3 MATH & STATISTICS 1](#_Toc507941909)

[1.4 TEXT FILE 1](#_Toc507941910)

[1.5 REGISTRY 1](#_Toc507941911)

[1.6 SYSTEM 1](#_Toc507941912)

[1.7 CONVERSIONS 1](#_Toc507941913)

[1.8 GLOBAL VARIABLES 1](#_Toc507941914)

[1.9 DATABASE 1](#_Toc507941915)

[1.10 ARRAY 1](#_Toc507941916)

[1.11 XML 1](#_Toc507941917)

[1.12 DATE 1](#_Toc507941918)

[1.13 MEASUREMENT (SUPPORT OF DEVICES) 1](#_Toc507941919)

[1.14 GIS 1](#_Toc507941920)

[1.15 LISTS 1](#_Toc507941921)

[1.16 SCRIPTORGANIZER *(from version X7)* 1](#_Toc507941922)

[1.17 STEM PROFILE 1](#_Toc507941923)

[1.18 SPECIES GROUPS 1](#_Toc507941924)

[2 LAYERWRAPPER 1](#_Toc507941925)

[2.1 BASIC 1](#_Toc507941926)

[2.2 SCREEN 1](#_Toc507941927)

[2.3 DATABASE 1](#_Toc507941928)

[2.4 XML 1](#_Toc507941929)

[2.5 GIS 1](#_Toc507941930)

[2.6 NAVIGATOR & DQT 1](#_Toc507941931)

[3 TABLEWRAPPER 1](#_Toc507941932)

[4 PROJECT 1](#_Toc507941933)

[4.1 BASIC 1](#_Toc507941934)

[4.2 SCREEN 1](#_Toc507941935)

[4.3 FILES 1](#_Toc507941936)

[4.4 DATABASE 1](#_Toc507941937)

[4.5 NAVIGATOR & DQT & EXTENSIONS 1](#_Toc507941938)

[4.6 GIS 1](#_Toc507941939)

[4.7 EXTRAS 1](#_Toc507941940)

[4.8 STEM PROFILE 1](#_Toc507941941)

[5 LISTS 1](#_Toc507941942)

[5.1 STRINGLIST 1](#_Toc507941943)

[5.2 INTEGERLIST 1](#_Toc507941944)

[5.3 VARIANTLIST 1](#_Toc507941945)

[5.4 MULTIVARIANTLIST 1](#_Toc507941946)

[5.5 VARVARIANTLIST 1](#_Toc507941947)

[*5.6* OBJECTLIST *(from version X7)* 1](#_Toc507941948)

[6 EXCELWRAPPER 1](#_Toc507941949)

[7 POINTWRAPPER 1](#_Toc507941950)

[8 LINEWRAPPER 1](#_Toc507941951)

[9 POLYGONWRAPPER 1](#_Toc507941952)

[10 SPECIESGROUPS 1](#_Toc507941953)

[11 STEM PROFILE 1](#_Toc507941954)

[12 VOLUMEMODELSWRAPPER *(from version X7)* 1](#_Toc507941955)

# GENERAL

## SCREEN

procedure ShowMessage(const str :string)

procedure ShowMsg(const str :string)

procedure ShowInformation(const str :string)

procedure ShowWarning(const str :string)

procedure ShowError(const str :string)

function YesNoQuestion(const str :string) :boolean

function BlinkingMessage (Msg,Msg2,DlgType :string; Timer\_ms :integer; Blink\_ :boolean; BlinkingInterval\_ms\_ :integer): integer *(from version X7)*

procedure OpenWebPage(const PageCaption,URL :string) *(from version X7)*

function InputInteger(const caption, msg :string) :variant

function InputIntegerWithCounter(const caption, msg :string; const Min\_,Max\_,Increment\_ :integer; const PresetValue\_ :variant) :variant

function InputFloat(const caption, msg :string) :variant

function InputString(const caption, msg :string) :variant

function InputText(const caption, msg :string; const text :variant) :variant

function InputOption(const caption, msg :string; labels :array of string; values :array of variant) :variant

function InputOptionInputOptionWithStartingValue(const caption, msg :string; labels :array of string; values :array of variant; const StartingValue :variant; const ShowCancelButton :boolean) :variant

function InputFromList(const caption :string; const msg :string; labels :array of string; values :array of variant; const ControlType :string) :variant

function InputFromListWithStartingValue(const caption :string; const msg :string; labels :array of string; values :array of variant; const StartingValue :variant; const ControlType :string) :variant

function InputFromCheckListTab (caption,msg :string; tab :TFieldMapTableScriptWrapper; FormatString{'%s (%s)},FormattedAtts{;},TaggedAtt,ValAtt{ID attrib} :string) :TVariantListWrapper *(from version X7)*

function InputFromListTab(const caption :string; const msg :string; tab :TTableWrapper; const ValueAttribute :string; const LabelAttributes :array of string; const LabelFormatString :string; const StartingValue :variant; const ControlType :string) :variant

procedure OpenLog(const caption :string)

function IsLogOpen :boolean

procedure HideLog

procedure ShowLog

procedure LogWindowStayOnTop

procedure CloseLog

procedure CloseLogWithDelay(const Delay\_ms :integer)

procedure Log(const str :string)

procedure LogExt(const str :string; appendNewLine :boolean)

procedure LogTabStops(const TabStops\_mm\_ :array of integer) *(from version X7)*

procedure PrepareProgressBar(const NumberOfCycles\_ :integer)

procedure StartProgressBarCycle(const MaxValue\_ :integer)

procedure StepProgressBar

procedure ResetProgressBar

procedure PerformOneStepCycle

procedure OpenNavigator(const XmlData :string)

procedure OpenNavigatorFromFile(const XmlFilename :string)

procedure CloseNavigator

procedure ShowNavigator

procedure ClearNavigator

procedure HideButton(const ButtonName\_ :string)

procedure ShowButton(const ButtonName\_ :string)

procedure SetFieldMapButtonLabel(const ButtonName\_,NewButtonLabel\_ :string)

function GetFingerControl :boolean

function GetButtonExpansionPercentage :integer

function GetFontExpansionPercentage :integer

procedure HideDataCollectorOptionsItem(const ItemName\_ :string)

procedure ShowDataCollectorOptionsItem(const ItemName\_ :string)

function PixelsPerInch :integer

function MainFormWidth :integer

function MainFormHeight :integer

function RGB(R, G, B :byte) :longint

function Color2RGB(const Color\_ :longint) :string

variable clWindow

variable clBtnFace

variable clBlack

variable clMaroon

variable clGreen

variable clOlive

variable clNavy

variable clPurple

variable clTeal

variable clGray

variable clSilver

variable clRed

variable clLime

variable clYellow

variable clBlue

variable clFuchsia

variable clAqua

variable clLtGray

variable clMedGray

variable clDkGray

variable clWhite

variable clMoneyGreen

variable clForestGreen

variable clLightBlueGreen

variable clOceanGreen

variable clMossGreen

variable clDarkGreen

variable clGrassGreen

variable clKentuckyGreen

variable clLightGreen

variable clSpringGreen

## STRINGS

function Format(s :string; const args :array of const) :string

function LZero(const Value :variant; const Len :integer) :string

procedure Delete(var Data :string; const Index :integer; const Count :integer)

function Trim(const s :string) :string

function TrimLeft(const s :string) :string

function TrimRight(const s :string) :string

function SameText(const Data1,Data2 :string) :boolean

function StringReplace(const Data,OldPattern,NewPattern :string; const ReplaceAll,CaseInsensitive :boolean) :string

function DeFormat1(Data :String; Format :String; var r1 :variant) :integer

function DeFormat2(Data :String; Format :String; var r1, r2 :variant) :integer

function DeFormat3(Data :String; Format :String; var r1, r2, r3 :variant) :integer

function DeFormat4(Data :String; Format :String; var r1, r2, r3, r4 :variant) :integer

function DeFormat5(Data :String; Format :String; var r1, r2, r3, r4, r5 :variant) :integer

function DeFormat6(Data :String; Format :String; var r1, r2, r3, r4, r5, r6 :variant) :integer

function DeFormat7(Data :String; Format :String; var r1, r2, r3, r4, r5, r6, r7 :variant) :integer

function DeFormat8(Data :String; Format :String; var r1, r2, r3, r4, r5, r6, r7, r8 :variant) :integer

function DeFormat9(Data :String; Format :String; var r1, r2, r3, r4, r5, r6, r7, r8, r9 :variant) :integer

function DeFormat10(Data :String; Format :String; var r1, r2, r3, r4, r5, r6, r7, r8, r9, r10 :variant) :integer

function GetFieldCountFromString(const Data :string; const FieldDelimiter :char) :integer

function GetFieldFromString(const Data :string; const Index :integer; const FieldDelimiter :char) :string

function IndexOfField(const Data,Field :string; const FieldDelimiter :char ' ') :integer

function SortNumericSequence(const Data :string; const FieldDelimiter :char; const Descending :boolean) :string

