

MBS Real Studio XMP Plugin Documentation

Christian Schmitz

May 15, 2012

0.1 Introduction

This is the PDF version of the documentation for the Real Studio Plug-in from Monkeybread Software Germany. Plugin part: MBS Real Studio XMP Plugin

0.2 Content

- 1 List of all topics 3
- 2 All items in this plugin 15
- 3 List of all classes 127

Chapter 1

List of Topics

• 2 XMP	15
– 2.10 class XMPMetaMBS	57
* 2.10.1 AppendArrayItem(schemaNS as string, arrayName as string, arrayOptions as integer, itemValue as string, options as integer=0)	58
* 2.10.1 ApplyTemplate(WorkingXMP as XMPMetaMBS, template as XMPMetaMBS, actions as integer)	58
* 2.10.1 CatenateArrayItems(schemaNS as string, arrayName as string, separator as string, quotes as string, options as integer) as string	59
* 2.10.1 Clone as XMPMetaMBS	59
* 2.10.1 ComposeArrayItemPath(schemaNS as string, arrayName as string, itemIndex as integer) as string	59
* 2.10.1 ComposeFieldSelector(schemaNS as string, arrayName as string, fieldNS as string, fieldName as string, fieldValue as string) as string	60
* 2.10.1 ComposeLangSelector(schemaNS as string, arrayName as string, langName as string) as string	61
* 2.10.1 ComposeQualifierPath(schemaNS as string, structName as string, qualNS as string, qualName as string) as string	61
* 2.10.1 ComposeStructFieldPath(schemaNS as string, structName as string, fieldNS as string, fieldName as string) as string	62
* 2.10.1 Constructor	62
* 2.10.1 Constructor(data as memoryblock, Offset as integer, Size as integer)	62
* 2.10.1 Constructor(data as string)	63
* 2.10.1 ConvertFromBool(value as boolean) as string	63
* 2.10.1 ConvertFromDate(value as XMPDateTimeMBS) as string	63
* 2.10.1 ConvertFromFloat(value as double, format as string) as string	63
* 2.10.1 ConvertFromInt(value as integer, format as string) as string	63

* 2.10.1 ConvertFromInt64double(value as double, format as string) as string	64
* 2.10.1 ConvertToBool(value as string) as boolean	64
* 2.10.1 ConvertToDate(value as string) as XMPDateTimeMBS	64
* 2.10.1 ConvertToFloat(value as string) as double	64
* 2.10.1 ConvertToInt(value as string) as integer	64
* 2.10.1 ConvertToInt64double(value as string) as double	65
* 2.10.1 CountArrayItems(schemaNS as string, arrayName as string) as integer	65
* 2.10.1 CurrentDateTime as XMPDateTimeMBS	65
* 2.10.1 DecodeFromBase64(text as string) as string	65
* 2.10.1 DeleteArrayItem(schemaNS as string, arrayName as string, itemIndex as integer)	65
* 2.10.1 DeleteLocalizedText(schemaNS as string="", altTextName as string="", genericLang as string="", specificLang as string="")	66
* 2.10.1 DeleteNamespace(namespaceURI as string)	66
* 2.10.1 DeleteProperty(schemaNS as string, propName as string)	66
* 2.10.1 DeleteQualifier(schemaNS as string, structName as string, qualNS as string, qualName as string)	67
* 2.10.1 DeleteStructField(schemaNS as string, structName as string, fieldNS as string, fieldName as string)	67
* 2.10.1 DoesArrayItemExist(schemaNS as string, arrayName as string, itemIndex as integer) as boolean	68
* 2.10.1 DoesPropertyExist(schemaNS as string, propName as string) as boolean	68
* 2.10.1 DoesQualifierExist(schemaNS as string, structName as string, qualNS as string, qualName as string) as boolean	69
* 2.10.1 DoesStructFieldExist(schemaNS as string, structName as string, fieldNS as string, fieldName as string) as boolean	69
* 2.10.1 DumpNamespaces(output as XMPTextOutputMBS) as integer	70
* 2.10.1 DumpObject(output as XMPTextOutputMBS) as integer	70
* 2.10.1 DuplicateSubtree(dest as XMPMetaMBS, sourceNS as string, sourceRoot as string, destNS as string="", destRoot as string="", options as integer=0)	70
* 2.10.1 EncodeToBase64(text as string) as string	71
* 2.10.1 Erase	71
* 2.10.1 GetArrayItem(schemaNS as string, arrayName as string, itemIndex as integer, byref itemValue as string, byref options as integer) as boolean	71
* 2.10.1 GetLocalizedText(schemaNS as string, altTextName as string, genericLang as string, specificLang as string, byref actualLang as string, byref itemValue as string, byref options as integer) as boolean	72
* 2.10.1 GetNamespacePrefix(namespaceURI as string, byref namespacePrefix as string) as boolean	73
* 2.10.1 GetNamespaceURI(namespacePrefix as string, byref namespaceURI as string) as boolean	73
* 2.10.1 GetProperty(schemaNS as string, propName as string, byref propValue as string, byref options as integer) as boolean	74
* 2.10.1 GetPropertyBoolean(schemaNS as string, propName as string, byref propValue as boolean) as boolean	74

- * 2.10.1 GetPropertyBoolean(schemaNS as string, propName as string, byref propValue as boolean, byref options as integer) as boolean 75
- * 2.10.1 GetPropertyDate(schemaNS as string, propName as string, byref propValue as XMP-DateTimeMBS, byref options as integer) as boolean 75
- * 2.10.1 GetPropertyFloat(schemaNS as string, propName as string, byref propValue as double) as boolean 76
- * 2.10.1 GetPropertyFloat(schemaNS as string, propName as string, byref propValue as double, byref options as integer) as boolean 76
- * 2.10.1 GetPropertyInt64Double(schemaNS as string, propName as string, byref propValue as double, byref options as integer) as boolean 77
- * 2.10.1 GetPropertyInteger(schemaNS as string, propName as string, byref propValue as integer) as boolean 77
- * 2.10.1 GetPropertyInteger(schemaNS as string, propName as string, byref propValue as integer, byref options as integer) as boolean 78
- * 2.10.1 GetPropertyInteger64(schemaNS as string, propName as string, byref propValue as Int64, byref options as integer) as boolean 79
- * 2.10.1 GetQualifier(schemaNS as string, propName as string, qualNS as string, qualName as string, byref qualValue as string, byref options as integer) as boolean 79
- * 2.10.1 GetStructField(schemaNS as string, structName as string, fieldNS as string, fieldName as string, byref itemValue as string, byref options as integer) as boolean 80
- * 2.10.1 GetVersionInfo as XMPVersionInfoMBS 81
- * 2.10.1 GlobalOptions as integer 81
- * 2.10.1 Iterator(schemaNS as string, propName as string, options as integer) as XMPIteratorMBS 81
- * 2.10.1 MergeFromJPEG(extendedXMP as XMPMetaMBS) 82
- * 2.10.1 Name as string 82
- * 2.10.1 PackageForJPEG(byref standardXMP as string, byref extendedXMP as string, byref extendedDigest as string) 82
- * 2.10.1 ParseFromBuffer(buffer as string, options as integer=0) 83
- * 2.10.1 RegisterNamespace(namespaceURI as string, suggestedPrefix as string, byref registeredPrefix as string) as boolean 84
- * 2.10.1 RemoveProperties(schemaNS as string="", propName as string="", options as integer=0) 85
- * 2.10.1 SendAssertNotify(message as string) 85
- * 2.10.1 SeparateArrayItems(schemaNS as string, arrayName as string, options as integer, cat-edStr as string) 85
- * 2.10.1 SerializeToBuffer(options as integer, padding as integer, newline as string, indent as string="", baseIndent as integer=0) as string 86
- * 2.10.1 SerializeToBuffer(options as integer=0, padding as integer=0) as string 87
- * 2.10.1 SetArrayItem(schemaNS as string, arrayName as string, itemIndex as integer, itemValue as string, options as integer=0) 88
- * 2.10.1 SetLocalizedText(schemaNS as string, altTextName as string, genericLang as string, specificLang as string, itemValue as string, options as integer=0) 88
- * 2.10.1 SetProperty(schemaNS as string, propName as string, propValue as string, options as integer=0) 89

* 2.10.1 SetPropertyBoolean(schemaNS as string, propName as string, propValue as boolean)	90
* 2.10.1 SetPropertyBoolean(schemaNS as string, propName as string, propValue as boolean, options as integer)	90
* 2.10.1 SetPropertyDate(schemaNS as string, propName as string, propValue as XMPDate-TimeMBS, options as integer=0)	91
* 2.10.1 SetPropertyFloat(schemaNS as string, propName as string, propValue as double)	91
* 2.10.1 SetPropertyFloat(schemaNS as string, propName as string, propValue as double, options as integer)	91
* 2.10.1 SetPropertyInt64Double(schemaNS as string, propName as string, propValue as double, options as integer=0)	92
* 2.10.1 SetPropertyInteger(schemaNS as string, propName as string, propValue as integer)	92
* 2.10.1 SetPropertyInteger(schemaNS as string, propName as string, propValue as integer, options as integer)	93
* 2.10.1 SetPropertyInteger64(schemaNS as string, propName as string, propValue as Int64, options as integer=0)	93
* 2.10.1 SetQualifier(schemaNS as string, propName as string, qualNS as string, qualName as string, qualValue as string, options as integer=0)	94
* 2.10.1 SetStructField(schemaNS as string, structName as string, fieldNS as string, fieldName as string, fieldValue as string, options as integer=0)	94
* 2.10.1 Sort	95
* 2.10.2 kAllowCommas = & h10000000	95
* 2.10.2 kArrayLastItem = -1	96
* 2.10.2 kDeleteEmptyValues = 4	96
* 2.10.2 kDeleteExisting = & h20000000	96
* 2.10.2 kDoAllProperties = 1	96
* 2.10.2 kEncodeUTF16Big = 2	96
* 2.10.2 kEncodeUTF16Little = 3	97
* 2.10.2 kEncodeUTF32Big = 4	97
* 2.10.2 kEncodeUTF32Little = 5	97
* 2.10.2 kEncodeUTF8 = 0	97
* 2.10.2 kEncodingMask = & h7	97
* 2.10.2 kExactPacketLength = & h200	98
* 2.10.2 kImplReservedMask = & h70000000	98
* 2.10.2 kIncludeAliases = & h800	98
* 2.10.2 kIncludeThumbnailPad = & h100	98
* 2.10.2 kInsertAfterItem = & h8000	98
* 2.10.2 kInsertBeforeItem = & h4000	98
* 2.10.2 kIterAliases = 1	99
* 2.10.2 kIterClassMask = & hFF	99
* 2.10.2 kIterIncludeAliases = & h800	99
* 2.10.2 kIterJustChildren = & h100	99
* 2.10.2 kIterJustLeafName = & h400	99

* 2.10.2 kIterJustLeafNodes = & h200	100
* 2.10.2 kIterNamespaces = 2	100
* 2.10.2 kIterOmitQualifiers = & h1000	100
* 2.10.2 kIterProperties = 0	100
* 2.10.2 kIterSkipSiblings = 2	100
* 2.10.2 kIterSkipSubtree = 1	101
* 2.10.2 kLittleEndianBit = 1	101
* 2.10.2 kNoOptions = 0	101
* 2.10.2 kNS_ AdobeStockPhoto = "http://ns.adobe.com/StockPhoto/1.0/"	101
* 2.10.2 kNS_ ASF = "http://ns.adobe.com/asf/1.0/"	101
* 2.10.2 kNS_ CameraRaw = "http://ns.adobe.com/camera-raw-settings/1.0/"	101
* 2.10.2 kNS_ CreatorAtom = "http://ns.adobe.com/creatorAtom/1.0/"	102
* 2.10.2 kNS_ DC = "http://purl.org/dc/elements/1.1/"	102
* 2.10.2 kNS_ DICOM = "http://ns.adobe.com/DICOM/"	102
* 2.10.2 kNS_ DM = "http://ns.adobe.com/xmp/1.0/DynamicMedia/"	102
* 2.10.2 kNS_ EXIF = "http://ns.adobe.com/exif/1.0/"	102
* 2.10.2 kNS_ EXIF_ Aux = "http://ns.adobe.com/exif/1.0/aux/"	102
* 2.10.2 kNS_ IPTCCore = "http://iptc.org/std/Iptc4xmpCore/1.0/xmlns/"	103
* 2.10.2 kNS_ JP2K = "http://ns.adobe.com/jp2k/1.0/"	103
* 2.10.2 kNS_ JPEG = "http://ns.adobe.com/jpeg/1.0/"	103
* 2.10.2 kNS_ PDF = "http://ns.adobe.com/pdf/1.3/"	103
* 2.10.2 kNS_ PDFA_ Extension = "http://www.aiim.org/pdfa/ns/extension/"	103
* 2.10.2 kNS_ PDFA_ Field = "http://www.aiim.org/pdfa/ns/field# "	103
* 2.10.2 kNS_ PDFA_ ID = "http://www.aiim.org/pdfa/ns/id/"	104
* 2.10.2 kNS_ PDFA_ Property = "http://www.aiim.org/pdfa/ns/property# "	104
* 2.10.2 kNS_ PDFA_ Schema = "http://www.aiim.org/pdfa/ns/schema# "	104
* 2.10.2 kNS_ PDFA_ Type = "http://www.aiim.org/pdfa/ns/type# "	104
* 2.10.2 kNS_ PDFX = "http://ns.adobe.com/pdfx/1.3/"	104
* 2.10.2 kNS_ PDFX_ ID = "http://www.npes.org/pdfx/ns/id/"	104
* 2.10.2 kNS_ Photoshop = "http://ns.adobe.com/photoshop/1.0/"	105
* 2.10.2 kNS_ PNG = "http://ns.adobe.com/png/1.0/"	105
* 2.10.2 kNS_ PSAlbum = "http://ns.adobe.com/album/1.0/"	105
* 2.10.2 kNS_ RDF = "http://www.w3.org/1999/02/22-rdf-syntax-ns# "	105
* 2.10.2 kNS_ SWF = "http://ns.adobe.com/swf/1.0/"	105
* 2.10.2 kNS_ TIFF = "http://ns.adobe.com/tiff/1.0/"	105
* 2.10.2 kNS_ WAV = "http://ns.adobe.com/xmp/wav/1.0/"	106
* 2.10.2 kNS_ XML = "http://www.w3.org/XML/1998/namespace"	106
* 2.10.2 kNS_ XMP = "http://ns.adobe.com/xap/1.0/"	106
* 2.10.2 kNS_ XMP_ BJ = "http://ns.adobe.com/xap/1.0/bj/"	106
* 2.10.2 kNS_ XMP_ Dimensions = "http://ns.adobe.com/xap/1.0/sType/Dimensions# "	106
* 2.10.2 kNS_ XMP_ Font = "http://ns.adobe.com/xap/1.0/sType/Font# "	107
* 2.10.2 kNS_ XMP_ Graphics = "http://ns.adobe.com/xap/1.0/g/"	107

* 2.10.2 kNS_XMP_G_IMG = "http://ns.adobe.com/xap/1.0/g/img/"	107
* 2.10.2 kNS_XMP_IdentifierQual = "http://ns.adobe.com/xmp/Identifier/qual/1.0/"	107
* 2.10.2 kNS_XMP_Image = "http://ns.adobe.com/xap/1.0/g/img/"	107
* 2.10.2 kNS_XMP_ManifestItem = "http://ns.adobe.com/xap/1.0/sType/ManifestItem# "	107
* 2.10.2 kNS_XMP_MM = "http://ns.adobe.com/xap/1.0/mm/"	108
* 2.10.2 kNS_XMP_Note = "http://ns.adobe.com/xmp/note/"	108
* 2.10.2 kNS_XMP_PagedFile = "http://ns.adobe.com/xap/1.0/t/pg/"	108
* 2.10.2 kNS_XMP_ResourceEvent = "http://ns.adobe.com/xap/1.0/sType/ResourceEvent# "	108
* 2.10.2 kNS_XMP_ResourceRef = "http://ns.adobe.com/xap/1.0/sType/ResourceRef# "	108
* 2.10.2 kNS_XMP_Rights = "http://ns.adobe.com/xap/1.0/rights/"	109
* 2.10.2 kNS_XMP_ST_Job = "http://ns.adobe.com/xap/1.0/sType/Job# "	109
* 2.10.2 kNS_XMP_ST_Version = "http://ns.adobe.com/xap/1.0/sType/Version# "	109
* 2.10.2 kNS_XMP_T = "http://ns.adobe.com/xap/1.0/t/"	109
* 2.10.2 kNS_XMP_Text = "http://ns.adobe.com/xap/1.0/t/"	109
* 2.10.2 kNS_XMP_T_PG = "http://ns.adobe.com/xap/1.0/t/pg/"	110
* 2.10.2 kOmitAllFormatting = & h800	110
* 2.10.2 kOmitPacketWrapper = & h10	110
* 2.10.2 kOmitXMPMetaElement = & h1000	110
* 2.10.2 kParseMoreBuffers = 2	110
* 2.10.2 kPropArrayFormMask = & h1E00	111
* 2.10.2 kPropArrayIsAlternate = & h800	111
* 2.10.2 kPropArrayIsAltText = & h1000	111
* 2.10.2 kPropArrayIsOrdered = & h400	111
* 2.10.2 kPropArrayIsUnordered = & h200	111
* 2.10.2 kPropArrayLocationMask = & hC000	112
* 2.10.2 kPropCompositeMask = & h1F00	112
* 2.10.2 kPropHasAliases = & h20000	112
* 2.10.2 kPropHasLang = & h40	112
* 2.10.2 kPropHasQualifiers = & h10	112
* 2.10.2 kPropHasType = & h80	113
* 2.10.2 kPropIsAlias = & h10000	113
* 2.10.2 kPropIsDerived = & h200000	113
* 2.10.2 kPropIsInternal = & h40000	113
* 2.10.2 kPropIsQualifier = & h20	113
* 2.10.2 kPropIsStable = & h100000	114
* 2.10.2 kPropValueIsArray = & h200	114
* 2.10.2 kPropValueIsStruct = & h100	115
* 2.10.2 kPropValueIsURI = 2	115
* 2.10.2 kPropValueOptionsMask = 2	115