## MATH & STATISTICS

function Power(X, Y :extended) :extended

function Round(X :double; NumberOfDigits\_ :integer) :double

function RoundVariant (X :variant; NumberOfDigits\_ :integer) :variant *(from version X7)*

function ArcCos(const x :extended) :extended

function ArcCosH(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 ArcTanH(const x :extended) :extended

function CosH(const x :extended) :extended

function SinH(const x :extended) :extended

function Tan(const x :extended) :extended

function TanH(const x :extended) :extended

function Random :extended

function RandomRange(const AFrom, ATo :integer) :integer

procedure Randomize

function Floor(X :extended) :integer

function Ceil(X :extended) :integer

function Int (const X :double) :double *(from version X7)*

function Frac (const X :double) :double *(from version X7)*

function Log10(const X :Extended) :extended

function Ln(const X :extended) :extended

function Exp(const X :extended) :extended

function Abs(const X :extended) :extended

function Min(const X, Y :extended) :extended

function Max(const X, Y :extended) :extended

function Odd(const X :integer) :boolean *(from version X7)*

function SecondsBetween(const DateTime1,DateTime2 :variant) :int64 *(from version X7)*

function CompareValue(const A, B :Double; Epsilon :Double 0) :integer {-1..+1}

function SameValue(const A, B :Double; Epsilon :Double 0) :boolean

function Mean(tab :TTableWrapper; const AttributeName\_ :string; const NullAsZero\_ :boolean) :extended

function QuadraticMean(tab :TTableWrapper; const AttributeName\_ :string; const NullAsZero\_ :boolean) :extended

function Median(tab :TTableWrapper; const AttributeName\_ :string; const NullAsZero\_ :boolean) :extended

function SampleVariance(tab :TTableWrapper; const AttributeName\_ :string; const NullAsZero\_ :boolean) :extended

function StandardDeviation(tab :TTableWrapper; const AttributeName\_ :string; const NullAsZero\_ :boolean) :extended

function StandardError(const StandardDeviation\_ :extended; const Nsample\_ :integer) :extended

function MeanError(const StandardDeviation\_ :extended; const Nsample\_ :integer) :extended

function VarianceOfMean(tab :TTableWrapper; const AttributeName\_ :string; const NullAsZero\_ :boolean; const Nsample\_ :integer; const NPopulation\_ :extended) :extended

function ConfidenceIntervalOfMean(const Nsample\_ :integer; const VarianceOfMean\_ :extended; const Alpha\_ :integer{ 5}) :extended

function VarianceOfTotal(const VarianceOfMean\_ :extended; const Npopulation\_ :extended) :extended

function ConfidenceIntervalOfTotal(const Nsample\_ :integer; const VarianceOfTotal\_ :extended; const Alpha\_ :integer{ 5}) :extended

function StratifiedConfidenceIntervalOfTotal(tab :TTableWrapper; const PopSizeAttribute\_,SampleSizeAttribute\_,SampleVarianceAttribute\_,VarianceOfMeanAttribute\_ :string; const Alpha\_ :integer; out ConfInterval\_ :variant) :boolean

function SampleSize(const RelativePrecission{0..1},SampleVariance\_,Mean\_ :extended; const StudentLevel :integer) :integer

procedure ShowDistributionChart(tab :TTableWrapper; const AttributeName\_ :string; const NullAsZero\_ :boolean; const ClassWidth\_ :variant; const Xlabel\_,ChartLabel\_,Options\_ :string)

function ParameterizeRegressionModel(const DataXML\_,XattributeName\_,YattributeName\_,Xlabel\_,Ylabel\_,ChartCaption\_,ModelType\_{e.g. POWER} :string; const Options\_ :string; out P1\_,P2\_,P3\_,Rcoef\_,Syx\_,Count\_ :variant) :boolean

function CalculateModelValue(const ModelType\_ :string; {e.g. POWER} const P1\_,P2\_,P3\_,Yshift\_ :variant; const VarValue\_ :variant) :variant

## TEXT FILE

function AssignFile(const filename :string) :integer

procedure Reset(f :integer)

procedure Rewrite(f :integer)

procedure Append(f :integer)

procedure WriteLn(f :integer; const line :string)

function ReadLn(f :integer) :string

function EOF(f :integer) :boolean

procedure CloseFile(f :integer)

function DeleteFile(const FilePath :string) :boolean

function SaveStringToFile(const Data,Filename :string) :boolean

function SaveStringToFileAsUTF8(const Data,Filename :string) :boolean

## REGISTRY

SaveValueToRegistry(const KeyName\_,ValueName\_ :string; const Value\_ :variant) *(from version X7)*

LoadValueFromRegistry(const KeyName\_,ValueName\_ :string; const DefaultValue\_ :variant) :variant *(from version X7)*

## SYSTEM

variable ScriptResult

variable ValidationFailedMessageText

variable DisableEvents

variable ProjectDir

variable UserDir

variable SystemDir

variable ProjectName

procedure CloseFieldMap(const Message\_ :string)

procedure Abort

procedure ProcessMessages

procedure Sleep(milliseconds :cardinal)

function SoftwareKey :string

function GetTickCount :integer

procedure Beep

procedure PlaySoundFromFile(const FilePath :string)

procedure PlaySoundFromLibrary(IDSound :integer)

procedure ReleaseSound(const FilePath :string)

procedure SwitchSoundsOff *(from version X7)*

procedure SwitchSoundsOn *(from version X7)*

procedure Speak(const AText: string)

function DecimalSeparator :char

function SetDecimalPoint :char

function SetDecimalComma :char

procedure ResetDecimalSeparator

function CreateGUID :string

function HourGlassCursor :integer

procedure DefaultCursor

function DisableFieldMapWindow :boolean

procedure EnableFieldMapWindow(const Enabled\_ :boolean)

procedure SetRightTitleBarCaption(const Caption\_ :string)

function RunFieldMapExtension(const DLLfilename :string) :boolean

## CONVERSIONS

function String2Float(const Data :string) :variant

function Variant2Float(const Data :variant) :double

function Variant2Integer(const Data :variant) :integer *(from version X7)*

function Variant2String(const Data :variant) :string

function String2Variant (const Data :string) :variant *(from version X7)*

function Boolean2Integer(const Data :boolean) :integer {0-false 1-true}

function Date2Integer(Date :variant) :variant {Date->YYYYMMDD}

function Integer2Date(Date :variant) :variant {YYYYMMDD->Date}

function Int2Hex(const Value :integer; const Digits\_ :integer) :string

function IntToHex(const Value :integer; const Digits\_ :integer) :string

function GetBitFromLastThreeDigits(const Value :variant; const Bit :integer{0..7}) :integer {0/1}

## GLOBAL VARIABLES

procedure Globals.SetGlobalVariable(const name :string; value :variant)

function Globals.GetGlobalVariable(const name :string) :variant

function Globals.GlobalVariableExists(const name :string) :boolean

procedure Globals.RemoveGlobalVariable(const name :string)

procedure Globals.RemoveGlobalVariables

property Globals.Value[const name :string] :variant

## DATABASE

function OpenTable(const a\_database\_path, a\_table\_name :string) :TTableWrapper

function LoadSimpleXML (const a\_filename :string) : TTableWrapper *(from version X7)*

function OpenParadoxTable(const a\_table\_path :string) :TTableWrapper

function CreateMemTableFromDefs(const FieldDefs\_,IndexFields\_ :string) :TTableWrapper

procedure CloseAndFreeTable(a\_table :TTableWrapper)

const dsEdit

const dsInsert

function AddintegerAttribute(const Name\_ :string; const Label\_ :string; Size\_ :integer; ReadOnly\_ :boolean) :boolean

function AddFloatAttribute(const Name\_ :string; const Label\_ :string; Size\_ :integer; DecimalPlaces\_ :integer; ReadOnly\_ :boolean) :boolean

function AddbooleanAttribute(const Name\_ :string; const Label\_ :string; ReadOnly\_ :boolean) :boolean

function AddStringAttribute(const Name\_ :string; const Label\_ :string; Size\_ :integer; ReadOnly\_ :boolean) :boolean

function AttributeExists(const Name\_ :string) :boolean

property DefaultValue[const name :string] :variant

property DefaultValueEnabled [const name :string] :boolean *(from version X7)*

function GetAttributesWithDefaultValue :string

## ARRAY

procedure SetArrayLength (var A; NewLength: integer) *(from version X7)*

function GetArrayLength (var A): integer; *(from version X7)*

function VarArrayOf(const Data :array of variant) :variant *(from version X7)*

function VarArrayCreate(const Bounds :array of integer; VarType\_ :word) :variant *(from version X7)*

function VarArrayDimCount(const Data: Variant): Integer; *(from version X7)*

function VarArrayLowBound(const Data: Variant; Dim: Integer): Integer; {Dim=1 = nejnizsi dimenze} *(from version X7)*

function VarArrayHighBound(const Data: Variant; Dim: Integer): Integer; *(from version X7)*

function VarType(const Data :variant) :word; *(from version X7)*

function VarIsArray(const Data :variant) :boolean; *(from version X7)*

function VarIsNumeric(const Data :variant) :boolean; *(from version X7)*

procedure SetVarArrayValue(var Data :variant; const Index :array of integer; const Value :variant) *(from version X7)*

function GetVarArrayValue(const Data :variant; const Index :array of integer) :variant *(from version X7)*