* 2.10.2 kReadOnlyPacket = & h20	115
* 2.10.2 kReplaceOldValues = 2	115
* 2.10.2 kRequireXMPMeta = 1	116
* 2.10.2 kStrictAliasing = 4	116
* 2.10.2 kUseCompactFormat = & h40	116
* 2.10.2 kUseNullTermination = 0	116
* 2.10.2 kUTF16Bit = 2	116
* 2.10.2 kUTF32Bit = 4	116
* 2.10.2 kWriteAliasComments = & h400	117
* 2.10.2 kXMPFiles_ IgnoreLocalText = 2	117
* 2.10.2 kXMPFiles_ ServerMode = 2	117
* 2.10.2 kXMPTemplate_ AddNewProperties = 8	117
* 2.10.2 kXMPTemplate_ ClearUnnamedProperties = & h10	117
* 2.10.2 kXMPTemplate_ IncludeInternalProperties = 1	118
* 2.10.2 kXMPTemplate_ ReplaceExistingProperties = 2	118
* 2.10.2 kXMPTemplate_ ReplaceWithDeleteEmpty = 4	118
* 2.10.2 kXMP_ NS_ BWF = "http://ns.adobe.com/bwf/bext/1.0/"	118
* 2.10.2 kXMP_ NS_ Script = "http://ns.adobe.com/xmp/1.0/Script/"	118
* 2.10.2 kXMP_ WriteAliasComments = & h400	118
* 2.10.2 Version = "5.1.2"	119
– 2.1 class XMPPacketInfoMBS	15
* 2.1.1 CharForm as Integer	15
* 2.1.1 HasWrapper as Boolean	15
* 2.1.1 Length as Integer	16
* 2.1.1 Offset as Int64	16
* 2.1.1 PadSize as Integer	16
* 2.1.1 Writeable as Boolean	16
– 2.2 class XMPScannerMBS	16
* 2.2.1 Constructor	18
* 2.2.1 Constructor(StreamLength as integer)	18
* 2.2.1 Report as integer	18
* 2.2.1 Scan(Buffer as string, Offset as Int64)	19
* 2.2.1 Snip(index as UInt32) as XMP_SnipMBS	19
* 2.2.1 SnipCount as UInt32	19
* 2.2.1 StreamAllScanned as boolean	19
– 2.4 class XMPTextOutputMBS	21
* 2.4.1 Output(text as string) as integer	22
– 2.3 class XMPVersionInfoMBS	19
* 2.3.1 Build as Integer	20
* 2.3.1 Flags as Integer	20
* 2.3.1 IsDebug as boolean	20

* 2.3.1 Major as Integer	20
* 2.3.1 Message as String	21
* 2.3.1 Micro as Integer	21
* 2.3.1 Minor as Integer	21
– 2.5 class XMPSnipMBS	22
* 2.5.1 Access as Integer	22
* 2.5.1 BytesAttr as Int64	22
* 2.5.1 CharForm as Integer	23
* 2.5.1 EncodingAttr as String	23
* 2.5.1 Length as Int64	23
* 2.5.1 Offset as Int64	23
* 2.5.1 OutOfOrder as Integer	24
* 2.5.1 State as Integer	24
– 2.9 class XMPFilesMBS	32
* 2.9.1 CanPutXMP(xmpPacket as string) as boolean	33
* 2.9.1 CanPutXMP(xmpPacket as XMPMetaMBS) as boolean	33
* 2.9.1 CheckFileFormat(path as string) as integer	34
* 2.9.1 CheckPackageFormat(path as string) as integer	34
* 2.9.1 CloseFile(closeFlags as integer)	34
* 2.9.1 Constructor	35
* 2.9.1 Constructor(path as folderitem, format as integer=& h20202020, OpenFlags as integer=0)	35
* 2.9.1 Constructor(path as string, format as integer=& h20202020, OpenFlags as integer=0)	37
* 2.9.1 GetFileInfo(byref path as string, byref openFlags as integer, byref format as integer, byref handlerFlags as integer) as boolean	37
* 2.9.1 GetFormatInfo(format as integer, byref handlerFlags as UInt32) as boolean	38
* 2.9.1 GetVersionInfo as XMPVersionInfoMBS	39
* 2.9.1 GetXMP(byref xmp as XMPMetaMBS, byref xmppacket as string, byref PacketInfo as XMPPacketInfoMBS) as boolean	39
* 2.9.1 OpenFile(path as folderitem, format as integer=& h20202020, OpenFlags as integer=0) as boolean	40
* 2.9.1 OpenFile(path as string, format as integer=& h20202020, OpenFlags as integer=0) as boolean	40
* 2.9.1 PutXMP(xmpPacket as string)	42
* 2.9.1 PutXMP(xmpPacket as XMPMetaMBS)	42
* 2.9.2 Abort as boolean	42
* 2.9.3 kAEFilterPresetFile = & h46465820	43
* 2.9.3 kAEProjectFile = & h41455020	43
* 2.9.3 kAEProjTemplateFile = & h41455420	43
* 2.9.3 kAIFFFFile = & h41494646	43
* 2.9.3 kAllowsOnlyXMP = & h00000020	43

* 2.9.3 kAllowsSafeUpdate = & h00000200	44
* 2.9.3 kArrayLastItem = -1	44
* 2.9.3 kAVCHDFile = & h41564844	44
* 2.9.3 kAVIFile = & h41564920	44
* 2.9.3 kCanExpand = 2	44
* 2.9.3 kCanInjectXMP = 1	44
* 2.9.3 kCanReconcile = & h00000010	45
* 2.9.3 kCanRewrite = 4	45
* 2.9.3 kCELFile = & h43454C20	45
* 2.9.3 kChar16BitBig = 2	45
* 2.9.3 kChar16BitLittle = 3	45
* 2.9.3 kChar16BitMask = 2	46
* 2.9.3 kChar32BitBig = 4	46
* 2.9.3 kChar32BitLittle = 5	46
* 2.9.3 kChar32BitMask = 4	46
* 2.9.3 kChar8Bit = 0	46
* 2.9.3 kCharLittleEndianMask = 1	46
* 2.9.3 kCharUnknown = 1	47
* 2.9.3 kCINFile = & h43494E20	47
* 2.9.3 kEncoreProjectFile = & h4E434F52	47
* 2.9.3 kEPSFile = & h45505320	47
* 2.9.3 kFLAFile = & h464C4120	47
* 2.9.3 kFLVFile = & h464C5620	48
* 2.9.3 kFolderBasedFormat = & h00001000	48
* 2.9.3 kGIFFFile = & h47494620	48
* 2.9.3 kHandlerOwnsFile = & h00000100	48
* 2.9.3 kHTMLFile = & h48544D4C	48
* 2.9.3 kIllustratorFile = & h41492020	48
* 2.9.3 kInDesignFile = & h494E4444	49
* 2.9.3 kJPEG2KFile = & h4A505820	49
* 2.9.3 kJPEGFile = & h4A504547	49
* 2.9.3 kMOVFile = & h4D4F5620	49
* 2.9.3 kMP3File = & h4D503320	49
* 2.9.3 kMPEG2File = & h4D503220	49
* 2.9.3 kMPEG4File = & h4D503420	50
* 2.9.3 kMPEGFile = & h4D504547	50
* 2.9.3 kNeedsReadOnlyPacket = & h00000400	50
* 2.9.3 kNoOptions = 0	50
* 2.9.3 kOpenCacheTNail = 8	50
* 2.9.3 kOpenForRead = 1	50
* 2.9.3 kOpenForUpdate = 2	51
* 2.9.3 kOpenInBackground = & h10000000	51

* 2.9.3 kOpenLimitedScanning = & h00000080	51
* 2.9.3 kOpenOnlyXMP = 4	51
* 2.9.3 kOpenRepairFile = & h00000100	51
* 2.9.3 kOpenStrictly = & h00000010	52
* 2.9.3 kOpenUsePacketScanning = & h00000040	52
* 2.9.3 kOpenUseSmartHandler = & h00000020	52
* 2.9.3 kP2File = & h50322020	52
* 2.9.3 kPDFFile = & h50444620	52
* 2.9.3 kPhotoshopFile = & h50534420	53
* 2.9.3 kPNGFile = & h504E4720	53
* 2.9.3 kPostScriptFile = & h50532020	53
* 2.9.3 kPrefersInPlace = 8	53
* 2.9.3 kPremiereProjectFile = & h5052504A	53
* 2.9.3 kPremiereTitleFile = & h5052544C	53
* 2.9.3 kReturnsRawPacket = & h00000040	54
* 2.9.3 kReturnsTNail = & h00000080	54
* 2.9.3 kSESFile = & h53455320	54
* 2.9.3 kSonyHDVFile = & h53484456	54
* 2.9.3 kSWFFile = & h53574620	54
* 2.9.3 kTextFile = & h74657874	55
* 2.9.3 kTIFFFile = & h54494646	55
* 2.9.3 kUCFFile = & h55434620	55
* 2.9.3 kUnknownFile = & h20202020	55
* 2.9.3 kUnknownLength = -1	55
* 2.9.3 kUnknownOffset = -1	55
* 2.9.3 kUpdateSafely = 1	56
* 2.9.3 kUseNullTermination = 0	56
* 2.9.3 kUsesSidecarXMP = & h00000800	56
* 2.9.3 kWAVFile = & h57415620	56
* 2.9.3 kWMAVFile = & h574D4156	56
* 2.9.3 kXDCAM_EXFile = & h58444358	56
* 2.9.3 kXDCAM_FAMFile = & h58444346	57
* 2.9.3 kXDCAM_SAMFile = & h58444353	57
* 2.9.3 kXMLFile = & h584D4C20	57
- 2.6 class XMPDateTimeMBS	24
* 2.6.1 ClearTimeZone	25
* 2.6.1 Clone as XMPDateTimeMBS	25
* 2.6.1 Compare(other as XMPDateTimeMBS) as integer	25
* 2.6.1 Constructor	25
* 2.6.1 Constructor(text as string)	26
* 2.6.1 ConvertToLocalTime	26

* 2.6.1 ConvertToUTCtime	26
* 2.6.1 IsDateOnly as Boolean	26
* 2.6.1 IsTimeOnly as Boolean	26
* 2.6.1 Operator_ Convert as string	27
* 2.6.1 Operator_ Convert(text as string)	27
* 2.6.1 SetTimeZone	27
* 2.6.1 Str as string	27
* 2.6.2 Day as Integer	27
* 2.6.2 hasDate as Boolean	28
* 2.6.2 hasTime as Boolean	28
* 2.6.2 hasTimeZone as Boolean	28
* 2.6.2 Hour as Integer	28
* 2.6.2 Minute as Integer	28
* 2.6.2 Month as Integer	29
* 2.6.2 NanoSecond as Integer	29
* 2.6.2 Second as Integer	29
* 2.6.2 TimezoneHour as Integer	29
* 2.6.2 TimezoneMinute as Integer	29
* 2.6.2 TimezoneSign as Integer	30
* 2.6.2 Year as Integer	30
* 2.6.3 kXMP_ TimeEastOfUTC = 1	30
* 2.6.3 kXMP_ TimeIsUTC = 0	30
* 2.6.3 kXMP_ TimeWestOfUTC = -1	30
- 2.7 class XMPAssertNotifyMBS	31
* 2.7.1 SendAssertNotify(message as string)	31
* 2.7.2 Assert(text as string)	31
- 2.11 class XMPIteratorMBS	119
* 2.11.1 Constructor	120
* 2.11.1 Constructor(meta as XMPMetaMBS, options as integer=0)	120
* 2.11.1 Constructor(meta as XMPMetaMBS, schemaNS as string, options as integer=0)	121
* 2.11.1 Constructor(meta as XMPMetaMBS, schemaNS as string, propName as string, options as integer=0)	122
* 2.11.1 Constructor(schemaNS as string, propName as string, options as integer)	122
* 2.11.1 NextItem() as boolean	123
* 2.11.1 NextItem(byref schemaNS as string) as boolean	123
* 2.11.1 NextItem(byref schemaNS as string, byref propPath as string) as boolean	124
* 2.11.1 NextItem(byref schemaNS as string, byref propPath as string, byref propValue as string) as boolean	125
* 2.11.1 NextItem(byref schemaNS as string, byref propPath as string, byref propValue as string, byref options as integer) as boolean	125
* 2.11.1 Skip(options as integer)	126

Chapter 2

XMP

2.1 class XMPPacketInfoMBS

class XMPPacketInfoMBS

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The class for information about a xmp packet.

2.1.1 Properties

CharForm as Integer

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Character format using the values kChar8Bit, kChar16BitBig, etc.

Notes: (Read and Write property)

HasWrapper as Boolean

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** True if there is a packet wrapper, the "<?xpacket...>" XML processing instructions.

Notes: (Read and Write property)

Length as Integer

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Packet length in the file in bytes, -1 if unknown.

Notes: (Read and Write property)

Offset as Int64

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Packet offset in the file in bytes, -1 if unknown.

Notes: (Read and Write property)

PadSize as Integer

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Packet padding size in bytes, zero if unknown.

Notes: (Read and Write property)

Writeable as Boolean

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** True if there is a packet wrapper and the trailer says writeable by dumb packet scanners.

Notes: (Read and Write property)

2.2 class XMPScannerMBS

class XMPScannerMBS

Plugin Version: 6.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The class to scan any file stream for XMP data.

Example:

```
dim c,n,i,l as integer
dim xs as XMPScannerMBS
dim xn as XMPSnipMBS
dim s as string
dim x as XMPMetaMBS
dim b as BinaryStream
dim source as FolderItem

// get file
source=GetOpenFolderItem("any")
if source=nil then Return

b=source.OpenAsBinaryFile(false)

if b=nil then Return

// read file to memory
l=b.Length
s=b.Read(l)

// scan for xmp blocks
MsgBox "Scanning " +str(lenb(s))+" of " +str(l)+" bytes."

xs=new XMPScannerMBS(l)
xs.Scan(s,0)

MsgBox "Found " +str(xs.Report)+" blocks"

n=xs.Report-1

c=0
for i=0 to n

xn=xs.Snip(i)

if xn.State=3 then // found
if xn.Length>50 then
try
b.Position=xn.Offset

s=b.Read(xn.Length)

// debug output xmp data
WriteInputXMP dest,s

// try to parse
```

```
x=new XMPMetaMBS(s)

// work with xmp meta data
catch r as XMPEXceptionMBS
MsgBox "ExtractXMP failed on: "+r.message
end try
else
MsgBox "Found small xmp block? "+str(xn.Length)
end if
end if
next
```

2.2.1 Methods

Constructor

Plugin Version: 6.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Default constructor doing nothing.

See also:

- 2.2.1 Constructor(StreamLength as integer) 18

Constructor(StreamLength as integer)

Plugin Version: 6.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Constructor to create a XMPScanner which reads the given number of bytes.

See also:

- 2.2.1 Constructor 18

Report as integer

Plugin Version: 6.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Produces a report of what is known about the input stream.

Notes: The snippets found are saved in an array which you can access using the `snip()` function. Returns the number of entries in this array.

Scan(Buffer as string, Offset as Int64)

Plugin Version: 6.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Scans the given part of the input, incorporating it in to the known snips.

Notes: The Offset is the offset of this block of input relative to the entire stream.

Snip(index as UInt32) as XMPSnipMBS

Plugin Version: 11.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Returns the snip with the given index.

Notes:

index is zero based.

Returns nil on any error.

You must call Report before this method.

SnipCount as UInt32

Plugin Version: 11.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Number of snips found so far.

StreamAllScanned as boolean

Plugin Version: 6.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Returns true if all of the stream has been seen.

2.3 class XMPVersionInfoMBS

class XMPVersionInfoMBS

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The XMP Version class.

Example:

```
dim v as XMPVersionInfoMBS
v=XMPMetaMBS.GetVersionInfo
MsgBox v.Message
```

Notes: Constructor sets values.

2.3.1 Properties

Build as Integer

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** A rolling build number, monotonically increasing in a release.

Notes: (Read and Write property)

Flags as Integer

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Individual feature implementation flags.

Notes: (Read and Write property)

IsDebug as boolean

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** True if this is a debug build.

Notes: (Read and Write property)

Major as Integer

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The primary release number, the "1" in version "1.2.3".

Notes: (Read and Write property)

Message as String

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** A comprehensive version information string.

Example:

```
dim v as XMPVersionInfoMBS
v=XMPMetaMBS.GetVersionInfo
MsgBox v.Message
```

Notes: (Read and Write property)

Micro as Integer

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The tertiary release number, the "3" in version "1.2.3".

Notes: (Read and Write property)

Minor as Integer

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The secondary release number, the "2" in version "1.2.3".

Notes: (Read and Write property)

2.4 class XMPTextOutputMBS

class XMPTextOutputMBS

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** An event class for text output.

Notes: You may want to make a subclass which collects all the strings into one string or write them to a file.

2.4.1 Events

Output(text as string) as integer

Plugin Version: 6.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** New text arrived.
Notes:

You may get more than one event per line.
Return 0 for success or something else for failure.

2.5 class XMPSnipMBS

class XMPSnipMBS

Plugin Version: 6.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The class for the XMP Snips found by the XMPScannerMBS class.

2.5.1 Properties

Access as Integer

Plugin Version: 6.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The read-only/read-write access from the end attribute.
Notes: (Read and Write property)

BytesAttr as Int64

Plugin Version: 6.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The value of the bytes attribute, -1 if not present.
Notes: (Read only property)

CharForm as Integer

Plugin Version: 6.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** How the packet is divided into characters.

Notes:

The values allow easy testing for 16/32 bit and big/little endian.

```
eChar8Bit          = 0
eChar16BitBig      = 2
eChar16BitLittle   = 3
eChar32BitBig      = 4
eChar32BitLittle   = 5
```

(Read and Write property)

EncodingAttr as String

Plugin Version: 6.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The value of the encoding attribute, if any, with nulls removed.

Notes: (Read and Write property)

Length as Int64

Plugin Version: 6.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The length in bytes of this snip.

Notes: (Read only property)

Offset as Int64

Plugin Version: 6.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The byte offset of this snip within the input stream.

Notes: (Read only property)

OutOfOrder as Integer

Plugin Version: 6.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** If true, this snip was seen before the one in front of it.

Notes: (Read and Write property)

State as Integer

Plugin Version: 6.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The state of this snip.

Notes:

constants:

eNotSeenSnip	0	This snip has not been seen yet.
ePendingSnip	1	This snip is an input buffer being processed.
eRawInputSnip	2	This snip is raw input, it doesn't contain any part of an XMP packet.
eValidPacketSnip	3	This snip is a complete, valid XMP packet.
ePartialPacketSnip	4	This snip contains the start of a possible XMP packet.
eBadPacketSnip	5	This snip contains a complete, but semantically incorrect XMP packet.

(Read and Write property)

2.6 class XMPDateTimeMBS**class XMPDateTimeMBS**

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The expanded type for a date and time.

Notes:

Dates and time in the serialized XMP are ISO 8601 strings. The XMPDateTimeMBS class allows easy conversion with other formats.

Date/Time values are occasionally used in cases with only a date or only a time component. A date without a time has zeros in the XMPDateTimeMBS class for all time fields. A time without a date has zeros for all

date fields (year, month, and day).

2.6.1 Methods

ClearTimeZone

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Resets the timezone by setting all time zone properties to zero.

Clone as XMPDateTimeMBS

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Creates a copy of this date.

Notes: Not just a new reference to the same object, but a real copy.

Compare(other as XMPDateTimeMBS) as integer

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Compare the order of two date/time values.

Notes:

Returns:

-1 if self is before other or other=nil

0 if self matches other

+1 if self is after other

Constructor

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Obtain the current date and time.

Notes: The returned time is UTC, properly adjusted for the local time zone. The resolution of the time is not guaranteed to be finer than seconds.

See also:

- 2.6.1 Constructor(text as string) 26

Constructor(text as string)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Convert from string to date/time.

See also:

- 2.6.1 Constructor 25

ConvertToLocalTime

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Make sure a time is local.

Notes: If the time zone is not the local zone, the time is adjusted and the time zone set to be local.

ConvertToUTCtime

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Make sure a time is UTC.

Notes: If the time zone is not UTC, the time is adjusted and the time zone set to be UTC.

IsDateOnly as Boolean

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether only the date is defined for this datetime object.

IsTimeOnly as Boolean

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether only the time is defined for this datetime object.

Operator_ Convert as string

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Convert from date/time to string.

See also:

- 2.6.1 Operator_ Convert(text as string) 27

Operator_ Convert(text as string)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Convert from string to date/time.

See also:

- 2.6.1 Operator_ Convert as string 27

SetTimeZone

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Set the local time zone.

Notes: Any existing time zone value is replaced, the other date/time fields are not adjusted in any way.

Str as string

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Convert from date/time to string.

2.6.2 Properties**Day as Integer**

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The day of the month in the range 1..31.

Notes: (Read and Write property)

hasDate as Boolean

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** A property to tell if the date is defined in this object.

Notes: (Read and Write property)

hasTime as Boolean

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** A property to tell if the time is defined in this object.

Notes: (Read and Write property)

hasTimeZone as Boolean

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Whether there is a time zone set.

Notes: (Read and Write property)

Hour as Integer

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The hour in the range 0..23.

Notes: (Read and Write property)

Minute as Integer

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The minute in the range 0..59.

Notes: (Read and Write property)

Month as Integer

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The month in the range 1..12.

Notes: (Read and Write property)

NanoSecond as Integer

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Nanoseconds within a second, often left as zero.

Notes: (Read and Write property)

Second as Integer

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The second in the range 0..59.

Notes: (Read and Write property)

TimezoneHour as Integer

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The time zone hour in the range 0..23.

Notes: (Read and Write property)

TimezoneMinute as Integer

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The time zone minute in the range 0..59.

Notes: (Read and Write property)

TimezoneSign as Integer

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The "sign" of the time zone, 0 means UTC, -1 is west, +1 is east.

Notes:

The "sign" of the time zone, kTimeIsUTC (0) means UTC, kTimeWestOfUTC (-1) is west, kTimeEastOfUTC (+1) is east.

(Read and Write property)

Year as Integer

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The year, can be negative.

Notes: (Read and Write property)

2.6.3 Constants

kXMP_TimeEastOfUTC = 1

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants for the TimezoneSign property.

Notes: UTC time.

kXMP_TimeIsUTC = 0

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants for the TimezoneSign property.

Notes: Time zone is east of UTC.

kXMP_TimeWestOfUTC = -1

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants for the TimezoneSign property.

Notes: Time zone is west of UTC.

2.7 class XMPAssertNotifyMBS

class XMPAssertNotifyMBS

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** This class provides a way to get notified for Assertion.

Notes: Only one instance can be registered at a given time.

2.7.1 Methods

SendAssertNotify(message as string)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** SendAssertNotify sends an assert notification with the given message.

Example:

```
dim a as MyXMPAssertNotifyMBS
a=new MyXMPAssertNotifyMBS
a.SendAssertNotify "test" // will show msgbox using event
```

2.7.2 Events

Assert(text as string)

Plugin Version: 6.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** An assertion was reported.

Example:

```
Sub Assert(text as string)
MsgBox text
End Sub
```

Notes: Display it to the user or just log it for debugging.

2.8 class XMPEXceptionMBS

class XMPEXceptionMBS

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The exception class for the other XMP classes.

Notes:

For current plugins (6.4): Windows and MachO works, Carbon/Classic PEF crashes on an exception. Linux not tested.

Nearly every method or function may raise this exception, so catch it!
Subclass of the RuntimeException class.

2.9 class XMPFilesMBS

class XMPFilesMBS

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The class for access to the main (document-level) metadata in a file.

Notes:

The Adobe XMP Toolkit's file handling component, XMPFiles, is a front end to a set of format-specific file handlers that support file I/O for XMP. The file handlers implement smart, efficient support for those file formats for which the means to embed XMP is defined in the XMP Specification. Where possible, this support allows:

- * Injection of XMP where none currently exists

- * Expansion of XMP without regard to existing padding

Reconciliation of the XMP and other legacy forms of metadata.

TXMPFiles is designed for use by clients interested in the metadata and not in the primary file content; the Adobe Bridge application is a typical example. TXMPFiles is not intended to be appropriate for files authored by an application; that is, those files for which the application has explicit knowledge of the file format.

Supported file formats:

PDF, PostScript, EPS, JPEG, JPEG2K, TIFF, GIF, PNG, SWF, FLA, FLV, MOV, AVI, CIN, WAV, MP3, SES, CEL, MPEG, MPEG2, MPEG4, WMAV, AIFF, P2, XDCAM_ FAM, XDCAM_ SAM, XDCAM_ EX, AVCHD, SonyHDV, HTML, XML, Text, Photoshop, Illustrator, InDesign, AEProject, AEProjTemplate, AEFILTERPreset, EncoreProject, PremiereProject, PremiereTitle and UCF.

Based on the XMP-Toolkit-SDK from Adobe.

2.9.1 Methods

CanPutXMP(xmpPacket as string) as boolean

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** CanPutXMP() reports whether this file can be updated with a specific XMP packet.

Notes: Use to determine if the file can probably be updated with a given set of XMP metadata. This depends on the size of the packet, the options with which the file was opened, and the capabilities of the handler for the file format. The function obtains the length of the serialized packet for the provided XMP, but does not keep it or modify it, and does not cause the file to be written when closed.

See also:

- 2.9.1 CanPutXMP(xmpPacket as XMPMetaMBS) as boolean 33

CanPutXMP(xmpPacket as XMPMetaMBS) as boolean

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:**

CanPutXMP() reports whether this file can be updated with a specific XMP packet, passed in a string object.

See also:

- 2.9.1 CanPutXMP(xmpPacket as string) as boolean 33

CheckFileFormat(path as string) as integer

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** CheckFileFormat() tries to determine the format of a file.

Notes:

CheckFileFormat tries to determine the format of a file, returning an format value. It uses the same logic as OpenFile will use to select a smart handler.

path: The path for the file, appropriate for the local operating system. Passed as a UTF-8 string. The path is the same as would be passed to OpenFile.

Returns the file's format if a smart handler would be selected, otherwise kUnknownFile.

CheckPackageFormat(path as string) as integer

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** CheckPackageFormat() tries to determine the format of a "package" folder.

Notes:

CheckPackageFormat tries to determine the format of a "package" given the name of the top level folder, returning an XMP_ FileFormat value. Examples of recognized packages include the video formats P2, XDCAM, or Sony HDV. These packages contain collections of "clips", stored as multiple files in specific subfolders.

Path: The path for the top level folder, appropriate for the local operating system. Passed as an UTF-8 string. The path is not the same as would be passed to OpenFile. For example the path passed to CheckPackageFormat might be ".../MyMovie", while the path passed to OpenFile would be ".../MyMovie/SomeClip".

Returns the package's format if a smart handler would be selected, otherwise kUnknownFile.

CloseFile(closeFlags as integer)

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** CloseFile() explicitly closes an opened file.

Notes:

Performs any necessary output to the file and closes it. Files that are opened for update are written to only when closing.

If the file is opened for read-only access (passing `kOpenForRead`), the disk file is closed immediately after reading the data from it; the `XMPFiles` object, however, remains in the open state. You must call `CloseFile()` when finished using it. Other methods, such as `GetXMP()`, can only be used between the `OpenFile()` and `CloseFile()` calls. The `XMPFiles` destructor does not call `CloseFile()`; if you call it without closing, any pending updates are lost.

If the file is opened for update (passing `kOpenForUpdate`), the disk file remains open until `CloseFile()` is called. The disk file is only updated once, when `CloseFile()` is called, regardless of how many calls are made to `PutXMP()`.

`closeFlags`: Option flags for optional closing actions. This bit-flag constant is defined:

* `kUpdateSafely` - Write into a temporary file then swap for crash safety.

Constructor

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Default constructor initializes an object that is associated with no file.

Notes: The destructor does not call `CloseFile()`; pending updates are lost when the destructor is run. See also:

- 2.9.1 `Constructor(path as folderitem, format as integer=& h20202020, OpenFlags as integer=0)` 35
- 2.9.1 `Constructor(path as string, format as integer=& h20202020, OpenFlags as integer=0)` 37

`Constructor(path as folderitem, format as integer=& h20202020, OpenFlags as integer=0)`

Plugin Version: 10.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Opens a file for metadata access.

Notes:

Opens a file for the requested forms of metadata access. Opening the file at a minimum causes the raw XMP packet to be read from the file. If the file handler supports legacy metadata reconciliation then legacy

metadata is also read, unless `kOpenOnlyXMP` is passed. If the file handler supports native thumbnails and `kOpenCacheTNail` is passed, the native thumbnail is cached.

If the file is opened for read-only access (passing `kOpenForRead`), the disk file is closed immediately after reading the data from it; the `XMPFiles` object, however, remains in the open state. You must call `CloseFile()` when finished using it. Other methods, such as `GetXMP()`, can only be used between the `OpenFile()` and `CloseFile()` calls. The `XMPFiles` destructor does not call `CloseFile()`; if you call it without closing, any pending updates are lost.

If the file is opened for update (passing `kOpenForUpdate`), the disk file remains open until `CloseFile()` is called. The disk file is only updated once, when `CloseFile()` is called, regardless of how many calls are made to `PutXMP()`.

Typically, the XMP is not parsed and legacy reconciliation is not performed until `GetXMP()` is called, but this is not guaranteed. Specific file handlers might do earlier parsing of the XMP. Delayed parsing and early disk file close for read-only access are optimizations to help clients implementing file browsers, so that they can access the file briefly and possibly display a thumbnail, then postpone more expensive XMP processing until later.

`path`: The path for the file, appropriate for the local operating system.

`format`: The format of the file. If the format is unknown (`kUnknownFile`) the format is determined from the file content. The first handler to check is guessed from the file's extension. Passing a specific format value is generally just a hint about what file handler to try first (instead of the one based on the extension). If `kOpenStrictly` is set, then any format other than `kUnknownFile` requires that the file actually be that format; otherwise an exception is thrown.

`openFlags`: A set of option flags that describe the desired access. By default (zero) the file is opened for read-only access and the format handler decides on the level of reconciliation that will be performed. A logical OR of these bit-flag constants:

* `kOpenForRead` - Open for read-only access.

`kOpenForUpdate` - Open for reading and writing.

`kOpenOnlyXMP` - Only the XMP is wanted, no reconciliation.

`kOpenCacheTNail` - Cache thumbnail if possible, `GetThumbnail` will be called.

`kOpenStrictly` - Be strict about locating XMP and reconciling with other forms. By default, a best effort is made to locate the correct XMP and to reconcile XMP with other forms (if reconciliation is done). This option forces stricter rules, resulting in exceptions for errors. The definition of strictness is specific to each handler, there might be no difference.

`kOpenUseSmartHandler` - Require the use of a smart handler.

`kOpenUsePacketScanning` - Force packet scanning, do not use a smart handler.

Throws an exception for serious problems.

See also:

- 2.9.1 Constructor 35
- 2.9.1 Constructor(path as string, format as integer=& h20202020, OpenFlags as integer=0) 37

Constructor(path as string, format as integer=& h20202020, OpenFlags as integer=0)

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Alternate constructor associates the new XMPFiles object with a specific file.