## XML

function OpenXMLDocument(const FileName :string) :TScriptXMLDocument

function OpenLockedXMLDocument(const Filename :string) :TScriptXMLDocument

function OpenXMLDocumentExt(const FileName :string; var ErrorMsg :string) :TScriptXMLDocument

function OpenXMLDocumentFromString(const xml :string) :TScriptXMLDocument

function OpenXMLDocumentFromStringExt(const xml :string; var ErrorMsg :string) :TScriptXMLDocument;

function GetAttributeValueFromXMLDocument(const Filename\_,ElementPath\_,AttributeName\_ :string) :variant *(from version X7)*

procedure FreeXMLDocument(var AXMLDoc :TScriptXMLDocument)

procedure FreeXMLElement(var AElement :TScriptXMLElement)

procedure FreeXMLElementList(var AElementList :TScriptXMLElementList)

function GetRootElementName(const Filename :string) :string *(from version X7)*

function TScriptXMLDocument.GetRootElement :TScriptXMLElement; *(from version X7)*

property TScriptXMLElement.Name :string read GetElementName *(from version X7)*

function TScriptXMLElement.GetElementName :string *(from version X7)*

function TScriptXMLElement.GetChildElements(const ElementPath :string) :TScriptXMLElementList *(from version X7)*

function TScriptXMLElement.GetFirstChildElement(const ElementPath :string) :TScriptXMLElement *(from version X7)*

function TScriptXMLElement.GetElementText :string *(from version X7)*

function TScriptXMLElement.GetElementinteger :variant *(from version X7)*

function TScriptXMLElement.GetElementDouble :variant *(from version X7)*

function TScriptXMLElement.GetElementboolean :variant *(from version X7)*

function TScriptXMLElement.GetElementCDATA :string *(from version X7)*

function TScriptXMLElement.GetFirstChildElementText(const ElementPath :string) :string

function TScriptXMLElement.GetFirstChildElementinteger(const ElementPath :string) :variant

function TScriptXMLElement.GetFirstChildElementDouble(const ElementPath :string) :variant

function TScriptXMLElement. GetFirstChildElementboolean(const ElementPath :string) :variant

function TScriptXMLElement.HasAttribute(const AttributeName :string) :boolean

function TScriptXMLElement.GetStringAttribute(const AttributeName :string) :string

function TScriptXMLElement.GetStringAttributeAsString(const AttributeName :string) :string

function GetStringAttributeAsDateTime(const AttributeName :string) :variant

function TScriptXMLElement.GetIntegerAttribute(const AttributeName :string) :variant

function TScriptXMLElement.GetIntegerAttributeAsinteger(const AttributeName :string; const NullValue :integer) :integer

function TScriptXMLElement.GetFloatAttribute(const AttributeName :string) :variant

function TScriptXMLElement.GetFloatAttributeAsFloat(const AttributeName :string; const NullValue :double) :double

function TScriptXMLElement.GetFirstChildElementStringAttribute(const ElementPath, AttributeName :string) :variant

function TScriptXMLElement.GetFirstChildElementStringAttributeAsString(const ElementPath, AttributeName :string) :string

function TScriptXMLElement.GetFirstChildElementintegerAttribute(const ElementPath, AttributeName :string) :variant

function TScriptXMLElement.GetFirstChildElementintegerAttributeAsinteger(const ElementPath, AttributeName :string; const NullValue :integer) :integer

function TScriptXMLElement.GetFirstChildElementFloatAttribute(const ElementPath, AttributeName :string) :variant

function TScriptXMLElement.GetFirstChildElementFloatAttributeAsFloat(const ElementPath, AttributeName :string; const NullValue :double) :double

function TScriptXMLElement.GetFirstChildElementbooleanAttribute(const ElementPath, AttributeName :string) :variant

function TScriptXMLElementList.GetCount :integer *(from version X7)*

function TScriptXMLElementList.GetElement(index :integer) :TScriptXMLElement *(from version X7)*

function CreateXMLOutput :TXmlOutputWrapper

function CreateXMLOutputFile(const FileName :string) :TXmlOutputWrapper

procedure FreeXMLOutput(var AXMLOutput :TXmlOutputWrapper)

## DATE

function Day :integer

function Month :integer

function Year :integer

procedure DecodeDate(const Date\_ double; out Year,Month,Day :word)

function DecodeDateExt(const Date\_ :variant; out Year,Month,Day :integer) :boolean

function DecodeDateTimeExt(const DateTime\_ :variant; out Year,Month,Day,Hour,Minute,Second :integer) :boolean

function YearOf(const Date\_ :variant) :variant

function MonthOf(const Date\_ :variant) :variant

function DayOf(const Date\_ :variant) :variant

function DaysInAMonth(const Year, Month :integer) :integer

function StrToDateTime(const Data :string) :double

function DateToStr(const Date :variant) :string

function DateTimeToStr(const DateTime :variant) :string

function DateTimeAsString(DateTime :variant) :variant; {DateTime->YYYYMMDDHHMMSS}

function FormatElapsedTime(const Seconds\_ :integer) :string

## MEASUREMENT (SUPPORT OF DEVICES)

procedure CancelMeasurement

function MeasureHorizDist(out HD\_m :variant; const Caption\_ :string) :boolean

function MeasureInclination(out INCL\_deg :variant; const Caption\_ :string) :boolean

function MeasureHorizDistAndInclination(out HD\_m,INCL\_deg :variant; const Caption\_ :string) :boolean

function MeasureRemoteDiameter(InclinationToStemBase\_deg,HoriDistToStemCenter\_m :variant; var Height\_m :variant; Diameter\_mm :variant; const Caption\_ :string) :boolean

function MeasureStemProfile(const Species,DBH\_mm,TreeHeight\_m,InclinationToStemBase\_deg,HoriDistToStemCenter\_m,AdditionalPoints :variant; var XML\_ :variant; const Caption\_ :string) :boolean

function GetContinuousGpsCoordinates(out X\_,Y\_ :variant) :boolean

procedure GetCurrentPosition(out X\_,Y\_,Z\_ :variant)

procedure SetCurrentPosition(out X\_,Y\_,Z\_ :variant) *(from version X7)*

procedure ResetOnlineCalliper(const LayerName\_ :string)

procedure SendMessageToNonSpecificEquipment(const EquipmentID\_ :integer const Message\_ :string)

procedure ManageNonSpecificEquipment(const EquipmentID\_ :integer; const Commands\_ :array of string) e.g. PURGEING,PURGEOUT

function NonSpecificEquipmentActive(const EquipmentID\_ :integer) :boolean

function DendroscopeSerialNumber :string *(from version X7)*

function CalcVerticalDistance(const HorizDist\_m,Inclination1\_deg,Inclination2\_deg :variant) :variant

function VerticalDistance(const HorizDist\_m,Inclination1\_deg,Inclination2\_deg :variant) :variant *(from version X7)*

function CalcInclination(const HorizDist\_m,VertDist\_m :variant) :variant *(from version X7)*

function CheckSum(const Data :string) :byte

function GetBearingAngle\_deg :variant

function GetBearingAzimuth\_deg :variant

function GetLaserEquipmentHeight\_cm :variant

function GetAzimuthFixOffset\_deg :variant

function GetAzimuthUnits :variant

function GetMagneticDeclination\_deg :variant

function GetMainPoleLength\_cm :variant

function GetMainPoleLengthCorrection\_cm :variant

function GetMainPoleOffsetForTreeMeasurement\_cm :variant

function GetMainPoleReflectorOffset\_cm :variant

function GetOffsetOfCylReflector\_cm :variant

function GetGPSAntennaHeight\_cm :variant

function GetPolePositionMode :variant

procedure SetPromptBeforeChangeTreePositionOfPreviousMeasurement(Value :boolean)

## GIS

function ExtendLine(var X1\_,Y1\_,X2\_,Y2\_ :double; const Addition\_ :double; const Mode\_ :integer) :boolean

function CombinePolygons(const pol1\_,pol2\_ :TPolygonWrapper) :TPolygonWrapper

function IntersectPolygons(const pol1\_,pol2\_ :TPolygonWrapper) :TPolygonWrapper

function FindLineIntersects(lin1\_,lin2\_ :TLineWrapper) :TPointsWrapper *(from version X7)*

function Distance(const X1\_,Y1\_,X2\_,Y2\_ :double) :double *(from version X7)*

procedure PolarToXY(const X0\_,Y0\_,Azimuth\_deg,Distance\_ :double; var X\_,Y\_ :double) *(from version X7)*

procedure Polar2XY(const X0\_,Y0\_,Azimuth\_deg,Distance\_ :double; var X\_,Y\_ :double) *(from version X7)*

function XYtoPolar(const X0\_,Y0\_,X\_,Y\_ :double; var Azimuth\_deg,Distance\_ :double) :boolean *(from version X7)*

function XY2polar(const X0\_,Y0\_,X\_,Y\_ :double; var Azimuth\_deg,Distance\_ :double) :boolean *(from version X7)*