Notes:

Calls `OpenFile()` to open the specified file after performing a default construct.

path: The path for the file, specified as an UTF-8 string. (On MacOSX use `UnixpathMBS`)

format: A format hint for the file, if known.

openFlags: Options for how the file is to be opened (for read or read/write, for example). Use a logical OR of these bit-flag constants:

```
* kOpenForRead
kOpenForUpdate
kOpenOnlyXMP
kOpenCacheTNail
kOpenStrictly
kOpenUseSmartHandler
kOpenUsePacketScanning
kOpenLimitedScanning
kOpenInBackground
```

The destructor does not call `CloseFile()`; pending updates are lost when the destructor is run. See also:

- 2.9.1 Constructor 35
- 2.9.1 Constructor(path as folderitem, format as integer=& h20202020, OpenFlags as integer=0) 35

GetFileInfo(byref path as string, byref openFlags as integer, byref format as integer, byref handlerFlags as integer) as boolean

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** `GetFileInfo()` retrieves basic information about an opened file.

Notes:

path: A buffer in which to return the path passed to `OpenFile()`. Can be null if value is not wanted.

openFlags: A variable in which to return the option flags passed to OpenFile.
 format: A variable in which to return the file format.
 handlerFlags: [out] A variable in which to return the handler's capability flags.

Returns true if the file object is in the open state; that is, OpenFile() has been called but CloseFile() has not. False otherwise. Even if the file object is open, the actual disk file might be closed in the host file-system sense; see OpenFile().

GetFormatInfo(format as integer, byref handlerFlags as UInt32) as boolean

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** GetFormatInfo() reports what features are supported for a specific file format.

Notes:

The file handlers for different file formats vary considerably in what features they support. Support depends on both the general capabilities of the format and the implementation of the handler for that format.

This function is static; make the call directly from the concrete class (SXMPFiles).

format: The file format whose support flags are desired.
 handlerFlags: A variable in which to return a logical OR of option bit flags.
 The following constants are defined:

- * kCanInjectXMP - Can inject first-time XMP into an existing file.
- kCanExpand - Can expand XMP or other metadata in an existing file.
- kCanRewrite - Can copy one file to another, writing new metadata (as in SaveAs)
- kCanReconcile - Supports reconciliation between XMP and other forms.
- kAllowsOnlyXMP - Allows access to just the XMP, ignoring other forms.
 This is only meaningful if kCanReconcile is set.
- kReturnsTNail - File handler returns native thumbnail information.
- kReturnsRawPacket - File handler returns raw XMP packet information and string.

Even if kReturnsRawPacket is set, the returned packet information might have an offset of -1 to indicate an unknown offset. While all file handlers should be able to return the raw packet, some might not know the offset of the packet within the file. This is typical in cases where external libraries are used. These cases might not even allow return of the raw packet.

Returns true if the format has explicit "smart" support, false if the format is handled by the default packet scanning plus heuristics.

GetVersionInfo as XMPVersionInfoMBS

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** GetVersionInfo() retrieves version information for the XMPFiles component.

GetXMP(byref xmp as XMPMetaMBS, byref xmppacket as string, byref PacketInfo as XMP-PacketInfoMBS) as boolean

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** GetXMP() retrieves the XMP metadata from an open file.

Notes:

The function reports whether XMP is present in the file; you can choose to retrieve any or all of the parsed XMP, the raw XMP packet, or information about the raw XMP packet. The options provided when the file was opened determine if reconciliation is done with other forms of metadata.

xmp: An XMP object in which to return the parsed XMP metadata.

xmppacket: A string in which to return the raw XMP packet as stored in the file. The encoding of the packet is given in the packetInfo. Returns an empty string if the low level file handler does not provide the raw packet.

packetInfo: A packetinfo object in which to return the location and form of the raw XMP in the file. PacketInfo.charForm and PacketInfo.writeable reflect the

raw XMP in the file. The parsed XMP property values are always UTF-8. The writeable flag is taken from the packet trailer; it applies only to "format ignorant" writing. The PacketInfo object always reflects the state of the XMP in the file. The offset, length, and character form do not change as a result of calling PutXMP() unless the file is also written. Some file handlers might not return location or contents of the raw packet string. To determine whether one does, check the kReturnsRawPacket bit returned by GetFormatInfo(). If the low-level file handler does not provide the raw packet location, PacketInfo.offset and PacketInfo.length are both 0, PacketInfo.charForm is UTF-8, and PacketInfo.writeable is false.

OpenFile(path as folderitem, format as integer=& h20202020, OpenFlags as integer=0) as boolean

Plugin Version: 10.5 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Alternate constructor associates the new XMPFiles object with a specific file.

Notes:

Calls OpenFile() to open the specified file after performing a default construct.

path: The path for the file, specified as folderitem.

format: A format hint for the file, if known.

openFlags: Options for how the file is to be opened (for read or read/write, for example). Use a logical OR of these bit-flag constants:

```
* kOpenForRead
kOpenForUpdate
kOpenOnlyXMP
kOpenCacheTNail
kOpenStrictly
kOpenUseSmartHandler
kOpenUsePacketScanning
kOpenLimitedScanning
kOpenInBackground
```

The destructor does not call CloseFile(); pending updates are lost when the destructor is run.

See also:

- 2.9.1 OpenFile(path as string, format as integer=& h20202020, OpenFlags as integer=0) as boolean

OpenFile(path as string, format as integer=& h20202020, OpenFlags as integer=0) as boolean

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Opens a file for metadata access.

Notes:

Opens a file for the requested forms of metadata access. Opening the file at a minimum causes the raw XMP packet to be read from the file. If the file handler supports legacy metadata reconciliation then legacy metadata is also read, unless kOpenOnlyXMP is passed. If the file handler supports native thumbnails and kOpenCacheTNail is passed, the native thumbnail is cached.

If the file is opened for read-only access (passing `kOpenForRead`), the disk file is closed immediately after reading the data from it; the XMPFiles object, however, remains in the open state. You must call `CloseFile()` when finished using it. Other methods, such as `GetXMP()`, can only be used between the `OpenFile()` and `CloseFile()` calls. The XMPFiles destructor does not call `CloseFile()`; if you call it without closing, any pending updates are lost.

If the file is opened for update (passing `kOpenForUpdate`), the disk file remains open until `CloseFile()` is called. The disk file is only updated once, when `CloseFile()` is called, regardless of how many calls are made to `PutXMP()`.

Typically, the XMP is not parsed and legacy reconciliation is not performed until `GetXMP()` is called, but this is not guaranteed. Specific file handlers might do earlier parsing of the XMP. Delayed parsing and early disk file close for read-only access are optimizations to help clients implementing file browsers, so that they can access the file briefly and possibly display a thumbnail, then postpone more expensive XMP processing until later.

`path`: The path for the file, specified as an UTF-8 string. (On MacOSX use `UnixpathMBS`)

`format`: The format of the file. If the format is unknown (`kUnknownFile`) the format is determined from the file content. The first handler to check is guessed from the file's extension. Passing a specific format value is generally just a hint about what file handler to try first (instead of the one based on the extension). If `kOpenStrictly` is set, then any format other than `kUnknownFile` requires that the file actually be that format; otherwise an exception is thrown.

`openFlags`: A set of option flags that describe the desired access. By default (zero) the file is opened for read-only access and the format handler decides on the level of reconciliation that will be performed. A logical OR of these bit-flag constants:

* `kOpenForRead` - Open for read-only access.

`kOpenForUpdate` - Open for reading and writing.

`kOpenOnlyXMP` - Only the XMP is wanted, no reconciliation.

`kOpenCacheTNail` - Cache thumbnail if possible, `GetThumbnail` will be called.

`kOpenStrictly` - Be strict about locating XMP and reconciling with other forms. By default, a best effort is made to locate the correct XMP and to reconcile XMP with other forms (if reconciliation is done). This option forces stricter rules, resulting in exceptions for errors. The definition of strictness is specific to each handler, there might be no difference.

`kOpenUseSmartHandler` - Require the use of a smart handler.

`kOpenUsePacketScanning` - Force packet scanning, do not use a smart handler.

Returns true if the file is successfully opened and attached to a file handler. False for anticipated problems, such as passing `kOpenUseSmartHandler` but not having an appropriate smart handler. Throws an exception for serious problems.

See also:

- 2.9.1 `OpenFile(path as folderitem, format as integer=& h20202020, OpenFlags as integer=0)` as boolean

PutXMP(xmpPacket as string)

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** PutXMP() updates the XMP metadata in this object without writing out the file,

Notes: Overloads the basic form of the function, allowing you to pass the metadata as a string object instead of an XMP object. It is otherwise identical; see details in the canonical form.

See also:

- 2.9.1 PutXMP(xmpPacket as XMPMetaMBS) 42

PutXMP(xmpPacket as XMPMetaMBS)

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** PutXMP() updates the XMP metadata in this object without writing out the file.

Notes: This function supplies new XMP for the file. However, the disk file is not written until the object is closed with CloseFile(). The options provided when the file was opened determine if reconciliation is done with other forms of metadata.

See also:

- 2.9.1 PutXMP(xmpPacket as string) 42

2.9.2 Events**Abort as boolean**

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** This event is called periodically to allow a user to cancel time-consuming operations.

Notes:

The event should return true to signal an abort, which results in an exception being thrown.

2.9.3 Constants

kAEFilterPresetFile = & h46465820

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kAEProjectFile = & h41455020

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kAEProjTemplateFile = & h41455420

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

Notes: After Effects Project Template

kAIFFFile = & h41494646

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kAllowsOnlyXMP = & h00000020

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for GetFormatInfo

Notes: Allows access to just the XMP, ignoring other forms.

kAllowsSafeUpdate = & h00000200

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for GetFormatInfo

Notes: The file handler allows crash-safe file updates.

kArrayLastItem = -1

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The index constant for the last item.

kAVCHDFile = & h41564844

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kAVIFile = & h41564920

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

Notes: Cineon

kCanExpand = 2

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for GetFormatInfo

Notes: Can expand XMP or other metadata in an existing file.

kCanInjectXMP = 1

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for GetFormatInfo

Notes: Can inject first-time XMP into an existing file.

kCanReconcile = & h00000010

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for GetFormatInfo

Notes: Supports reconciliation between XMP and other forms.

kCanRewrite = 4

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for GetFormatInfo

Notes: Can copy one file to another, writing new metadata.

kCELFile = & h43454C20

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

Notes: Audition loop

kChar16BitBig = 2

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to define the character format.

Notes: 16-bit big-endian

kChar16BitLittle = 3

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to define the character format.

Notes: 16-bit little-endian

kChar16BitMask = 2

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to define the character format.

kChar32BitBig = 4

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to define the character format.

Notes: 32-bit big-endian

kChar32BitLittle = 5

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to define the character format.

Notes: 32-bit little-endian

kChar32BitMask = 4

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to define the character format.

kChar8Bit = 0

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to define the character format.

Notes: 8-bit

kCharLittleEndianMask = 1

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to define the character format.

kCharUnknown = 1

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to define the character format.

Notes: Variable or not-yet-known cases

kCINFile = & h43494E20

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kEncoreProjectFile = & h4E434F52

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kEPSFile = & h45505320

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

Notes: encapsulated PostScript

kFLAFile = & h464C4120

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kFLVFile = & h464C5620

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kFolderBasedFormat = & h00001000

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for GetFormatInfo

Notes: The format is folder oriented, for example the P2 video format.

kGIFFile = & h47494620

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kHandlerOwnsFile = & h00000100

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for GetFormatInfo

Notes: The file handler does the file open and close.

kHTMLFile = & h48544D4C

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kIllustratorFile = & h41492020

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kInDesignFile = & h494E4444

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kJPEG2KFile = & h4A505820

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

Notes: JPEG 2000, ISO 15444-1

kJPEGFile = & h4A504547

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kMOVFile = & h4D4F5620

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

Notes: Quicktime

kMP3File = & h4D503320

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kMPEG2File = & h4D503220

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kMPEG4File = & h4D503420

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

Notes: ISO 14494-12 and -14

kMPEGFile = & h4D504547

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kNeedsReadOnlyPacket = & h00000400

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for GetFormatInfo

Notes: The file format needs the XMP packet to be read-only.

kNoOptions = 0

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The constant to specify to use no options.

kOpenCacheTNail = 8

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for OpenFile().

Notes: Cache thumbnail if possible, TXMPFiles::GetThumbnail() will be called.

kOpenForRead = 1

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for OpenFile().

Notes: Open for read-only access.

kOpenForUpdate = 2

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for OpenFile().

Notes: Open for reading and writing.

kOpenInBackground = & h10000000

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for OpenFile().

Notes: Set if calling from background thread.

kOpenLimitedScanning = & h00000080

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for OpenFile().

Notes: Only packet scan files "known" to need scanning.

kOpenOnlyXMP = 4

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for OpenFile().

Notes: Only the XMP is wanted, allows space/time optimizations.

kOpenRepairFile = & h00000100

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for OpenFile().

Notes: Attempt to repair a file opened for update, default is to not open (throw an exception).

kOpenStrictly = & h00000010

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for OpenFile().

Notes: Be strict about locating XMP and reconciling with other forms.

kOpenUsePacketScanning = & h00000040

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for OpenFile().

Notes: Force packet scanning, do not use a smart handler.

kOpenUseSmartHandler = & h00000020

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for OpenFile().

Notes: Require the use of a smart handler.

kP2File = & h50322020

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

Notes: a collection not really a single file

kPDFFile = & h50444620

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kPhotoshopFile = & h50534420

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kPNGFile = & h504E4720

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kPostScriptFile = & h50532020

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

Notes: general PostScript following DSC conventions

kPrefersInPlace = 8

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for GetFormatInfo

Notes: Can expand, but prefers in-place update.

kPremiereProjectFile = & h5052504A

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kPremiereTitleFile = & h5052544C

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kReturnsRawPacket = & h00000040

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for GetFormatInfo

Notes: File handler returns raw XMP packet information.

kReturnsTNail = & h00000080

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for GetFormatInfo

Notes: File handler returns native thumbnail.

kSESFile = & h53455320

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

Notes: Audition session

kSonyHDVFile = & h53484456

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

Notes: a collection not really a single file

kSWFFile = & h53574620

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kTextFile = & h74657874

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kTIFFFile = & h54494646

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kUCFFile = & h55434620

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

Notes: Universal Container Format

kUnknownFile = & h20202020

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

Notes: Unknown file format constant

kUnknownLength = -1

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Constant for an unknown packet length within a file.

kUnknownOffset = -1

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Constant for an unknown packet offset within a file.

kUpdateSafely = 1

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants for the option bit flags for CloseFile().

Notes: Write into a temporary file and swap for crash safety.

kUseNullTermination = 0

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The length constants for a string to determinate the length automatically.

kUsesSidecarXMP = & h00000800

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for GetFormatInfo

Notes: The file handler uses a "sidecar" file for the XMP.

kWAVFile = & h57415620

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

kWMAVFile = & h574D4156

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

Notes: Windows Media Audio and Video

kXDCAM_EXFile = & h58444358

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

Notes: a collection not really a single file

kXDCAM_ FAMFile = & h58444346

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

Notes: a collection not really a single file

kXDCAM_ SAMFile = & h58444353

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

Notes: a collection not really a single file

kXMLFile = & h584D4C20

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants to specify a file type.

2.10 class XMPMetaMBS

class XMPMetaMBS

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The main class for the Adobe XMP SDK.

2.10.1 Methods

AppendArrayItem(schemaNS as string, arrayName as string, arrayOptions as integer, itemValue as string, options as integer=0)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Appends an item to an array.

Notes:

AppendArrayItem simplifies construction of an array by not requiring that you pre-create an empty array. The array that is assigned is created automatically if it does not yet exist. Each call to AppendArrayItem appends an item to the array. The corresponding parameters have the same use as SetArrayItem. The arrayOptions parameter is used to specify what kind of array. If the array exists, it must have the specified form.

Parameters:

schemaNS	The namespace URI for the array. Has the same usage as in GetProperty.
arrayName	The name of the array. May be a general path expression, must not be null or the empty string. Has the same namespace prefix usage as propPath in GetProperty.
arrayOptions	Option flags describing the array form. The only valid bits are those that are part of kXMP_ PropArrayFormMask: kXMP_ PropValueIsArray (& h200), kXMP_ PropArrayIsOrdered (& h400), kXMP_ PropArrayIsAlternate (& h800), or kXMP_ PropArrayIsAltText (& h1000).
itemValue	An UTF-8 string that is the value of the array item, if the array item has a value. Has the same usage as propValue in GetProperty.
itemOptions	Option flags describing the item.

ApplyTemplate(WorkingXMP as XMPMetaMBS, template as XMPMetaMBS, actions as integer)

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Applies an xmp template.

CatenateArrayItems(schemaNS as string, arrayName as string, separator as string, quotes as string, options as integer) as string

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Create a single edit string from an array of strings.

Notes:

xmpObj	The XMP object containing the array to be catenated.
schemaNS	The schema namespace URI for the array. Must not be an empty string.
arrayName	The name of the array. May be a general path expression, must not be an empty string. Each item in the array must be a simple string value.
separator	The string to be used to separate the items in the catenated string. Defaults to "; ", ASCII semicolon and space (U+003B, U+0020).
quotes	The characters to be used as quotes around array items that contain a separator. Defaults to "'", ASCII quote (U+0022).
options	Option flags to control the catenation.

returns the string with the catenated array items.

Clone as XMPMetaMBS

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Creates a copy of the XMPMeta Object.

Notes:

Deep copy, not just a new reference.
Returns nil on any error.

ComposeArrayItemPath(schemaNS as string, arrayName as string, itemIndex as integer) as string

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Compose the path expression for an item in an array.

Notes:

Returns the composed path. This will be of the form `<tt>ns:arrayName [i] </tt>`, where "ns" is the prefix for schemaNS and "i" is the decimal representation of itemIndex. If the value of itemIndex is kXMP_ArrayLastItem, the path is `<tt>ns:arrayName [last()] </tt>`.

schemaNS	The namespace URI for the array. Must not be null or the empty string.
arrayName	The name of the array. May be a general path expression, must not be an empty string.
itemIndex	The index of the desired item. Arrays in XMP are indexed from 1. The constant <code>kXMP_ArrayLastItem</code> always refers to the last existing array item.

ComposeFieldSelector(schemaNS as string, arrayName as string, fieldNS as string, fieldName as string, fieldValue as string) as string

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Compose the path expression to select an alternate item by a field's value.

Notes:

The path syntax allows two forms of "content addressing" that may be used to select an item in an array of alternatives. The form used in `ComposeFieldSelector` lets you select an item in an array of structs based on the value of one of the fields in the structs. The other form of content addressing is shown in `ComposeLangSelector`.

For example, consider a simple struct that has two fields, the name of a city and the URI of an FTP site in that city. Use this to create an array of download alternatives. You can show the user a popup built from the values of the city fields. You can then get the corresponding URI as follows:

```
path=ComposeFieldSelector ( schemaNS, "Downloads", fieldNS, "City", chosenCity)
exists = GetStructField ( schemaNS, path, fieldNS, "URI", uri )
```

schemaNS	The namespace URI for the array. Must not be null or the empty string.
arrayName	The name of the array. May be a general path expression, must not be an empty string.
fieldNS	The namespace URI for the field used as the selector. Must not be an empty string.
fieldName	The name of the field used as the selector. Must be a simple XML name, must not be an empty string. It must be the name of a field that is itself simple.
fieldValue	The desired value of the field.

Returns the string with the composed path. This will be of the form `<tt>ns:arrayName [fNS:fieldName='fieldValue'] </tt>`, where "ns" is the prefix for schemaNS and "fNS" is the prefix for fieldNS.

ComposeLangSelector(schemaNS as string, arrayName as string, langName as string) as string

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Compose the path expression to select an alternate item by language.

Notes:

The path syntax allows two forms of "content addressing" that may be used to select an item in an array of alternatives. The form used in `ComposeLangSelector` lets you select an item in an alt-text array based on the value of its `<tt>xml:lang</tt>` qualifier. The other form of content addressing is shown in `ComposeFieldSelector`.

`ComposeLangSelector` does not supplant `SetLocalizedText` or `GetLocalizedText`.

They should generally be used, as they provide extra logic to choose the appropriate language and maintain consistency with the 'x-default' value. `ComposeLangSelector` gives you an path expression that is explicitly and only for the language given in the `langName` parameter.

<code>schemaNS</code>	The namespace URI for the array. Must not be null or the empty string.
<code>arrayName</code>	The name of the array. May be a general path expression, must not be an empty string.
<code>langName</code>	The RFC 3066 code for the desired language.

Returns the composed path. This will be of the form `<tt>ns:arrayName [\@xml:lang='langName'] </tt>`, where "ns" is the prefix for `schemaNS`.

ComposeQualifierPath(schemaNS as string, structName as string, qualNS as string, qualName as string) as string

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Compose the path expression for a qualifier.

Notes:

`schemaNS` The namespace URI for the property to which the qualifier is attached. Must not be an empty string.

`propName` The name of the property to which the qualifier is attached. May be a general path expression, must not be an empty string.

`qualNS` The namespace URI for the qualifier. May be an empty string if the qualifier is in the XML empty namespace.

`qualName` The name of the qualifier. Must be a simple XML name, must not be an empty string.

Returns the composed path. This will be of the form `<tt>ns:propName/?qNS:qualName</tt>`, where "ns" is the prefix for `schemaNS` and "qNS" is the prefix for `qualNS`.

ComposeStructFieldPath(schemaNS as string, structName as string, fieldNS as string, fieldName as string) as string

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Compose the path expression for a field in a struct.

Notes:

schemaNS	The namespace URI for the struct. Must not be null or the empty string.
structName	The name of the struct. May be a general path expression, must not be an empty string.
fieldNS	The namespace URI for the field. Must not be an empty string.
fieldName	The name of the field. Must be a simple XML name, must not be an empty string.

Returns the composed path. This will be of the form `<tt>ns:structName/fNS:fieldName</tt>`, where "ns" is the prefix for schemaNS and "fNS" is the prefix for fieldNS.

Constructor

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Default constructor, creates an empty object.

See also:

- 2.10.1 Constructor(data as memoryblock, Offset as integer, Size as integer) 62
- 2.10.1 Constructor(data as string) 63

Constructor(data as memoryblock, Offset as integer, Size as integer)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Construct an object and parse one buffer of RDF into it.

Notes: This constructor creates a new TXMPMeta object and populates it with metadata from a buffer containing serialized RDF. This buffer must be a complete RDF parse stream. Pass "" to construct an empty XMPMetaMBS object. The result of an actual parse is identical to creating an empty object then calling XMPMetaMBS.ParseFromBuffer. The RDF must be complete. If you need to parse with multiple buffers, create an empty object and use XMPMetaMBS.ParseFromBuffer.

See also:

- 2.10.1 Constructor 62
- 2.10.1 Constructor(data as string) 63

Constructor(data as string)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Construct an object and parse one buffer of RDF into it.

Notes: This constructor creates a new TXMPMeta object and populates it with metadata from a buffer containing serialized RDF. This buffer must be a complete RDF parse stream. Pass "" to construct an empty XMPMetaMBS object. The result of an actual parse is identical to creating an empty object then calling XMPMetaMBS.ParseFromBuffer. The RDF must be complete. If you need to parse with multiple buffers, create an empty object and use XMPMetaMBS.ParseFromBuffer.

See also:

- 2.10.1 Constructor 62
- 2.10.1 Constructor(data as memoryblock, Offset as integer, Size as integer) 62

ConvertFromBool(value as boolean) as string

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Convert from Boolean to string.

ConvertFromDate(value as XMPDateTimeMBS) as string

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Convert from date/time to string.

ConvertFromFloat(value as double, format as string) as string

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Convert from floating point to string.

Notes: format: Optional C sprintf format for the conversion. Defaults to "% f".

ConvertFromInt(value as integer, format as string) as string

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Convert from integer to string.

Notes: format: Optional C sprintf format for the conversion. Defaults to "% d".

ConvertFromInt64double(value as double, format as string) as string

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Convert from integer to string.

Notes: format: Optional C sprintf format for the conversion. Defaults to "% d".

ConvertToBool(value as string) as boolean

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Convert from string to Boolean.

ConvertToDate(value as string) as XMPDateTimeMBS

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Convert from string to date/time.

ConvertToFloat(value as string) as double

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Convert from string to floating point.

ConvertToInt(value as string) as integer

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Convert from string to integer.

ConvertToInt64double(value as string) as double

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Convert from string to 64 bit integer.

CountArrayItems(schemaNS as string, arrayName as string) as integer

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Counts array items of given array.

Notes: Returns 0 on any error.

CurrentDateTime as XMPDateTimeMBS

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Current date and time as a XMPDateTimeMBS object.

Notes: Returns nil on any error.

DecodeFromBase64(text as string) as string

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Decode from Base64 encoded string to raw data.

Example:

```
dim x as XMPMetaMBS
x=new XMPMetaMBS
MsgBox x.DecodeFromBase64("dGVzdA==") // test
```

Notes: This is a global method which does not need a valid handle.

DeleteArrayItem(schemaNS as string, arrayName as string, itemIndex as integer)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** DeleteArrayItem deletes the given XMP subtree rooted at the given array item.

Notes:

It is not an error if the array item does not exist.

Parameters:

<code>schemaNS</code>	The namespace URI for the array. Has the same usage as in <code>GetProperty</code> .
<code>arrayName</code>	The name of the array. May be a general path expression, must not <code>""</code> . Has the same namespace prefix usage as <code>propName</code> in <code>GetProperty</code> .
<code>itemIndex</code>	The index of the desired item. Arrays in XMP are indexed from 1. The constant <code>kXMP_ArrayLastItem</code> (-1) always refers to the last existing array item.

DeleteLocalizedText(`schemaNS` as string="", `altTextName` as string="", `genericLang` as string="", `specificLang` as string="")

Plugin Version: 11.0 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Deletes localized text.

DeleteNamespace(`namespaceURI` as string)

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Deletes a namespace from the registry.

Notes:

Not implemented?

Does nothing if the URI is not registered, or if the parameter is null or the empty string.

`namespaceURI`: The URI for the namespace.

DeleteProperty(`schemaNS` as string, `propName` as string)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** `DeleteProperty` deletes the given XMP subtree rooted at the given property.

Notes:

It is not an error if the property does not exist.

Parameters:

schemaNS The namespace URI for the property. Has the same usage as in GetProperty.
propName The name of the property. Has the same usage as in GetProperty.

DeleteQualifier(schemaNS as string, structName as string, qualNS as string, qualName as string)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** DeleteQualifier deletes the given XMP subtree rooted at the given qualifier.

Notes:

It is not an error if the qualifier does not exist.

Parameters:

schemaNS The namespace URI for the struct. Has the same usage as in GetProperty.
propName The name of the property to which the qualifier is attached. Has the same usage as in GetProperty.
qualNS The namespace URI for the qualifier. Has the same URI and prefix usage as the schemaNS parameter.
qualName The name of the qualifier. Must be a single XML name, must not be "" Has the same namespace prefix usage as the propName parameter.

DeleteStructField(schemaNS as string, structName as string, fieldNS as string, fieldName as string)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** DeleteStructField deletes the given XMP subtree rooted at the given struct field.

Notes:

It is not an error if the field does not exist.

Parameters:

schemaNS	The namespace URI for the struct. Has the same usage as in <code>GetProperty</code> .
structName	The name of the struct. May be a general path expression, must not be <code>""</code> . Has the same namespace prefix usage as <code>propName</code> in <code>GetProperty</code> .
fieldNS	The namespace URI for the field. Has the same URI and prefix usage as the <code>schemaNS</code> parameter.
fieldName	The name of the field. Must be a single XML name, must not be <code>""</code> . Has the same namespace prefix usage as the <code>structName</code> parameter.

DoesArrayItemExist(schemaNS as string, arrayName as string, itemIndex as integer) as boolean

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** `DoesArrayItemExist` tells if the array item exists.

Notes:

Returns true if the array item exists.

Parameters:

schemaNS	The namespace URI for the array. Has the same usage as in <code>GetProperty</code> .
arrayName	The name of the array. May be a general path expression, must not be <code>""</code> . Has the same namespace prefix usage as <code>propName</code> in <code>GetProperty</code> .
itemIndex	The index of the desired item. Arrays in XMP are indexed from 1. The constant <code>kXMP_ArrayLastItem</code> (-1) always refers to the last existing array item.

DoesPropertyExist(schemaNS as string, propName as string) as boolean

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** `DoesPropertyExist` tells if the property exists.

Notes:

Returns true if the property exists.

Parameters:

schemaNS	The namespace URI for the property. Has the same usage as in <code>GetProperty</code> .
propName	The name of the property. Has the same usage as in <code>GetProperty</code> .

DoesQualifierExist(schemaNS as string, structName as string, qualNS as string, qualName as string) as boolean

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** DoesQualifierExist tells if the qualifier exists.

Notes:

Returns true if the qualifier exists.

Parameters:

schemaNS	The namespace URI for the struct. Has the same usage as in GetProperty.
propName	The name of the property to which the qualifier is attached. Has the same usage as in GetProperty.
qualNS	The namespace URI for the qualifier. Has the same URI and prefix usage as the schemaNS parameter.
qualName	The name of the qualifier. Must be a single XML name, must not be "". Has the same namespace prefix usage as the propName parameter.

DoesStructFieldExist(schemaNS as string, structName as string, fieldNS as string, fieldName as string) as boolean

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** DoesStructFieldExist tells if the struct field exists.

Notes:

Returns true if the field exists.

Parameters:

schemaNS	The namespace URI for the struct. Has the same usage as in GetProperty.
structName	The name of the struct. May be a general path expression, must not be "". Has the same namespace prefix usage as propName in GetProperty.
fieldNS	The namespace URI for the field. Has the same URI and prefix usage as the schemaNS parameter.
fieldName	The name of the field. Must be a single XML name, must not be "". Has the same namespace prefix usage as the structName parameter.

DumpNamespaces(output as XMPTextOutputMBS) as integer

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** DumpNamespaces dumps the list of registered namespace URIs and prefixes.

Notes:

This is a global method which does not need a valid handle.
Returns status code. (0=success and -1=error)

DumpObject(output as XMPTextOutputMBS) as integer

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** DumpObject dumps the content of an XMP object.

Notes: Returns status code. (0=success and -1=error)

DuplicateSubtree(dest as XMPMetaMBS, sourceNS as string, sourceRoot as string, destNS as string="", destRoot as string="", options as integer=0)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Replicate a subtree from one XMP object into another, possibly at a different location.

Notes:

Parameters:

self	The source XMP object.
dest	The destination XMP object.
sourceNS	The schema namespace URI for the source subtree.
sourceRoot	The root location for the source subtree. May be a general path expression, must not be null or the empty string.
destNS	The schema namespace URI for the destination. Defaults to the source namespace.
destRoot	The root location for the destination. May be a general path expression. Defaults to the source location.
options	Option flags to control the separation.

EncodeToBase64(text as string) as string

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Convert from raw data to Base64 encoded string.

Example:

```
dim x as XMPMetaMBS
x=new XMPMetaMBS
MsgBox x.EncodeToBase64("test") // dGVzdA==
```

Notes: This is a global method which does not need a valid handle.

Erase

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Restores the object to a "just constructed" state.

GetArrayItem(schemaNS as string, arrayName as string, itemIndex as integer, byref item-Value as string, byref options as integer) as boolean

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** GetArrayItem provides access to items within an array.

Notes:

The index is passed as an integer, you need not worry about the path string syntax for array items, convert a loop index to a string, etc.

Returns true if the array item exists.

Parameters:

schemaNS	The namespace URI for the array. Has the same usage as in <code>GetProperty</code> .
arrayName	The name of the array. May be a general path expression, must not be <code>""</code> . Has the same namespace prefix usage as <code>propName</code> in <code>GetProperty</code> .
itemIndex	The index of the desired item. Arrays in XMP are indexed from 1. The constant <code>kXMP_ArrayLastItem</code> (-1) always refers to the last existing array item.
itemValue	A string that is assigned the value of the array item, if the array item has a value. Arrays and non-leaf levels of structs do not have values.
options	An integer variable that is assigned option flags describing the array item.

GetLocalizedText(schemaNS as string, altTextName as string, genericLang as string, specificLang as string, byref actualLang as string, byref itemValue as string, byref options as integer) as boolean

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** `GetLocalizedText` returns information about a selected item in an alt-text array.

Notes:

The array item is selected according to the rules given above.

Returns true if an appropriate array item exists.

Parameters:

schemaNS	The namespace URI for the alt-text array. Has the same usage as in <code>GetProperty</code> .
altTextName	The name of the alt-text array. May be a general path expression, must not be <code>""</code> . Has the same namespace prefix usage as <code>propName</code> in <code>GetProperty</code> .
genericLang	The name of the generic language as an RFC 3066 primary subtag. May be <code>""</code> if no generic language is wanted.
specificLang	The name of the specific language as an RFC 3066 tag. Must not be null or the empty string.
actualLang	A string that is assigned the language of the selected array item, if an appropriate array item is found.
itemValue	A string that is assigned the value of the array item, if an appropriate array item is found.
options	An integer variable that is assigned option flags describing the array item.

GetNamespacePrefix(namespaceURI as string, byref namespacePrefix as string) as boolean

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Obtain the prefix for a registered namespace URI.

Notes:

It is not an error if the namespace URI is not registered. The output namespacePrefix string is "" if the namespace URI is not registered.