## LISTS

function CreateStringList :TStringListWrapper

procedure FreeStringList(var List\_ :TStringListWrapper)

function CreateIntegerList :TIntegerListWrapper

procedure FreeIntegerList(var List\_ :TIntegerListWrapper)

function CreateVariantList :TVariantListWrapper

procedure FreeVariantList(var List\_ :TVariantListWrapper)

function CreateMultiVariantList(const NumberOfVariantsPerItem\_ :integer) :TMultiVariantListWrapper

procedure FreeMultiVariantList(var List\_ :TMultiVariantListWrapper)

function CreateVarVariantList :TVarVariantListWrapper

procedure FreeVarVariantList(var List\_ :TVariantListWrapper)

## SCRIPTORGANIZER *(from version X7)*

constructor Create

procedure Free

property Caption :variant

property Header :variant

procedure AddGroup(const Caption\_,Note\_ :string)

procedure AddSubGroup(const Caption\_,Note\_ :string)

procedure AddAttributeScript(const Caption\_,LayerName\_,AttributeName\_,Note\_ :string)

procedure AddOnDemandScript(const Caption\_,ScriptName\_,Note\_ :string)

function Show :boolean

property RegistryNameForLastPositionStyle{-B} :string

## STEM PROFILE

function CreateStemProfile :TStemProfileWrapper

procedure FreeStemProfile(var StemProfile\_ :TStemProfileWrapper)

## SPECIES GROUPS

function CreateSpeciesGroups :TSpeciesGroupsWrapper

procedure FreeSpeciesGroups (var SpeciesGroups\_ :TSpeciesGroupsWrapper)

# LAYERWRAPPER

## BASIC

property Name :string

procedure Free

property ParentName :string

property LayerType :integer

function LayerLevel :integer

property LayerLabel :string

property AutoTimeStampEnabled[const name :string] :boolean

function PerformOnChangeScript(const AttributeName\_ :string) :boolean

function PerformOnChangeScriptForAllRecords(const AttributeName\_ :string) :boolean

function PerformLayerScript(const EventTypeName\_ :string) :boolean

procedure OnLineCalliper\_AddAttribute(const AttributeName\_ :string)

procedure OnLineCalliper\_RemoveAttribute(const AttributeName\_ :string)

procedure OnLineCalliper\_ClearAttributes

function OnLineCalliper\_GotoAttribute(const AttributeName\_ :string) :boolean *(from version X7)*

property OnLineCalliper\_AddRecordUponReceipt :boolean *(from version X7)*

function GetCrownProfileVolumeAndSurface (const TreeID\_ :integer; const ProfileID\_ :variant; out Volume\_m3,Surface\_m2 :variant) :boolean *(from version X7)*

function GetCrownProjectionAsLine (const TreeID\_ :integer; const Smoothed\_ :boolean) :TLineWrapper *(from version X7)*

function GetCrownProjectionAsLine (const TreeID\_ :integer; const Smoothed\_ :boolean) :TPolygonWrapper *(from version X7)*

procedure OpenExternalForm(const DLLname\_ :string)

property ImportShapefile.Filename :string *(from version X7)*

procedure ImportShapefile.AssignAttribute(const From\_,To\_ :string) *(from version X7)*

property ImportShapefile.ImportOnlyCentroids :boolean *(from version X7)*

function ImportShapefile.Execute :integer *(from version X7)*

## SCREEN

procedure RefreshAttributeLayout

property TabStop[const AttributeName :string] :boolean

property Enabled[const name :string] :boolean

procedure EnableAttributes(const ExcludingAttribs\_ :string)

procedure DisableAttributes(const ExcludingAttribs\_ :string)

property Visible[const name :string] :boolean

procedure ShowAttributes(const ExcludingAttribs\_ :string)

procedure HideAttributes(const ExcludingAttribs\_ :string)

property AlignAttribute[const name : string] :string LEFT,RIGHT,CENTER *(from version X7)*

procedure AlignAttributes(const AttributeNames\_,Alignment\_ :string) LEFT,RIGHT,CENTER *(from version X7)*

procedure AlignOverviewGridTitles(const Alignment\_ :string) LEFT,RIGHT,CENTER *(from version X7)*

property OverviewGridAttributeTitle[name\_ :string] :string *(from version X7)*

procedure ClickToAttributeControl(const name :string) *(from version X7)*

property UseKeyboard4ComboBox[const name :string] :boolean *(from version X7)*

property OverviewGridAttributeVisible[name :string] :boolean

procedure ShowOverviewGridAttribute(const name :string)

procedure HideOverviewGridAttribute(const name :string)

property Color[const name :string] :longint

procedure ResetColor(const name :string)

procedure SetParentColor(const name :string)

procedure SetColorOfAttributes(const Color\_ :integer; const ExcludingAttribs\_ :string)

procedure ResetColorOfAttributes(const ExcludingAttribs\_ :string)

procedure SetParentColorOfAttributes(const ExcludingAttribs\_ :string)

procedure EnableControls

procedure DisableControls

function ControlsDisabled :boolean

procedure EnableScreenControls

procedure DisableScreenControls

function ScreenControlsDisabled :boolean

function SetFocus(const AttributeName\_ :string) :boolean

property DisplayFormat[const name :string] :string

property AttributeHint[AttributeName\_ :string] :string

procedure ResetAttributeHint(const AttributeName\_ :string)

property AttributeLabel[AttributeName\_ :string] :string

property PlacerStrategy :string {LEFT,TOP,FULL}

propertyMaxSpaceForControlsPerc :integer

function GoToLayerPage :boolean

function IsLayerPageOnTop :boolean

property TabVisible :boolean

property LayerPageVisible :boolean

property StemProfilePageVisible :boolean *(from version X7)*

procedure ShowInSeparateWindow *(from version X7)*

## DATABASE

property Value[const name :string] :variant

property ValueAsString[const name :string] :string

property ValueAsinteger[const name :string] :integer

property ValueAsFloat[const name :string] :double

property ValueAsboolean[const name :string] :boolean

property OldValue[const name :string] :variant

property PreviousValue[const name :string] :variant

property BOF :boolean

property EOF :boolean

property RecordCount :integer

property State :integer

procedure SetValues(const AttributeNames\_ :string; const Values :array of variant)

procedure SetValuesFromXML(el\_ :TScriptXMLElement)

property CrownProjection :TTableWrapper

property CrownProfile :TTableWrapper

property CrownProfilePoints :TTableWrapper

property StemProfile :TTableWrapper

property OldLayer :TFieldMapLayerWrapper read WOldLayer

procedure SavePosition

procedure RestorePosition

procedure SaveState

procedure RestoreState

procedure SavePositionAndState

procedure RestorePositionAndState

procedure Open

procedure Close

procedure First

procedure Last

procedure Next

procedure Prior

procedure Edit

procedure Post

function Append :boolean

procedure Delete

procedure Cancel

procedure Refresh

function Locate(const KeyFields :string; const KeyValues :Variant; CaseInsensitive :boolean; PartialKey :boolean) :boolean

procedure ClearTable

property Filter :string

property Filtered :boolean

function CheckFilter(const Filter\_ :string; out ErrorMessage\_ :string) :boolean

function SetFilterExt(const Filter\_ :string; const CanseInsensitive\_,PartialFit\_ :boolean) :boolean

function ClearFilterForChildLayers :boolean

property DeepFilter :string

procedure EnableChachedUpdates

procedure ApplyUpdates

function GetAttributeValue(const AttributeName\_ :string; const Where\_ :string) :variant

function GetQueryResult(const Select\_,Where\_ :string) :TTableWrapper

procedure DeleteFrom(const Where\_ :string)

procedure Update(const Set\_,Where\_ :string)

function CreateMemTable :TTableWrapper

function SaveToDBF(const Filename\_ :string) :boolean

function SaveToExcel(const Filename\_ :string; const SheetName\_ :string; const OpenExcel\_ :boolean =false) :boolean

function GetMinOfAttributeValues(const AttributeName\_ :string; const Where\_ :string) :variant

function GetMaxOfAttributeValues(const AttributeName\_ :string; const Where\_ :string) :variant

function GetMeanOfAttributeValues(const AttributeName\_ :string; const Where\_ :string) :variant

function GetSumOfAttributeValues(const AttributeName\_ :string; const Where\_ :string) :variant

function GetRecordCount(const Where\_ :string) :integer

function GetLookupTable(const AttributeName\_ :string) :TTableWrapper

function GetLookupListValue(AttributeName\_ :string; AttributeValue\_ :variant) :string

function GetLookupListAttributeValue(AttributeName\_ :string; AttributeValue\_ :variant; LookupListAttribName\_ :string) :variant

function GetID(const RelativeLevel\_ :integer) :integer

function GetIdAttributeName(const RelativeLevel\_ :integer) :string

function GetIdSQL(const StartFromRelativeLevel\_ :integer) :string

procedure SaveTimeStamp(const AttributeName\_ :string)

function File2Blob(const SourceFilename\_,BlobAttributeName\_ :string) :boolean

function Blob2Folder(const DestFolder\_,BlobAttributeName\_ :string) :boolean

function String2Blob(const Data\_,BlobAttributeName\_ :string) :boolean

function Blob2String(const BlobAttributeName\_ :string) :string

function FillAttributeValues(const AttributeNames\_ :string; Values\_ :array of variant)

procedure MarkValuesAsConditional(const AttributeNames\_,Where\_ :string)

procedure RoundAttributeValues(const AttributeNames\_ :string; const PlacesBehindDecimalPoint\_ :integer)