Parameters:

namespaceURI The URI for the namespace. Must not be null or the empty string.
namespacePrefix Returns the prefix registered for this URI, with a terminating ':

Returns true if the namespace URI is registered.

This is a global method which does not need a valid handle.

GetNamespaceURI(namespacePrefix as string, byref namespaceURI as string) as boolean

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Obtain the URI for a registered namespace prefix.

Notes:

It is not an error if the namespace prefix is not registered. The output namespaceURI string is "" if the namespace prefix is not registered.

Parameters:

namespacePrefix The prefix for the namespace. Must not be "".
namespaceURI Returns the URI registered for this prefix.

Returns true if the namespace prefix is registered.

This is a global method which does not need a valid handle.

GetProperty(schemaNS as string, propName as string, byref propValue as string, byref options as integer) as boolean

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** GetProperty is the simplest property getter, mainly for top level simple properties or after using the path composition functions.

Notes:

Returns true if the property exists.

Parameters:

schemaNS	The namespace URI for the property. May be an empty string if the first component of the propName path contains a namespace prefix. The URI must be for a registered namespace.
propName	The name of the property. May be a general path expression, must not be an empty string. Using a namespace prefix on the first component is optional. If present without a schemaNS value then the prefix specifies the namespace. The prefix must be for a registered namespace. If both a schemaNS URI and propName prefix are present, they must be corresponding parts of a registered namespace.
propValue	A string that is assigned the value of the property, if the property has a value. Arrays and non-leaf levels of structs do not have values.
options	An integer variable that is assigned option flags describing the property.

GetPropertyBoolean(schemaNS as string, propName as string, byref propValue as boolean) as boolean

Plugin Version: 11.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** GetProperty-Boolean returns the value of a Boolean property as a boolean.

Notes:

Returns true if the property exists.

Parameters:

See also:

- 2.10.1 GetPropertyBoolean(schemaNS as string, propName as string, byref propValue as boolean, byref

schemaNS	The namespace URI for the property. Has the same usage as in <code>GetProperty</code> .
propName	The name of the property. Has the same usage as in <code>GetProperty</code> .
propValue	A boolean variable that is assigned the value of the property.
options	Optional, an integer variable that is assigned option flags describing the property.

options as integer) as boolean

75

GetPropertyBoolean(schemaNS as string, propName as string, byref propValue as boolean, byref options as integer) as boolean

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** `GetPropertyBoolean` returns the value of a Boolean property as a boolean.

Notes:

Returns true if the property exists.

Parameters:

schemaNS	The namespace URI for the property. Has the same usage as in <code>GetProperty</code> .
propName	The name of the property. Has the same usage as in <code>GetProperty</code> .
propValue	A boolean variable that is assigned the value of the property.
options	Optional, an integer variable that is assigned option flags describing the property.

See also:

- 2.10.1 `GetPropertyBoolean(schemaNS as string, propName as string, byref propValue as boolean) as boolean` 74

GetPropertyDate(schemaNS as string, propName as string, byref propValue as XMPDateTimeMBS, byref options as integer) as boolean

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** `GetPropertyDate` returns the value of a date/time property as an `XMPDateTimeMBS` object.

Notes:

Returns true if the property exists.

Parameters:

schemaNS	The namespace URI for the property. Has the same usage as in <code>GetProperty</code> .
propName	The name of the property. Has the same usage as in <code>GetProperty</code> .
propValue	A <code>XMPDateTimeMBS</code> variable that is assigned the value of the property.
options	An integer variable that is assigned option flags describing the property.

GetPropertyFloat(schemaNS as string, propName as string, byref propValue as double) as boolean

Plugin Version: 11.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** `GetPropertyFloat` returns the value of a floating point property as a double value.

Notes:

Returns true if the property exists.

Parameters:

schemaNS	The namespace URI for the property. Has the same usage as in <code>GetProperty</code> .
propName	The name of the property. Has the same usage as in <code>GetProperty</code> .
propValue	A double variable that is assigned the value of the property.
options	Optional, an integer variable that is assigned option flags describing the property.

See also:

- 2.10.1 `GetPropertyFloat(schemaNS as string, propName as string, byref propValue as double, byref options as integer) as boolean` 76

GetPropertyFloat(schemaNS as string, propName as string, byref propValue as double, byref options as integer) as boolean

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** `GetPropertyFloat` returns the value of a floating point property as a double value.

Notes:

Returns true if the property exists.

Parameters:

schemaNS	The namespace URI for the property. Has the same usage as in <code>GetProperty</code> .
propName	The name of the property. Has the same usage as in <code>GetProperty</code> .
propValue	A double variable that is assigned the value of the property.
options	Optional, an integer variable that is assigned option flags describing the property.

See also:

- 2.10.1 `GetPropertyFloat(schemaNS as string, propName as string, byref propValue as double) as boolean` 76

GetPropertyInt64Double(schemaNS as string, propName as string, byref propValue as double, byref options as integer) as boolean

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** `GetPropertyInt64Double` returns the value of a 64 bit integer property as a double value.

Notes:

Returns true if the property exists.

Parameters:

schemaNS	The namespace URI for the property. Has the same usage as in <code>GetProperty</code> .
propName	The name of the property. Has the same usage as in <code>GetProperty</code> .
propValue	A double variable that is assigned the value of the property.
options	An integer variable that is assigned option flags describing the property.

GetPropertyInteger(schemaNS as string, propName as string, byref propValue as integer) as boolean

Plugin Version: 11.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** `GetPropertyInteger` returns the value of an integer property as an integer.

Notes:

Returns true if the property exists.

Parameters:

schemaNS	The namespace URI for the property. Has the same usage as in <code>GetProperty</code> .
propName	The name of the property. Has the same usage as in <code>GetProperty</code> .
propValue	An integer variable that is assigned the value of the property.
options	Optional, an integer variable that is assigned option flags describing the property.

See also:

- 2.10.1 `GetPropertyInteger(schemaNS as string, propName as string, byref propValue as integer, byref options as integer) as boolean` 78

GetPropertyInteger(schemaNS as string, propName as string, byref propValue as integer, byref options as integer) as boolean

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** `GetPropertyInteger` returns the value of an integer property as an integer.

Notes:

Returns true if the property exists.

Parameters:

schemaNS	The namespace URI for the property. Has the same usage as in <code>GetProperty</code> .
propName	The name of the property. Has the same usage as in <code>GetProperty</code> .
propValue	An integer variable that is assigned the value of the property.
options	Optional, an integer variable that is assigned option flags describing the property.

See also:

- 2.10.1 `GetPropertyInteger(schemaNS as string, propName as string, byref propValue as integer) as boolean` 77

GetPropertyInteger64(schemaNS as string, propName as string, byref propValue as Int64, byref options as integer) as boolean

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Retrieves the value of an integer property as an Int64.

Notes:

Reports whether a property exists, and retrieves its binary value and property type information.

schemaNS: The namespace URI; see GetProperty().

propName: The name of the property. Can be a general path expression, must not be empty string; see GetProperty() for namespace prefix usage.

propValue: A variable in which to return the binary value.

options: A variable in which to return the option flags describing the property, a logical OR of allowed bit-flag constants; see kPropValueIsStruct and following.

Returns true if the property exists.

GetQualifier(schemaNS as string, propName as string, qualNS as string, qualName as string, byref qualValue as string, byref options as integer) as boolean

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** GetQualifier provides access to a qualifier attached to a property.

Notes:

The namespace for the qualifier is passed as a URI, you need not worry about the path string syntax. In many regards qualifiers are like struct fields.

The names of qualifiers should be XML qualified names, that is within an XML namespace. The path syntax for a qualified name uses the namespace prefix. This is unreliable since the prefix is never guaranteed. The URI is the formal name, the prefix is just a local shorthand in a given sequence of XML text.

Note: Qualifiers are only supported for simple leaf properties at this time. (in the XMP SDK)

Returns true if the qualifier exists.

Parameters:

schemaNS	The namespace URI for the struct. Has the same usage as in <code>GetProperty</code> .
propName	The name of the property to which the qualifier is attached. May be a general path expression, must not be an empty string. Has the same namespace prefix usage as in <code>GetProperty</code> .
qualNS	The namespace URI for the qualifier. Has the same URI and prefix usage as the <code>schemaNS</code> parameter.
qualName	The name of the qualifier. Must be a single XML name, must not be an empty string. Has the same namespace prefix usage as the <code>propName</code> parameter.
qualValue	A string variable that is assigned the value of the qualifier, if the qualifier has a value. Arrays and non-leaf levels of structs do not have values.
options	An integer variable that is assigned option flags describing the qualifier.

GetStructField(`schemaNS` as string, `structName` as string, `fieldNS` as string, `fieldName` as string, `byref itemValue` as string, `byref options` as integer) as boolean

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** `GetStructField` provides access to fields within a nested structure.

Notes:

The namespace for the field is passed as a URI, you need not worry about the path string syntax.

The names of fields should be XML qualified names, that is within an XML namespace. The path syntax for a qualified name uses the namespace prefix. This is unreliable since the prefix is never guaranteed. The URI is the formal name, the prefix is just a local shorthand in a given sequence of XML text.

Returns true if the field exists.

Parameters:

schemaNS	The namespace URI for the struct. Has the same usage as in <code>GetProperty</code> .
structName	The name of the struct. May be a general path expression, must not be an empty string. Has the same namespace prefix usage as <code>propName</code> in <code>GetProperty</code> .
fieldNS	The namespace URI for the field. Has the same URI and prefix usage as the <code>schemaNS</code> parameter.
fieldName	The name of the field. Must be a single XML name, must not be an empty string. Has the same namespace prefix usage as the <code>structName</code> parameter.
fieldValue	An string variable that is assigned the value of the field, if the field has a value. Arrays and non-leaf levels of structs do not have values.
options	An integer variable that is assigned option flags describing the field. May be null if the flags are not wanted.

GetVersionInfo as XMPVersionInfoMBS

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Retrieves runtime version information.

GlobalOptions as integer

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Sets or retrieves the set of global option flags.

Notes: (Read and Write computed property)

Iterator(schemaNS as string, propName as string, options as integer) as XMPIteratorMBS

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Creates an iterator for the properties within an XMP object.

Notes:

Returns nil on failure.

schemaNS: Optional schema namespace URI to restrict the iteration. Omitted (visit all schema) by passing "".

propName: Optional property name to restrict the iteration. May be an arbitrary path expression. Omitted (visit all properties) by passing "". If not empty a schema URI must also be provided.

options: Option flags to control the iteration.

The available option flags are:

kXMP_ IterJustChildren	= & h100	Just visit the immediate children of the root, default is subtree.
kXMP_ IterJustLeafNodes	= & h200	Just visit the leaf nodes, default visits all nodes.
kXMP_ IterJustLeafName	= & h400	Return just the leaf part of the path, default is the full path.
kXMP_ IterOmitQualifiers	= & h1000	Omit all qualifiers.

MergeFromJPEG(extendedXMP as XMPMetaMBS)

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** merges standard and extended XMP retrieved from a JPEG file.

Notes:

When an extended partition stores properties that do not fit into the JPEG file limitation of 64K bytes, this function integrates those properties back into the same XMP object with those from the standard XMP packet.

self: An XMP object which the caller has initialized from the standard XMP packet in a JPEG file. The extended XMP is added to this object.

extendedXMP: An XMP object which the caller has initialized from the extended XMP packet in a JPEG file.

Name as string

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The object name.

Notes: (Read and Write computed property)

PackageForJPEG(byref standardXMP as string, byref extendedXMP as string, byref extendedDigest as string)

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** creates XMP serializations appropriate for a JPEG file.

Notes:

The standard XMP in a JPEG file is limited to 64K bytes. This function serializes the XMP metadata in

an XMP object into a string of RDF (see `\c SerializeToBuffer`) the data does not fit into the 64K byte limit, it creates a second packet string with the extended data.

`self`: The XMP object containing the metadata.

`standardXMP`: A string object in which to return the full standard XMP packet.

`extendedXMP`: A string object in which to return the serialized extended XMP, empty if not needed.

`extendedDigest`: A string object in which to return an MD5 digest of the serialized extended XMP, empty if not needed.

ParseFromBuffer(buffer as string, options as integer=0)

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** ParseFromBuffer parses RDF from a series of input buffers.

Notes:

The buffers may be any length. The buffer boundaries need not respect XML tokens or even Unicode characters.

Parameters:

`buffer` Input data buffer. Termination of an input loop is convenient by passing `kXMP_ParseMoreBuffers (2)` for all real input, then having a final call with a zero length and `kXMP_NoOptions`.

`options` Options controlling the parsing.

The available options are:

<code>kXMP_ParseMoreBuffers</code>	= 2	This is not the last buffer of input, more calls follow.
<code>kXMP_RequireXMPMeta</code>	= 1	The <code>x:xmpmeta</code> XML element is required around <code>rdf:RDF</code> .
<code>kXMP_StrictAliasing</code>	= 4	Do not reconcile alias differences, throw an exception.

Note: The `kXMP_StrictAliasing` option is not yet implemented.

RegisterNamespace(namespaceURI as string, suggestedPrefix as string, byref registeredPrefix as string) as boolean

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Register a namespace URI with a suggested prefix.

Example:

```
dim x as new XMPMetaMBS
dim registeredPrefix as string

call x.RegisterNamespace ("xwnv", "xwnv", registeredPrefix)

x.SetProperty "xwnv", "MZSTID", "test"

MsgBox x.SerializeToBuffer
```

Notes:

It is not an error if the URI is already registered, no matter what the prefix is. If the URI is not registered but the suggested prefix is in use, a unique prefix is created from the suggested one. The actual registered prefix is always returned. The function result tells if the registered prefix is the suggested one.

Parameters:

namespaceURI	The URI for the namespace. Must be a valid XML URI.
suggestedPrefix	The suggested prefix to be used if the URI is not yet registered. Must be a valid XML name.
registeredPrefix	Returns the prefix actually registered for this URI.

Returns true if the registered prefix matches the suggested prefix.

Note: No checking is presently done on either the URI or the prefix.

This is a global method which does not need a valid handle.

RemoveProperties(schemaNS as string="", propName as string="", options as integer=0)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Remove multiple properties from an XMP object.

Notes:

RemoveProperties was created to support the File Info dialog's Delete button, and has been generalized somewhat from those specific needs. It operates in one of three main modes depending on the schemaNS and propName parameters:

Non-empty schemaNS and propName - The named property is removed if it is an external property, or if the kXMPUI_ DoAllProperties option is passed. It does not matter whether the named property is an actual property or an alias.

Non-empty schemaNS and empty propName - The all external properties in the named schema are removed. Internal properties are also removed if the kXMPUI_ DoAllProperties option is passed. In addition, aliases from the named schema will be removed if the kXMPUI_ IncludeAliases option is passed.

Empty schemaNS and empty propName - All external properties in all schema are removed. Internal properties are also removed if the kXMPUI_ DoAllProperties option is passed. Aliases are implicitly handled because the associated actuals are.

It is an error to pass an empty schemaNS and non-empty propName.

xmpObj	The XMP object containing the properties to be removed.
schemaNS	Optional schema namespace URI for the properties to be removed.
propName	Optional path expression for the property to be removed.
options	Option flags to control the deletion. The defined flags are: kXMPUI_ DoAllProperties - Do internal properties in addition to external properties. kXMPUI_ IncludeAliases - Include aliases in the "named schema" case above.

SendAssertNotify(message as string)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** SendAssertNotify sends an assert notification with the given message.

Notes: This is a static method and can be used even with handle being 0.

SeparateArrayItems(schemaNS as string, arrayName as string, options as integer, catedStr as string)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Separate a single edit string into an array of strings.

Notes:

Parameters:

xmpObj	The XMP object containing the array to be updated.
schemaNS	The schema namespace URI for the array. Must not be null or the empty string.
arrayName	The name of the array. May be a general path expression, must not an empty string. Each item in the array must be a simple string value.
options	Option flags to control the separation.
catedStr	The string to be separated into the array items.

SerializeToBuffer(options as integer, padding as integer, newline as string, indent as string="", baseIndent as integer=0) as string

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** SerializeToBuffer serializes an XMP object into a string as RDF.

Notes:

Returns the serialized RDF.

Parameters:

options	Option flags to control the serialization.
padding	The amount of padding to be added if a writeable XML packet is created. If zero is passed (the default) an appropriate amount of padding is computed.
newline	The string to be used as a line terminator. If empty it defaults to linefeed, U+000A, the standard XML newline.
indent	The string to be used for each level of indentation in the serialized RDF. If empty it defaults to two ASCII spaces, U+0020.
baseIndent	The number of levels of indentation to be used for the outermost XML element in the serialized RDF. This is convenient when embedding the RDF in other text.

The available option flags are:

The specified options must be logically consistent, an exception is thrown if not. You cannot specify both `kXMP_ OmitPacketWrapper` along with `kXMP_ ReadOnlyPacket`, `kXMP_ IncludeThumbnailPad`, or `kXMP_ ExactPacketLength`.

In addition, one of the following encoding options may be included:

kXMP_ OmitPacketWrapper	= & h10	Do not include an XML packet wrapper.
kXMP_ ReadOnlyPacket	= & h20	Create a read-only XML packet wrapper.
kXMP_ UseCompactFormat	= & h40	Use a highly compact RDF syntax and layout.
kXMP_ WriteAliasComments	= & h400	Include XML comments for aliases.
kXMP_ IncludeThumbnailPad	= & h100	Include typical space for a JPEG thumbnail in the padding if no xmp:Thumbnails property is present.
kXMP_ ExactPacketLength	= & h200	The padding parameter provides the overall packet length. The actual amount of padding is computed. An exception is thrown if the packet exceeds this length with no padding.
kXMP_ EncodeUTF8	= 0	Encode as UTF-8, the default.
kXMP_ EncodeUTF16Big	= 2	Encode as big-endian UTF-16.
kXMP_ EncodeUTF16Little	= 3	Encode as little-endian UTF-16.
kXMP_ EncodeUTF32Big	= 4	Encode as big-endian UTF-32.
kXMP_ EncodeUTF32Little	= 5	Encode as little-endian UTF-32.

See also:

- 2.10.1 SerializeToBuffer(options as integer=0, padding as integer=0) as string

87

SerializeToBuffer(options as integer=0, padding as integer=0) as string

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** SerializeToBuffer serializes an XMP object into a string as RDF.

Example:

```
dim x as new XMPMetaMBS
dim registeredPrefix as string

call x.RegisterNamespace ("xwv", "xwv", registeredPrefix)

x.SetProperty "xwv", "MZSTID", "test"

MsgBox x.SerializeToBuffer
```

Notes: Same as other SerializeToBuffer method, but with newline="", indent=0 and baseIndent=0.

See also:

- 2.10.1 SerializeToBuffer(options as integer, padding as integer, newline as string, indent as string="", baseIndent as integer=0) as string

86

SetArrayItem(schemaNS as string, arrayName as string, itemIndex as integer, itemValue as string, options as integer=0)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** SetArrayItem provides access to items within an array.

Notes:

The index is passed as an integer, you need not worry about the path string syntax for array items, convert a loop index to a string, etc. The array passed to SetArrayItem must already exist. See also AppendArrayItem.

In normal usage the selected array item is modified. A new item is automatically appended if the index is the array size plus 1. A new item may be inserted before or after any item by using one of the following option flags:

kXMP_InsertBeforeItem = & h4000 Insert a new array item before the selected one.
kXMP_InsertAfterItem = & h8000 Insert a new array item after the selected one.

Parameters:

schemaNS	The namespace URI for the array. Has the same usage as in GetProperty.
arrayName	The name of the array. May be a general path expression, must not be an empty string. Has the same namespace prefix usage as propName in GetProperty.
itemIndex	The index of the desired item. Arrays in XMP are indexed from 1. The constant kXMP_ArrayLastItem always refers to the last existing array item.
itemValue	An UTF-8 string that is the value of the array item, if the array item has a value. Has the same usage as propValue in GetProperty.
options	Option flags describing the item. See the earlier description.

SetLocalizedText(schemaNS as string, altTextName as string, genericLang as string, specificLang as string, itemValue as string, options as integer=0)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** SetLocalizedText modifies the value of a selected item in an alt-text array.

Notes:

Creates an appropriate array item if necessary, and handles special cases for the x-default item.

If the selected item is from a match with the specific language, the value of that item is modified. If the existing value of that item matches the existing value of the x-default item, the x-default item is also modified. If the array only has 1 existing item (which is not x-default), an x-default item is added with the given

value.

If the selected item is from a match with the generic language and there are no other generic matches, the value of that item is modified. If the existing value of that item matches the existing value of the x-default item, the x-default item is also modified. If the array only has 1 existing item (which is not x-default), an x-default item is added with the given value.

If the selected item is from a partial match with the generic language and there are other partial matches, a new item is created for the specific language. The x-default item is not modified.

If the selected item is from the last 2 rules then a new item is created for the specific language. If the array only had an x-default item, the x-default item is also modified. If the array was empty, items are created for the specific language and x-default.

Parameters:

schemaNS	The namespace URI for the alt-text array. Has the same usage as in SetProperty.
altTextName	The name of the alt-text array. May be a general path expression, must not be an empty string. Has the same namespace prefix usage as propName in SetProperty.
genericLang	The name of the generic language as an RFC 3066 primary subtag.
specificLang	The name of the specific language as an RFC 3066 tag. Must not be an empty string.
itemValue	An UTF-8 string that is the new value for the appropriate array item.
options	Option flags, none are defined at present.

SetProperty(schemaNS as string, propName as string, propValue as string, options as integer=0)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** SetProperty is the simplest property setter, mainly for top level simple properties or after using the path composition functions.

Notes:

Parameters:

schemaNS	The namespace URI for the property. Has the same usage as in SetProperty.
propName	The name of the property. Has the same usage as in SetProperty.
propValue	An UTF-8 string that is the value of the property, if the property has a value. Arrays and non-leaf levels of structs do not have values.
options	Option flags describing the property.

SetPropertyBoolean(schemaNS as string, propName as string, propValue as boolean)

Plugin Version: 11.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** SetPropertyBoolean sets the value of a Boolean property from a boolean.

Notes:

Parameters:

schemaNS	The namespace URI for the property. Has the same usage as in SetProperty.
propName	The name of the property. Has the same usage as in SetProperty.
propValue	The bool value to be assigned to the property.
options	An integer variable that is assigned option flags describing the property.

See also:

- 2.10.1 SetPropertyBoolean(schemaNS as string, propName as string, propValue as boolean, options as integer) 90

SetPropertyBoolean(schemaNS as string, propName as string, propValue as boolean, options as integer)

Plugin Version: 11.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** SetPropertyBoolean sets the value of a Boolean property from a boolean.

Notes:

Parameters:

schemaNS	The namespace URI for the property. Has the same usage as in SetProperty.
propName	The name of the property. Has the same usage as in SetProperty.
propValue	The bool value to be assigned to the property.
options	An integer variable that is assigned option flags describing the property.

See also:

- 2.10.1 SetPropertyBoolean(schemaNS as string, propName as string, propValue as boolean) 90

SetPropertyDate(schemaNS as string, propName as string, propValue as XMPDateTimeMBS, options as integer=0)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** SetPropertyDate sets the value of a date/time property from an XMPDateTimeMBS object.

Notes:

Parameters:

schemaNS	The namespace URI for the property. Has the same usage as in SetProperty.
propName	The name of the property. Has the same usage as in SetProperty.
propValue	The XMPDateTimeMBS object to be assigned to the property.
options	Option flags describing the property.

SetPropertyFloat(schemaNS as string, propName as string, propValue as double)

Plugin Version: 11.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** SetPropertyFloat sets the value of a floating point property from a double.

Notes:

Parameters:

schemaNS	The namespace URI for the property. Has the same usage as in SetProperty.
propName	The name of the property. Has the same usage as in SetProperty.
propValue	The double float value to be assigned to the property.
options	Option flags describing the property.

See also:

- 2.10.1 SetPropertyFloat(schemaNS as string, propName as string, propValue as double, options as integer) 91

SetPropertyFloat(schemaNS as string, propName as string, propValue as double, options as integer)

Plugin Version: 11.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** SetPropertyFloat sets the value of a floating point property from a double.

Notes:

Parameters:

schemaNS	The namespace URI for the property. Has the same usage as in <code>GetProperty</code> .
propName	The name of the property. Has the same usage as in <code>GetProperty</code> .
propValue	The double float value to be assigned to the property.
options	Option flags describing the property.

See also:

- 2.10.1 `SetPropertyFloat(schemaNS as string, propName as string, propValue as double)` 91

SetPropertyInt64Double(schemaNS as string, propName as string, propValue as double, options as integer=0)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** `SetPropertyInt64Double` sets the value of a 64 bit integer property from a double.

Notes:

Parameters:

schemaNS	The namespace URI for the property. Has the same usage as in <code>GetProperty</code> .
propName	The name of the property. Has the same usage as in <code>GetProperty</code> .
propValue	The double value to be assigned to the property.
options	Option flags describing the property.

SetPropertyInteger(schemaNS as string, propName as string, propValue as integer)

Plugin Version: 11.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** `SetPropertyInteger` sets the value of an integer property from an integer.

Notes:

Parameters:

schemaNS	The namespace URI for the property. Has the same usage as in <code>GetProperty</code> .
propName	The name of the property. Has the same usage as in <code>GetProperty</code> .
propValue	The long integer value to be assigned to the property.
options	Option flags describing the property.

See also:

- 2.10.1 SetPropertyInteger(schemaNS as string, propName as string, propValue as integer, options as integer) 93

SetPropertyInteger(schemaNS as string, propName as string, propValue as integer, options as integer)

Plugin Version: 11.3 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** SetPropertyInteger sets the value of an integer property from an integer.

Notes:

Parameters:

schemaNS	The namespace URI for the property. Has the same usage as in SetProperty.
propName	The name of the property. Has the same usage as in SetProperty.
propValue	The long integer value to be assigned to the property.
options	Option flags describing the property.

See also:

- 2.10.1 SetPropertyInteger64(schemaNS as string, propName as string, propValue as integer) 92

SetPropertyInteger64(schemaNS as string, propName as string, propValue as Int64, options as integer=0)

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Sets the value of an integer property using an Int64.

Notes:

Sets a property with a binary value, creating it if necessary.

schemaNS: The namespace URI; see SetProperty().

propName: The name of the property. Can be a general path expression, must not be "" or the empty string; see SetProperty() for namespace prefix usage.

propValue: The new binary value.

options: Option flags describing the property; a logical OR of allowed bit-flag constants; see kPropValueIsStruct and following. Must match the type of a property that already exists.

SetQualifier(schemaNS as string, propName as string, qualNS as string, qualName as string, qualValue as string, options as integer=0)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** SetQualifier provides access to a qualifier attached to a property.

Notes:

The namespace for the qualifier is passed as a URI, you need not worry about the path string syntax. In many regards qualifiers are like struct fields. See the introductory discussion of qualified properties for more information.

The names of qualifiers should be XML qualified names, that is within an XML namespace. The path syntax for a qualified name uses the namespace prefix, which is unreliable because the prefix is never guaranteed. The URI is the formal name, the prefix is just a local shorthand in a given sequence of XML text.

Parameters:

schemaNS	The namespace URI for the struct. Has the same usage as in GetProperty.
propName	The name of the property to which the qualifier is attached. Has the same usage as in GetProperty.
qualNS	The namespace URI for the qualifier. Has the same URI and prefix usage as the schemaNS parameter.
qualName	The name of the qualifier. Must be a single XML name, must not be an empty string. Has the same namespace prefix usage as the propName parameter.
qualValue	An UTF-8 string that is the value of the qualifier, if the qualifier has a value. Has the same usage as propValue in GetProperty.
options	Option flags describing the qualifier.

SetStructField(schemaNS as string, structName as string, fieldNS as string, fieldName as string, fieldValue as string, options as integer=0)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** SetStructField provides access to fields within a nested structure.

Notes:

The namespace for the field is passed as a URI, you need not worry about the path string syntax.

The names of fields should be XML qualified names, that is within an XML namespace. The path syntax for a qualified name uses the namespace prefix, which is unreliable because the prefix is never guaranteed.

The URI is the formal name, the prefix is just a local shorthand in a given sequence of XML text.

Parameters:

schemaNS	The namespace URI for the struct. Has the same usage as in <code>GetProperty</code> .
structName	The name of the struct. May be a general path expression, must not be an empty string. Has the same namespace prefix usage as <code>propName</code> in <code>GetProperty</code> .
fieldNS	The namespace URI for the field. Has the same URI and prefix usage as the <code>schemaNS</code> parameter.
fieldName	The name of the field. Must be a single XML name, must not be null or the empty string. Has the same namespace prefix usage as the <code>structName</code> parameter.
fieldValue	An UTF-8 string that is the value of the field, if the field has a value. Has the same usage as <code>propValue</code> in <code>GetProperty</code> .
options	Option flags describing the field.

Sort

Plugin Version: 9.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Sorts the data model tree of an XMP object.

Notes:

Use this function to sort the data model of an XMP object into a canonical order. This can be convenient when comparing data models, (e.g. by text comparison of `DumpObject` output).

At the top level the namespaces are sorted by their prefixes. Within a namespace, the top level properties are sorted by name. Within a struct, the fields are sorted by their qualified name, i.e. their XML prefix:local form. Unordered arrays of simple items are sorted by value. Language Alternative arrays are sorted by the `xml:lang` qualifiers, with the "x-default" item placed first.

2.10.2 Constants

kAllowCommas = & h10000000

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for the property accessor functions.

Notes: Used by `CatenateArrayItems` and `SeparateArrayItems`.

kArrayLastItem = -1

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The index constant for the last item.

kDeleteEmptyValues = 4

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for `RemoveProperties()` and `AppendProperties()`.

Notes: Delete properties if the new value is empty.

kDeleteExisting = & h20000000

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants.

Notes: Used by `SetXyz` functions to delete any pre-existing property.

kDoAllProperties = 1

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for `RemoveProperties()` and `AppendProperties()`.

Notes: Do all properties, default is just external properties.

kEncodeUTF16Big = 2

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for `SerializeToBuffer()`.

Notes: Use UTF16 big-endian encoding

kEncodeUTF16Little = 3

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for SerializeToBuffer().

Notes: Use UTF16 little-endian encoding

kEncodeUTF32Big = 4

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for SerializeToBuffer().

Notes: Use UTF32 big-endian encoding

kEncodeUTF32Little = 5

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for SerializeToBuffer().

Notes: Use UTF32 little-endian encoding

kEncodeUTF8 = 0

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for SerializeToBuffer().

Notes: Use UTF8 encoding

kEncodingMask = & h7

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for SerializeToBuffer().

Notes: Bit-flag mask for encoding-type bits

kExactPacketLength = & h200

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for `SerializeToBuffer()`.

Notes: The padding parameter is the overall packet length.

kImplReservedMask = & h70000000

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants.

kIncludeAliases = & h800

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for `RemoveProperties()` and `AppendProperties()`.

Notes: Include aliases, default is just actual properties.

kIncludeThumbnailPad = & h100

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for `SerializeToBuffer()`.

Notes: Include a padding allowance for a thumbnail image.

kInsertAfterItem = & h8000

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants for the option for array item location.

Notes: Insert a new item after the given index.

kInsertBeforeItem = & h4000

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants for the option for array item location.

Notes: Insert a new item before the given index.

kIterAliases = 1

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants for the option bit flags for XMPIteratorMBS construction.

Notes: Iterate the global alias table.

kIterClassMask = & hFF

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants for the option bit flags for XMPIteratorMBS construction.

Notes: The low 8 bits are an enum of what data structure to iterate.

kIterIncludeAliases = & h800

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants for the option bit flags for XMPIteratorMBS construction.

Notes: Include aliases, default is just actual properties.

kIterJustChildren = & h100

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants for the option bit flags for XMPIteratorMBS construction.

Notes: Just do the immediate children of the root, default is subtree.

kIterJustLeafName = & h400

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants for the option bit flags for XMPIteratorMBS construction.

Notes: Return just the leaf part of the path, default is the full path.

kIterJustLeafNodes = & h200

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants for the option bit flags for XMPIteratorMBS construction.

Notes: Just do the leaf nodes, default is all nodes in the subtree.

kIterNamespaces = 2

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants for the option bit flags for XMPIteratorMBS construction.

Notes: Iterate the global namespace table.

kIterOmitQualifiers = & h1000

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants for the option bit flags for XMPIteratorMBS construction.

Notes: Omit all qualifiers.

kIterProperties = 0

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants for the option bit flags for XMPIteratorMBS construction.

Notes: Iterate the property tree of a TXMPMeta object.

kIterSkipSiblings = 2

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants for the option bit flags for Skip.

Notes: Skip the subtree below and remaining siblings of the current node.

kIterSkipSubtree = 1

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants for the option bit flags for Skip.