## XML

procedure AddDataToXML(xmlout\_ :TXmlOutputWrapper; const TableElementName\_ :string)

function GetRecordAsXML(const ID\_ :variant; const IncludingChildLayers\_ :boolean) :string

function GetRecordsAsXML(const IncludingChildLayers\_ :boolean) :string

function GetLookupListContentAsXML(const AttributeName\_ :string; const OnlyActiveItems\_, IncludingConditionalLookupLists\_ :boolean) :string

function GetLookupListsContentAsXML(const OnlyActiveItems\_, IncludingConditionalLookupLists\_ :boolean) :string

procedure AddDataToXML(xmlout\_ :TXmlOutputWrapper; const IncludingChildLayers\_ :boolean)

## GIS

property GisLayerReference :TFieldMapLayerWrapper read GetGisLayerReference

function GetParentPolygonArea\_m2 :variant

function FindIntersectingPolygons(const SearchPolygonLayerName\_ :string; const SearchPolygonID\_ :integer) :TIntegerListWrapper

function GetIntersectingPolygonAreas(const SearchPolygonLayerName\_ :string; const SearchPolygonID\_ :integer) :TMultiVariantListWrapper

function FindContainedPointsOrCentroids(const SearchPolygonLayerName\_ :string; const SearchPolygonID\_ :integer) :TIntegerListWrapper

function FindLinesIntersectingPolygon(const SearchPolygonLayerName\_ :string; const SearchPolygonID\_ :integer) :TIntegerListWrapper

function FindClosestLine(const X\_,Y\_ :double) :variant

function FindPolygonContainingPoint(const SearchPointLayerName\_ :string; const SearchPointID\_ :integer) :TIntegerListWrapper

function AddPoint(const X\_,Y\_ :double; const ID\_ :integer) :boolean

function AddLine(const XY\_ :string; const ID\_ :integer) :boolean

function AddPolygon(const XY\_ :string; const ID\_ :integer) :boolean

function AddPolygonWithCentroid(const XY\_ :string; const CenterX\_m,CenterY\_m :variant; const ID\_ :integer) :boolean

function AddCircularPolygonWithCentroid(const Radius\_m,CenterX\_m,CenterY\_m :double; const ID\_ :variant) :boolean *(from version X7)*

procedure SplitMultiPolygon(const ID\_ :integer)

procedure DeletePoint(const ID\_ :integer)

procedure DeleteLine(const ID\_ :integer)

procedure DeletePolygon(const ID\_ :integer)

procedure DeletePolygonsKeepCentroidsAndAttributes(const IDs\_ :string)

procedure SaveShapesToBlobs

procedure UpdateSymbolAndLabelInShapefiles(const ChangeSymbol\_,ChangeLabel\_ :boolean)

function CreatePointsAlongLine(const LineID\_ :integer; const TargetPointLayerName\_ :string; const NumberOfPoints\_ :integer; const Offset\_m\_ :variant) :string

procedure RefreshMap

procedure RefreshMapRectangle(const Left\_m,Right\_m,Bottom\_m,Top\_m :double)

procedure ZoomAndSelect(tab :TTableWrapper)

procedure ZoomAndSelectEntity(const ID\_ :integer)

procedure SelectEntities(const IDs\_ :string)

function MeasureLineLength\_m(const ID\_ :integer) :Variant

function MeasurePolygonArea\_m2(const ID\_ :integer) :Variant

function GetLineCoordinates(const ID\_ :integer; out XY\_ :string) :boolean

function GetPolygonCoordinates(const ID\_ :integer; out XY\_ :string) :boolean

function GisEntityExists(const ID\_ :integer) :boolean

function GetCombinedPolygonsCoordinates(const IDs\_ :string; out XY\_ :string) :boolean

function SelectLine(const X\_m,Y\_m :double; out ID\_ :integer) :boolean

## NAVIGATOR & DQT

procedure AddItemToNavigatorXML(const XMLfilename\_ :string; const Caption\_ :string; const Hint\_ :string ='')

property DBNavigatorVisible :boolean

property DBNavigatorConfirmDelete :boolean

property QueryBuilderEnabled :boolean

procedure EnableQueryBuilder

procedure DisableQueryBuilder

# TABLEWRAPPER

procedure Free

function FieldExists(const name :string) :boolean

property Value[const name :string] :variant

property ValueAsString[const name :string] :string

property ValueAsinteger[const name :string] :integer

property ValueAsFloat[const name :string] :double

property ValueAsboolean[const name :string] :boolean

property Visible[const name :string] :boolean

property Enabled[const name :string] :boolean

property BOF :boolean

property EOF :boolean

property RecordCount :integer

property RecNo :integer

property State :integer

procedure SavePosition

procedure RestorePosition

procedure SaveState

procedure RestoreState

procedure SavePositionAndState

procedure RestorePositionAndState

procedure EnableControls

procedure DisableControls

function ControlsDisabled :boolean

procedure Open

function IsTableOpen :boolean

procedure Close

procedure First

procedure Last

procedure Next

procedure Prior

procedure Edit

procedure Post

function Locate(const KeyFields :string; const KeyValues :Variant; CaseInsensitive :boolean; PartialKey :boolean) :boolean

property Filter :string

property Filtered :boolean

function CheckFilter(const Filter\_ :string; out ErrorMessage\_ :string) :boolean

function SetFilterExt(const Filter\_ :string; const CanseInsensitive\_,PartialFit\_ :boolean) :boolean

procedure Append

procedure Delete

procedure Cancel

procedure Refresh

procedure EnableChachedUpdates

procedure ApplyUpdates

procedure ClearTable

procedure AddDataToXML(xmlout\_ :TXmlOutputWrapper; const TableElementName\_ :string) *(from version X7)*

function AsXML :string

function GetRecordsAsXML :string

procedure SetValues(const AttributeNames\_ :string; const Values :array of variant)

procedure SetValuesFromXML(el\_ :TScriptXMLElement)

function CreateMemTable :TTableWrapper

function SaveToDBF(const Filename\_ :string) :boolean

function SaveToDbfExt(const Filename\_,FieldSizes\_ :string) :boolean

function SaveToExcel(const Filename\_ :string; const SheetName\_ :string; const OpenExcel\_ :boolean =false) :boolean

procedure AlterTable(const FieldName\_,DataType\_ :string)

property DisplayFormat[const name :string] :string

property DisplayLabel[const name :string] :string

function File2Blob(const SourceFilename\_,BlobAttributeName\_ :string) :boolean

function Blob2Folder(const DestFolder\_,BlobAttributeName\_ :string) :boolean

function String2Blob(const Data\_,BlobAttributeName\_ :string) :boolean

function Blob2String(const BlobAttributeName\_ :string) :string

property BlobAsString [BlobAttributeName\_ :string] :string *(from version X7)*

# PROJECT

## BASIC

property Name :string

property DatabaseType :string

property UserLoginName :string

property UserGroupName :string

property LayerCount :integer

property Layer[const name :string] :TFieldMapLayerWrapper

property ActiveLayer :TFieldMapLayerWrapper read GetActiveLayer

property ActiveLayerName :string read GetActiveLayerName

property LayerAtIndex[index :integer] :TFieldMapLayerWrapper

function SysTab\_LayerList :TTableWrapper

function SysTab\_AttributeList :TTableWrapper

function SendDataToAnotherRunningProject (const TargetProjectDir\_ :string; const MessageID\_ :integer; const MessageGUID\_, Data\_ :string) :boolean *(from version X7)*

function GetResourceString(const Language\_,Name\_ :string) :string

function OpenInactiveLayers(InactiveLayers\_ :string) :boolean

procedure CloseInactiveLayers

function OpenPlot(ID :integer) :boolean

function OpenPlotUsingDataCollector(ID :integer) :boolean

procedure RunPolyshape *(from version X7)*

procedure AddLayerWrapper(a\_layer :TFieldMapLayerWrapper)

property ScriptsEnabled :boolean

property LayerEventScriptsEnabled :boolean

property AfterCancelScriptsEnabled :boolean

property AfterDeleteScriptsEnabled :boolean

property AfterEditScriptsEnabled :boolean

property AfterInsertScriptsEnabled :boolean

property AfterPostScriptsEnabled :boolean

property AfterScrollScriptsEnabled :boolean

property BeforeCancelScriptsEnabled :boolean

property BeforeDeleteScriptsEnabled :boolean

property BeforeEditScriptsEnabled :boolean

property BeforeInsertScriptsEnabled :boolean

property BeforePostScriptsEnabled :boolean

property BeforeScrollScriptsEnabled :boolean

property OnValidateScriptsEnabled :boolean

property OnChangeScriptsEnabled :boolean

property OnChangeTimeStampEnabled :boolean

procedure DisableScripts

procedure DisableLayerEventScripts

procedure DisableAfterCancelScripts

procedure DisableAfterDeleteScripts

procedure DisableAfterEditScripts

procedure DisableAfterInsertScripts

procedure DisableAfterPostScripts

procedure DisableAfterScrollScripts

procedure DisableBeforeCancelScripts

procedure DisableBeforeDeleteScripts

procedure DisableBeforeEditScripts

procedure DisableBeforeInsertScripts

procedure DisableBeforePostScripts

procedure DisableBeforeScrollScripts

procedure DisableOnValidateScripts

procedure DisableOnChangeScripts

procedure DisableOnChangeTimeStamp

procedure EnableScripts

procedure EnableLayerEventScripts

procedure EnableAfterCancelScripts

procedure EnableAfterDeleteScripts

procedure EnableAfterEditScripts

procedure EnableAfterInsertScripts

procedure EnableAfterPostScripts

procedure EnableAfterScrollScripts

procedure EnableBeforeCancelScripts

procedure EnableBeforeDeleteScripts

procedure EnableBeforeEditScripts

procedure EnableBeforeInsertScripts

procedure EnableBeforePostScripts

procedure EnableBeforeScrollScripts

procedure EnableOnValidateScripts

procedure EnableOnChangeScripts

procedure EnableOnChangeTimeStamp

function RunSynchronization(const AutoStart\_ :boolean; const PlotID\_ :variant; const RunBeforeScript\_,RunAfterScript\_ :boolean) :boolean

function RunSynchronizationExt(const Params\_ :string) :boolean

procedure AdjustOnLineCalliperConnection

## SCREEN

function GoToLayerPage(const LayerName\_ :string) :boolean

function IsLayerPageVisible(const LayerName\_ :string) :boolean

## FILES

function DeleteFile(const FilePath :string) :boolean

function CreateDir(const Path\_ :string) :boolean

function DeleteDir(const Path\_ :string) :boolean

function GetFilesOfDir(const Path\_ :string) :string *(from version X7)*

procedure OpenFileByWindowsAssociation(const Filename :string) *(from version X7)*

procedure SaveStringToIniFile(const section, key :string; value :variant; const INIFileName :string)

procedure SaveIntegerToIniFile(const section, key :string; value :variant; const INIFileName :string)

procedure SaveDoubleToIniFile(const section, key :string; value :variant; const INIFileName :string)

procedure SaveBooleanToIniFile(const section, key :string; value :variant; const INIFileName :string)

function LoadStringFromIniFile(const section, key :string; const INIFileName :string) :variant

function LoadIntegerFromIniFile(const section, key :string; const INIFileName :string) :variant

function LoadDoubleFromIniFile(const section, key :string; const INIFileName :string) :variant

function LoadBooleanFromIniFile(const section, key :string; const INIFileName :string) :variant

function ExtractFilename (const Filename : string) :string *(from version X7)*

function ExtractFilePath (const Filename : string) :string *(from version X7)*

function ExtractFileExt (const Filename : string) :string *(from version X7)*

function FileExists() (const Filename : string) : boolean *(from version X7)*

function DirectoryExists() (const Directory: string): boolean; *(from version X7)*

## DATABASE

procedure Refresh

procedure EnableChachedUpdates

procedure ApplyUpdates

procedure StartTransaction

procedure Commit

procedure CommitRetaining

procedure Rollback

procedure RollbackRetaining

function ClearAllFilters :boolean

function GetQueryResult(const SQL\_ :string) :TTableWrapper

procedure PerformQuery(const SQL\_ :string)

function OpenTable(const Tablename\_ :string) :TTableWrapper

procedure SaveRecordToDeletedLog(const PlotsGUID\_,LayerName\_ :string; const TimeStamp\_ :double; const GUID\_ :string)

procedure SaveRecordsToDeletedLog(tab :TTableWrapper; const PlotsGUID\_, LayerName\_ :string; const TimeStamp\_ :double)

procedure BackupDatabase(const FileCaption\_ :string)

procedure PostAllData(const ReturnToEditMode\_ :boolean)

procedure ReloadLookupLists

procedure PostProjectAndLookupTablesToSynchroJournal

procedure PostLookupTablesToSynchroJournal

procedure PostSelectedLookupTablesToSynchroJournal(const LookupAttributes\_,XmlFileName\_,Keywords\_,TextMessage\_ :string)

procedure PostFileToSynchroJournal(const Filename\_,Folder\_,Keywords\_,TextMessage\_ :string; const RequireFieldMapRestart\_ :boolean)

function ChangedAfterLastSynchronization(const PlotID\_ :variant) :boolean

## NAVIGATOR & DQT & EXTENSIONS

function RunFieldMapExtension(const ExtensionName\_ :string) :integer

procedure HideExtensionMenu(const ExtensionName\_ :string)

procedure ShowExtensionMenu(const ExtensionName\_ :string)

procedure RunDatabaseQueryTool(const Filename\_ :string; const AllowEditing\_,AllowNavigator\_,AllowStatistics\_ :boolean)

procedure AddItemToNavigatorXML(DataSet\_ :TTableWrapper; const ReferenceLayerName\_,XMLfilename\_,Caption\_,Hint\_ :string)

property DBNavigatorVisible :boolean

property DBNavigatorConfirmDelete :boolean

function GetSpeciesGroupID (const SpeciesCode :variant; const SpeciesGroupsXML :string) :variant *(from version X7)*

## GIS

function GetPolygonIntersectArea\_m2(const LayerName1\_ :string; const ID1\_ :integer; const LayerName2\_ :string; const ID2\_ :integer) :variant

function OverlayPolygonLayers(const LayerName1\_,LayerName2\_ :string) :TMultiVariantListWrapper

function TransferAttributeValuesFromPolygonsToPointsOrCentroids(const TargetPointOrPolygonLayerName\_ :string; const TargetAttributesName\_ :string; const SourcePolygonLayerName\_ :string; const SourceAttributesName\_ :string) :integer

function GetPoint(out X\_m,Y\_m,Z\_m :double) :boolean

procedure DrawText(const X\_m,Y\_m :double; const Text\_ :string; const HorizontalAlignment\_,VerticalAlignment\_ :integer; const Color\_,FontName\_ :string; const Size\_ :double; const Rotation\_deg :double; const Bold\_,Italic\_,Underline\_ :boolean)

procedure DrawPoint(const X\_m,Y\_m :double; const PointStyle\_,PointSize\_ :integer; const PointColor\_ :string; const CharacterIndex\_ :integer; const FontName\_ :string)

procedure DrawLine(const XY\_ :string; const LineStyle\_,LineWidth\_ :integer; const LineColor\_ :string)

procedure DrawPolygon(const XY\_ :string; const LineWidth\_ :integer; const LineColor\_ :string; const FillStyle\_ :integer; const FillColor\_ :string)

procedure DrawCircle(const CenterX\_m,CenterY\_m,Radius\_m :double; const LineStyle\_,LineWidth\_ :integer; const LineColor\_ :string; const FillStyle\_ :integer; const FillColor\_ :string)

procedure DrawRectangle(const Left\_m,Right\_m,Bottom\_m,Top\_m :double; const LineStyle\_,LineWidth\_ :integer; const LineColor\_ :string; const FillStyle\_ :integer; const FillColor\_ :string)

procedure FlashPolygon(const XY\_ :string; const FlashCount\_ :integer)

procedure FlashLine(const XY\_ :string; const FlashCount\_ :integer)

procedure FlashPoint(const X\_,Y\_ :double; const FlashCount\_ :integer)

function SelectedPointCount :integer

function SelectedPointIDs :string

function SelectedLineCount :integer

function SelectedLineIDs :string

procedure GetCurrentPosition(out X\_,Y\_,Z\_ :variant)

procedure SetCurrentPosition(const X\_,Y\_,Z\_ :variant) *(from version X7)*

procedure SetBasicExtent(const Left,Bottom,Right,Top :double) *(from version X7)*

procedure AddReferencePoint(const X\_,Y\_ :double; const Caption\_ :string; const PoleLength\_cm\_ :integer) *(from version X7)*

property InvertColors :boolean *(from version X7)*

function FindTargetHeight(const HorizontalDistance\_m,InclinationToBase\_deg,TargetHeight\_m,HeightTolerance\_m :variant) :boolean *(from version X7)*

procedure SetCircularPlotBoundary(const Radius\_m,CenterX\_m,CenterY\_m :double)

function TrimLineByPlotBoundary(var X1\_m,Y1\_m,X2\_m,Y2\_m :double) :integer

procedure RefreshMap

procedure RefreshMapRectangle(const Left\_m,Right\_m,Bottom\_m,Top\_m :double)

procedure RefreshTrackingLayer(const Erase\_ :boolean)

procedure RefreshTrackingLayerRectangle(const Erase\_ :boolean; const Left\_m,Right\_m,Bottom\_m,Top\_m :double)

function ScreenMilimeters2MapDistance\_m(const Distance\_mm :double) :double

function MapToScreen(const X\_m,Y\_m :double; out X\_,Y\_ :integer) :boolean

procedure SaveShapesToBlobs

procedure SaveShapesOfCurrentPlotToBlobs

procedure SaveShapesOfAllPlotsToBlobs

function ProjectionToWGS84 (const Northing\_m,Easting\_m :double; out Lat\_deg,Lon\_deg :double) :boolean *(from version X7)*

function WGS84ToProjection (const Lat\_deg,Lon\_deg :double; out Northing\_m,Easting\_m :double) :boolean *(from version X7)*

## EXTRAS

function EditSpeciesGroups(const SpecGroupsXmlBlobLayerName\_, SpecGroupsXmlBlobAttribName\_, SpeciesLookupTableName\_, AlternativeSpecGroupsFile\_, Memo\_ :string; const EnableGroupChanging\_ :boolean; const XmlEncoding\_ :string) :boolean

function EditSpeciesGroupsUsingDesigner(const SpecGroupsXmlBlobLayerName\_, SpecGroupsXmlBlobAttribName\_, SpeciesLkpTableName\_, SpeciesLkpValueAttribName\_, TreesXML\_, TreeSpeciesAttribName\_ :string; const FingerControl\_, EnableGroupChanging\_ :boolean) :boolean; *(from version X7)*

procedure ShowHeightCandidates (const TreeLayerName\_, SampleTreeLayerName\_, SpeciesAttribName\_, DbhAttribName\_,HeightAttribName\_, sSpeciesLkpTableName\_, WHERE\_ :string) *(from version X7)*

## STEM PROFILE

function LoadGlobalStemProfileModels(const Filename\_ :string) :boolean

function LoadGlobalStemProfileModelsForRegion(const Filename\_ :string; const RegionID\_ :variant) :boolean

function GlobalStemProfileModelsFilename :string

procedure FreeGlobalStemProfileModels

# LISTS

## STRINGLIST

procedure Free

property Duplicates :boolean

property Sorted :boolean

property CaseSensitive :boolean

procedure BeginUpdate

procedure EndUpdate

function Add(const S :string) :integer

procedure AddStrings(Strings :TStringListWrapper)

procedure Assign(Source :TStringListWrapper)

procedure Clear

procedure Delete(Index :integer)

function Count :integer

procedure Exchange(Index1, Index2 :integer)

procedure Move(CurIndex, NewIndex :integer)

function Find(const S :string; var Index :integer) :boolean

function IndexOf(const S :string) :integer

procedure Insert(Index :integer; const S :string)

procedure Sort

procedure LoadFromFile(const FileName :string)

procedure SaveToFile(const FileName :string)

property Strings[Index :integer] :string

function Text :string

property Delimiter :char

property DelimitedText :string

property NameValueSeparator :char

function IndexOfName(const Name :string) :integer

property Name[Index :integer] :string

property Value[const Name :string] :string

property AsVarArray :variant *(from version X7)*

## INTEGERLIST

procedure Free

procedure Add(const Value :integer)

procedure Clear

procedure Assign(Source :TintegerListWrapper)

procedure Delete(Index :integer)

function Count :integer

function IndexOf(const Value :integer) :integer

procedure SortAscending

procedure SortDescending

procedure SortAscendingAndRemoveDuplicates

property Value[Index :integer] :integer

property Delimiter :char

property DelimitedText :string

property AsVarArray :variant *(from version X7)*

## VARIANTLIST

procedure Free

procedure Add(const Value :variant)

procedure Clear

procedure Assign(Source :TVariantListWrapper)

procedure Delete(Index :integer)

function Count :integer

function IndexOf(const Value :variant) :integer

property Value[Index :integer] :variant

property ValueAsFloat[Index :Integer]:double

property ValueAsInteger[Index :Integer]:integer

property ValueAsString[Index :Integer]:string

property AsVarArray :variant *(from version X7)*

## MULTIVARIANTLIST

procedure Free

procedure Add(const Values :array of variant)

procedure Clear

procedure Assign(Source :TVariantListWrapper)

procedure Delete(Index :integer)

function Count :integer

property Value[ItemIndex,ValueIndex :Integer]:variant

property ValueAsFloat[ItemIndex,ValueIndex :Integer]:double

property ValueAsInteger[ItemIndex,ValueIndex :Integer]:integer

property ValueAsString[ItemIndex,ValueIndex :Integer]:string

procedure SortAscending(const ValueIndex :integer) *(from version X7)*

procedure SortDescending(const ValueIndex :integer) *(from version X7)*

procedure SortMultiAscending(const ValueIndex :array of integer) *(from version X7)*

procedure SortMultiDescending(const ValueIndex :array of integer) *(from version X7)*

function IndexOf(const ValueIndex,Value: Integer): Integer *(from version X7)*

## VARVARIANTLIST

procedure Free

function ListCount :integer

function ValueCount(const ListIndex :integer) :integer

function AddList :TVariantListWrapper

procedure Clear

procedure Assign(Source:TVarVariantListWrapper)

function List(const ListIndex :integer) :TVariantListWrapper

property Value[ListIndex,ValueIndex :Integer]:variant

property ValueAsFloat[ItemIndex,ValueIndex :Integer]:double

property ValueAsInteger[ItemIndex,ValueIndex :Integer]:integer

property ValueAsString[ItemIndex,ValueIndex :Integer]:string

## OBJECTLIST *(from version X7)*

procedure Free

procedure Clear

property OwnsObjects :boolean

function Count :integer

procedure Add(Object\_ :TObjectWrapper)

property Item[Index :Integer]: TObjectWrapper

# EXCELWRAPPER

constructor Create

procedure Free

procedure FillColor(const row1,col1,row2,col2 :integer; const Color\_ :integer)

procedure MergeCells(const row1,col1,row2,col2 :integer)

procedure SetTextOrientation(const row1,col1,row2,col2 :integer;const Degrees\_ :integer)

procedure Align(const row1,col1,row2,col2 :integer; const HorizontalAlignment\_,VerticalAlignment\_ :string) {LEFT,RIGHT,CENTER; TOP,BOTTOM,CENTER}

procedure DrawBorders(const row1,col1,row2,col2 :integer; const Borders\_,BorderStyle\_ :string); {LEFT,RIGHT,TOP,BOTTOM; HAIR,THIN,MEDIUM,THICK,DOUBLE}

procedure FormatCells(const row1,col1,row2,col2 :integer;const FontSize\_ :integer;const FontStyle\_ :string;const DecimalPlaces\_ :integer;const Indent\_ :boolean;const HorizontalAlignment\_,VerticalAlignment\_ :string)

procedure SetOptimalColWidth(const col1,col2 :integer; const Adjustment\_ :double{e.g. 1.05});

procedure SetColWidth(const col1,col2 :integer; const Width\_ :integer);

procedure SetRowHeight(const row1,row2 :integer; const Height\_ :integer);

procedure SetColWidth\_mm(const col1,col2 :integer; const Width\_mm\_ :double)

procedure SetRowHeight\_mm(const row1,row2 :integer; const Height\_mm\_ :double)

function GetTextWidth(const Text\_ :string; const Format\_ :string) :integer

function GetTextHeight(const Format\_ :string) :integer

function GetCellWidth(const Row,Col :integer; const Format\_ :string) :integer

procedure SetPaperSize(const PaperSize\_ :string)

property Landscape :boolean

procedure SetHeaderAndFooter(const HeaderLeft\_,HeaderCenter\_,HeaderRight\_,FooterLeft\_,FooterRight\_ :string)

procedure SetTitleRows(const row1,row2 :integer)

procedure SetTitleRowsAndCols(const row1,row2,col1,col2 :integer)

procedure CreateHeaderOfTableContent

procedure CreateTableOfContent(lst :TStringListWrapper) {list of task caption|sheet name}

procedure AddBackToTableOfContent(const Row\_,RowNoInContent\_,nCol\_ :integer)

procedure SetPrintArea(const row1,col1,row2,col2 :integer)

function RowCount :integer *(from version X7)*

function ColCount :integer *(from version X7)*

procedure SetValue(const row,col :integer;const Value :variant;const Format\_,Borders\_,BorderStyle\_ :string)

procedure SetValueAndMerge(const row1,col1,row2,col2 :integer;const Value :variant;const Format\_,Borders\_,BorderStyle\_ :string)

procedure SetFormula(const row,col :integer; const Formula,Format\_,FormulaReferenceStyle\_ :string) {A1 nebo R1C1}

function GetCellValue(const row,col :integer) :variant

procedure CreateFile(const Filename\_ :string)

procedure OpenFile(const Filename\_ :string)

property Filename :string

procedure AddSheet(const SheetName\_ :string; const OverwriteSheet\_ :boolean)

procedure MakeSheetActive(const SheetName\_ :string) *(from version X7)*

procedure ClearSheet(const SheetName\_ :string) *(from version X7)*

procedure CloseAndSave(const OpenExcel\_ :boolean) *(from version X7)*

procedure OpenInExcel *(from version X7)*

# POINTWRAPPER

constructor Create(const X\_,Y\_ :extended; const ID\_ :integer)

procedure Free

property ID :integer

property X :extended

property Y :extended

# LINEWRAPPER

constructor Create(const ID\_ :integer)

procedure Free

property ID :integer

procedure AddPoint(const X\_,Y\_ :extended)

property PointX[IndexOfPoint :integer] :extended

property PointY[IndexOfPoint :integer] :extended

function Length :double

function GetParallelLine(const Distance\_m :double) :TLineWrapper

LoadFromPolygon|procedure LoadFromPolygon(Polygon\_ :TPolygonWrapper); *(from version X7)*

# POLYGONWRAPPER

constructor Create(const ID\_ :integer)

procedure Free

property ID :integer

procedure AddRing(Line\_ :TLineWrapper; const AsMainRing\_ :boolean)

procedure AddPointToRing(const Ring\_ :integer; const X\_,Y\_ :double)

procedure LoadFromVarArray(Data\_ :variant) *(from version X7)*

procedure AddPolygon(Polygon\_ :TPolygonWrapper) *(from version X7)*

property RingRotation[Ring\_ :integer] :integer read GetRingRotation write SetRingRotation {+1 clockwise, -1 counterclockwise, 0 unknown} *(from version X7)*

function RingCount :integer *(from version X7)*

function MainRingCount :integer

property PointCount[const Ring\_ :integer] :integer

function TotalPointCount :integer

property PointX[Ring\_,Point\_ :integer] :extended

property PointY[Ring\_,Point\_ :integer] :extended

function Area :extended

function PointInPolygon(const X\_,Y\_ :double; const IncludingPolygonBorder\_ :boolean) :boolean

procedure CreateCircle(const CenterX\_m,CenterY\_m,Radius\_m :double)

function CoordinatesAsText :string

function CreateSamplePoints(const Npoints :integer; const BufferDist\_m :double; out FinalDist\_m :double) :TPointsWrapper *(from version X7)*

# SPECIESGROUPS

constructor Create

procedure Free

procedure Assign(Source :TSpeciesGroupsWrapper)

property SpeciesLookupTableName :string

function LoadFromTable(TableOrLayer :TObjectWrapper; const FieldName\_ :string) :boolean

function SaveToTable(TableOrLayer :TObjectWrapper; const FieldName\_ :string) :boolean

function LoadFromXML(const XmlFilename\_ :string) :boolean

function SaveToXML(const XmlFilename\_ :string) :boolean

function GetSpeciesCompositionInfo(TableOrLayer :TObjectWrapper; const SpeciesAttributeName\_ :string) :string;

function UpdateSpeciesGroupsLookupList(const LayerName\_,AttributeName\_ :string) :boolean

function PublishClassificationToMemTable :TTableWrapper

function EditClassification(const EnableGroupChanging\_ :boolean; const Memo\_ :string) :boolean

property ID :variant

property Name :string

property FullName :string

function SpecGroupsCount

function SpecGroupName(const SpecGroupID\_ :variant) :string

function SpecGroupFullName(const SpecGroupID\_ :variant) :string

function SpeciesListForGroup(const SpecGroupID\_ :variant) :TVariantListWrapper *(from version X7)*

property SpecGroupID[Index :integer] :variant *(from version X7)*

function SpecGroupIDForSpecies(const Species\_ :variant) :variant

function ClassifyTable(TableOrLayer :TObjectWrapper; const SpeciesAttributeName\_,SpecGroupAttributeName\_ :string) :boolean

function ClassifyLayerUsingSQL(const LayerName\_,SpeciesAttributeName\_,SpecGroupAttributeName\_,WHERE\_ :string) :boolean

function UngroupedSpeciesCount :integer

# STEM PROFILE

procedure Free

procedure Assign(Source :TStemProfileWrapper)

procedure ClearContent

procedure ResetData

property ID :variant

property SpeciesCode :variant

property SpeciesName :string

property DBH\_mm :variant

property TreeHeight\_m :variant

procedure AddPoint(const H\_m :double;const D\_mm :integer)

procedure DeletePointAtHeight(const H\_m :double) *(from version X7)*

procedure DeletePointsBelowHeight(const H\_m :double) *(from version X7)*

procedure AddTop *(from version X7)*

procedure ResetStemBaseAndTop *(from version X7)*

function Count

procedure SortByHeight

function Diameter\_mm(Index :integer; UnderBark\_ :boolean) :integer *(from version X7)*

property Height\_m[Index :integer] :double *(from version X7)*

function Diameter2\_mm(Index :integer; UnderBark\_ :boolean) :integer *(from version X7)*

property SD\_m[Index :integer] :variant *(from version X7)*

property Ticks[Index :integer] :variant *(from version X7)*

function LoadBarkThicknessXML(const XmlFilename\_ :string) :boolean

function LoadParametersFromStandModelXML(const XmlFilename\_ :string) :boolean

function LoadDataFromXML(XML\_ :string) :boolean

function ExportStemPointsToXML :string *(from version X7)*

function ApplyGlobalModel :boolean

function DiamForHeightByModel\_mm(const Height\_m\_ :extended; const UnderBark\_ :boolean) :extended

function HeightForDiamByModel\_m(const Diam\_mm\_ :extended; const UnderBark\_ :boolean) :extended

property RoughBarkHeight\_m :variant

function DoubleBarkThickness\_mm (const DiameterOverBark\_mm,Height\_m :variant) :variant *(from version X7)*

property RoughBarkSmoothTransition :boolean *(from version X7)*

property StumpDiameter\_mm :variant

property StumpHeightCalcMethod :variant '1/3D' or '1/100H'

property StumpHeight\_m :extended

procedure AddDiameterReduction(const Height\_m\_,Reduction\_ :extended) *(from version X7)*

procedure ResetDiameterReductions *(from version X7)*

procedure SaveDiameterReductions *(from version X7)*

procedure RestoreDiameterReductions *(from version X7)*

function CumulativeDiameterReduction(const Height\_m\_ :extended) :extended *(from version X7)*

procedure ExpandStemDataByDiameterReductions *(from version X7)*

function Volume\_m3(const LowerHeight\_m\_,UpperHeight\_m\_ :extended; const UnderBark\_ :boolean) :extended

function TotalVolume\_m3(const UnderBark\_ :boolean) :extended

function StemVolume\_m3(const UnderBark\_ :boolean) :extended

function MerchantableVolume\_m3(const LimitDiam\_mm\_ :integer =70; const UnderBark\_ :boolean) :extended

function ParametrizeModel :boolean

function ModelAvailable :boolean

property Param[Index :integer] :variant {1..3}

procedure AdjustForKnownVolume(const TargetVolume\_m3,LowerHeight\_m\_,UpperHeight\_m\_ :extended; const UnderBark\_ :boolean)

procedure AdjustForKnownMerchantableVolume(const TargetVolume\_m3 :Extended;const LimitDiam\_mm\_ :integer;const UnderBark\_ :boolean)

procedure ShowStemProfile

procedure ShowStemProfileComaprison(AnotherStemProfile\_ :TStemProfileWrapper)

procedure ShowStemProfileWithPoints(DHpoints\_mmm{H[m],D[mm]} :TMultiVariantListWrapper)

procedure ShowStemProfileExt(const const Options\_ :string; {e.g. MODEL;POINTS;ID;SPECIES} AnotherStemProfile\_ :TStemProfileWrapper; DHpoints\_mmm :TMultiVariantListWrapper)

procedure RecalcDiametersUsingCurrentDendroscope

# VOLUMEMODELSWRAPPER *(from version X7)*

constructor Create

procedure Free

function LoadFromXML(const XmlFilename\_ :string) :boolean

property UseSpeciesGroups :boolean

function Volume\_m3(const ModelName :string; const Species, DBH\_cm, H\_m, D03\_cm, BreakHeight\_m :variant) :variant;