Notes: Skip the subtree below the current node.

kLittleEndianBit = 1

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for SerializeToBuffer().

kNoOptions = 0

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The constant to specify to use no options.

kNS_AdobeStockPhoto = "http://ns.adobe.com/StockPhoto/1.0/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

kNS_ASF = "http://ns.adobe.com/asf/1.0/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

kNS_CameraRaw = "http://ns.adobe.com/camera-raw-settings/1.0/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

kNS_CreatorAtom = "http://ns.adobe.com/creatorAtom/1.0/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

kNS_DC = "http://purl.org/dc/elements/1.1/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants from outside Adobe.

Notes: The XML namespace for the Dublin Core schema.

kNS_DICOM = "http://ns.adobe.com/DICOM/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants from outside Adobe.

kNS_DM = "http://ns.adobe.com/xmp/1.0/DynamicMedia/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

kNS_EXIF = "http://ns.adobe.com/exif/1.0/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

Notes: The XML namespace for Adobe's EXIF schema.

kNS_EXIF_Aux = "http://ns.adobe.com/exif/1.0/aux/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

kNS_IPTCCore = "http://iptc.org/std/Iptc4xmpCore/1.0/xmlns/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants from outside Adobe.

Notes: The XML namespace for the IPTC Core schema.

kNS_JP2K = "http://ns.adobe.com/jp2k/1.0/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

kNS_JPEG = "http://ns.adobe.com/jpeg/1.0/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

kNS_PDF = "http://ns.adobe.com/pdf/1.3/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

Notes: The XML namespace for the PDF schema.

kNS_PDFA_Extension = "http://www.aiim.org/pdfa/ns/extension/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants from outside Adobe.

kNS_PDFA_Field = "http://www.aiim.org/pdfa/ns/field# "

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants from outside Adobe.

kNS_PDF_A_ID = "http://www.aiim.org/pdfa/ns/id/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants from outside Adobe.

kNS_PDF_A_Property = "http://www.aiim.org/pdfa/ns/property#" "

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants from outside Adobe.

kNS_PDF_A_Schema = "http://www.aiim.org/pdfa/ns/schema#" "

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants from outside Adobe.

kNS_PDF_A_Type = "http://www.aiim.org/pdfa/ns/type#" "

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants from outside Adobe.

kNS_PDF_X = "http://ns.adobe.com/pdfx/1.3/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants from outside Adobe.

kNS_PDF_X_ID = "http://www.npes.org/pdfx/ns/id/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants from outside Adobe.

kNS_Photoshop = "http://ns.adobe.com/photoshop/1.0/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

Notes: The XML namespace for the Photoshop custom schema.

kNS_PNG = "http://ns.adobe.com/png/1.0/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

kNS_PSAAlbum = "http://ns.adobe.com/album/1.0/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

kNS_RDF = "http://www.w3.org/1999/02/22-rdf-syntax-ns#" "

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants from outside Adobe.

Notes: The XML namespace for RDF.

kNS_SWF = "http://ns.adobe.com/swf/1.0/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

kNS_TIFF = "http://ns.adobe.com/tiff/1.0/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

Notes: The XML namespace for Adobe's TIFF schema.

kNS_WAV = "http://ns.adobe.com/xmp/wav/1.0/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

kNS_XML = "http://www.w3.org/XML/1998/namespace"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants from outside Adobe.

Notes: The XML namespace for XML.

kNS_XMP = "http://ns.adobe.com/xap/1.0/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

Notes: The XML namespace for the XMP "basic" schema.

kNS_XMP_BJ = "http://ns.adobe.com/xap/1.0/bj/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

Notes: The XML namespace for the job management schema.

kNS_XMP_Dimensions = "http://ns.adobe.com/xap/1.0/sType/Dimensions#" "

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for qualifiers and structured property fields.

Notes: The XML namespace for fields of the Dimensions type.

kNS_XMP_Font = "http://ns.adobe.com/xap/1.0/sType/Font#" "

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for qualifiers and structured property fields.

kNS_XMP_Graphics = "http://ns.adobe.com/xap/1.0/g/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for qualifiers and structured property fields.

kNS_XMP_G_IMG = "http://ns.adobe.com/xap/1.0/g/img/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

Notes: Deprecated XML namespace constant.

kNS_XMP_IdentifierQual = "http://ns.adobe.com/xmp/Identifier/qual/1.0/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for qualifiers and structured property fields.

Notes: The XML namespace for qualifiers of the xmp:Identifier property.

kNS_XMP_Image = "http://ns.adobe.com/xap/1.0/g/img/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for qualifiers and structured property fields.

Notes: The XML namespace for fields of a graphical image. Used for the Thumbnail type.

kNS_XMP_ManifestItem = "http://ns.adobe.com/xap/1.0/sType/ManifestItem#" "

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for qualifiers and structured property fields.

kNS_XMP_MM = "http://ns.adobe.com/xap/1.0/mm/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

Notes: The XML namespace for the XMP digital asset management schema.

kNS_XMP_Note = "http://ns.adobe.com/xmp/note/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

kNS_XMP_PagedFile = "http://ns.adobe.com/xap/1.0/t/pg/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for qualifiers and structured property fields.

kNS_XMP_ResourceEvent = "http://ns.adobe.com/xap/1.0/sType/ResourceEvent#" "

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for qualifiers and structured property fields.

Notes: The XML namespace for fields of the ResourceEvent type.

kNS_XMP_ResourceRef = "http://ns.adobe.com/xap/1.0/sType/ResourceRef#" "

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for qualifiers and structured property fields.

Notes: The XML namespace for fields of the ResourceRef type.

kNS_XMP_Rights = "http://ns.adobe.com/xap/1.0/rights/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

Notes: The XML namespace for the XMP copyright schema.

kNS_XMP_ST_Job = "http://ns.adobe.com/xap/1.0/sType/Job#" "

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for qualifiers and structured property fields.

Notes: The XML namespace for fields of the JobRef type.

kNS_XMP_ST_Version = "http://ns.adobe.com/xap/1.0/sType/Version#" "

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for qualifiers and structured property fields.

Notes: The XML namespace for fields of the Version type.

kNS_XMP_T = "http://ns.adobe.com/xap/1.0/t/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

Notes:

The XML namespace for the XMP text document schema.

Deprecated XML namespace constant.

kNS_XMP_Text = "http://ns.adobe.com/xap/1.0/t/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for qualifiers and structured property fields.

kNS_XMP_T_PG = "http://ns.adobe.com/xap/1.0/t/pg/"

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

Notes:

The XML namespace for the XMP paged document schema.
Deprecated XML namespace constant.

kOmitAllFormatting = & h800

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for `SerializeToBuffer()`.

Notes: Omit all formatting whitespace.

kOmitPacketWrapper = & h10

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for `SerializeToBuffer()`.

Notes: Omit the XML packet wrapper.

kOmitXMPMetaElement = & h1000

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for `SerializeToBuffer()`.

Notes: Omit the `x:xmpmeta` element surrounding the `rdf:RDF` element.

kParseMoreBuffers = 2

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** This is the not last input buffer for this parse stream.

kPropArrayFormMask = & h1E00

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for the property accessor functions.

Notes: Property type bit-flag mask for all array types

kPropArrayIsAlternate = & h800

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for the property accessor functions.

Notes: Implies kPropArrayIsOrdered, items are alternates. It is serialized using an rdf:Alt container.

kPropArrayIsAltText = & h1000

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for the property accessor functions.

Notes: Implies kPropArrayIsAlternate, items are localized text. Each array element is a simple property with an xml:lang attribute.

kPropArrayIsOrdered = & h400

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for the property accessor functions.

Notes: Implies kPropValueIsArray, item order matters. It is serialized using an rdf:Seq container.

kPropArrayIsUnordered = & h200

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for the property accessor functions.

Notes: The item order does not matter.

kPropArrayLocationMask = & hC000

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants for the option bit flags for the property setting functions.

Notes: Bit-flag mask for array-item location bits

kPropCompositeMask = & h1F00

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for the property accessor functions.

Notes: Property type bit-flag mask for composite types (array and struct)

kPropHasAliases = & h20000

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for the property accessor functions.

Notes: This property is the base value (actual) for a set of aliases. This is only returned by `GetProperty()` and then only if the property name is simple, not an path expression.

kPropHasLang = & h40

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for the property accessor functions.

Notes: Implies `kPropHasQualifiers`, property has `xml:lang`.

kPropHasQualifiers = & h10

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for the property accessor functions.

Notes: The property has qualifiers, includes `rdf:type` and `xml:lang`.

kPropHasType = & h80

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for the property accessor functions.

Notes: Implies kPropHasQualifiers, property has rdf:type.

kPropIsAlias = & h10000

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for the property accessor functions.

Notes: This property is an alias name for another property. This is only returned by GetProperty() and then only if the property name is simple, not a path expression.

kPropIsDerived = & h200000

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for the property accessor functions.

Notes: The value of this property is derived from the document content.

kPropIsInternal = & h40000

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for the property accessor functions.

Notes: The value of this property is "owned" by the application, and should not generally be editable in a UI.

kPropIsQualifier = & h20

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for the property accessor functions.

Notes:

This is a qualifier for some other property, includes rdf:type and xml:lang.

Qualifiers can have arbitrary structure, and can themselves have qualifiers. If the qualifier itself has a structured value, this flag is only set for the top node of the qualifier's subtree.

kPropIsStable = & h100000

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for the property accessor functions.

Notes: The value of this property is not derived from the document content.

kPropValueIsArray = & h200

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for the property accessor functions.

Notes: The value is an array (RDF alt/bag/seq). The "ArrayIs..." flags identify specific types of array; default is a general unordered array, serialized using an rdf:Bag container.

kPropValueIsStruct = & h100

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for the property accessor functions.

Notes: The value is a structure with nested fields.

kPropValueIsURI = 2

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for the property accessor functions.

Notes: The XML string form of the property value is a URI, use rdf:resource attribute. DISCOURAGED

kPropValueOptionsMask = 2

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants of the option bit flags for the property setting functions.

Notes: Bit-flag mask for property-value option bits

kReadOnlyPacket = & h20

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for SerializeToBuffer().

Notes: Default is a writeable packet.

kReplaceOldValues = 2

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for RemoveProperties() and AppendProperties().

Notes: Replace existing values, default is to leave them.

kRequireXMPMeta = 1

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Require a surrounding x:xmpmeta element.

kStrictAliasing = 4

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the constants for the option bit flags for ParseFromBuffer().

Notes: Do not reconcile alias differences, throw an exception.

kUseCompactFormat = & h40

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for SerializeToBuffer().

Notes: Use a compact form of RDF.

kUseNullTermination = 0

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The length constants for a string to determinate the length automatically.

kUTF16Bit = 2

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for SerializeToBuffer().

kUTF32Bit = 4

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for SerializeToBuffer().

kWriteAliasComments = & h400

Plugin Version: 9.4 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option bit flags for SerializeToBuffer().

Notes: Show aliases as XML comments.

kXMPFiles_ IgnoreLocalText = 2

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the flags for XMPFiles initialize.

Notes: Ignore non-XMP text that uses an undefined "local" encoding.

kXMPFiles_ ServerMode = 2

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the flags for XMPFiles initialize.

Notes: Combination of flags necessary for server products using XMPFiles.

kXMPTemplate_ AddNewProperties = 8

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option constants for ApplyTemplate function.

Notes: Perform an Add operation, add properties if they don't already exist.

kXMPTemplate_ ClearUnnamedProperties = & h10

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option constants for ApplyTemplate function.

Notes: Perform a Clear operation, keep named properties and delete everything else.

kXMPTemplate_ IncludeInternalProperties = 1

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option constants for ApplyTemplate function.

Notes: Do all properties, default is just external properties.

kXMPTemplate_ ReplaceExistingProperties = 2

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option constants for ApplyTemplate function.

Notes: Perform a Replace operation, add new properties and modify existing ones.

kXMPTemplate_ ReplaceWithDeleteEmpty = 4

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option constants for ApplyTemplate function.

Notes: Similar to Replace, also delete if the template has an empty value.

kXMP_ NS_ BWF = "http://ns.adobe.com/bwf/bext/1.0/"

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

kXMP_ NS_ Script = "http://ns.adobe.com/xmp/1.0/Script/"

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the XML namespace constants for standard XMP schema.

kXMP_ WriteAliasComments = & h400

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** One of the option flags for SerializeToBuffer.

Notes: Show aliases as XML comments.

Version = "5.1.2"

Plugin Version: 11.0 Console & Web: No Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The version string of the XMP library.

2.11 class XMPIteratorMBS

class XMPIteratorMBS

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** The XMPIteratorMBS class provides a uniform means to iterate over several XMP data structures, including the schema and properties within an XMP object plus global tables such as registered namespaces.

Notes:

Note: Only XMP object iteration is implemented at this time. There are no table iterators yet.

Iteration over the schema and properties within an XMP object is the most important and complex use of XMPIteratorMBS. It is helpful to have a thorough understanding of the XMP data tree. One way to learn this is to create some complex XMP and examine the output of XMPMetaMBS.DumpObject. This is also described in the XMP Specification, in the XMP Data Model chapter.

The top of the XMP data tree is a single root node. This does not explicitly appear in the dump and is never visited by an iterator (that is, it is never returned from XMPIteratorMBS.Next). Beneath the root are schema nodes. These are just collectors for top level properties in the same namespace. They are created and destroyed implicitly. Beneath the schema nodes are the property nodes. The nodes below a property node depend on its type (simple, struct, or array) and whether it has qualifiers.

A XMPIteratorMBS constructor defines a starting point for the iteration and options that control how it proceeds. By default the iteration starts at the root and visits all nodes beneath it in a depth first manner. The root node is not visited, the first visited node is a schema node. You can provide a schema name or property path to select a different starting node. By default this visits the named root node first then all nodes beneath it in a depth first manner.

The XMPIteratorMBS.Next method delivers the schema URI, path, and option flags for the node being visited. If the node is simple it also delivers the value. Qualifiers for this node are visited next. The fields

of a struct or items of an array are visited after the qualifiers of the parent.

The options to control the iteration are:

<code>kXMP_ IterJustChildren</code>	Visit just the immediate children of the root. Skip the root itself and all nodes below the immediate children. This omits the qualifiers of the immediate children, the qualifier nodes being below what they qualify.
<code>kXMP_ IterJustLeafNodes</code>	Visit just the leaf property nodes and their qualifiers.
<code>kXMP_ IterJustLeafName</code>	Return just the leaf component of the node names. The default is to return the full path name.
<code>kXMP_ IterIncludeAliases</code>	Include aliases as part of the iteration. Since aliases are not actual nodes the default iteration does not visit them.
<code>kXMP_ IterOmitQualifiers</code>	Do not visit the qualifiers of a node.

2.11.1 Methods

Constructor

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Empty constructor which does nothing.

See also:

- 2.11.1 `Constructor(meta as XMPMetaMBS, options as integer=0)` 120
- 2.11.1 `Constructor(meta as XMPMetaMBS, schemaNS as string, options as integer=0)` 121
- 2.11.1 `Constructor(meta as XMPMetaMBS, schemaNS as string, propName as string, options as integer=0)` 122
- 2.11.1 `Constructor(schemaNS as string, propName as string, options as integer)` 122

Constructor(meta as XMPMetaMBS, options as integer=0)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Construct an iterator for the global tables of the XMP toolkit.

Notes:

options: Option flags to control the iteration.

The available option flags are:

kXMP_ IterJustChildren	= & h100	Just visit the immediate children of the root, default is subtree.
kXMP_ IterJustLeafNodes	= & h200	Just visit the leaf nodes, default visits all nodes.
kXMP_ IterJustLeafName	= & h400	Return just the leaf part of the path, default is the full path.
kXMP_ IterOmitQualifiers	= & h1000	Omit all qualifiers.

Note: Not yet implemented in the XMP SDK.

See also:

- 2.11.1 Constructor 120
- 2.11.1 Constructor(meta as XMPMetaMBS, schemaNS as string, options as integer=0) 121
- 2.11.1 Constructor(meta as XMPMetaMBS, schemaNS as string, propName as string, options as integer=0) 122
- 2.11.1 Constructor(schemaNS as string, propName as string, options as integer) 122

Constructor(meta as XMPMetaMBS, schemaNS as string, options as integer=0)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Construct an iterator for the global tables of the XMP toolkit.

Notes:

schemaNS: Optional schema namespace URI to restrict the iteration. Omitted (visit all schema) by passing "".

options: Option flags to control the iteration.

The available option flags are:

kXMP_ IterJustChildren	= & h100	Just visit the immediate children of the root, default is subtree.
kXMP_ IterJustLeafNodes	= & h200	Just visit the leaf nodes, default visits all nodes.
kXMP_ IterJustLeafName	= & h400	Return just the leaf part of the path, default is the full path.
kXMP_ IterOmitQualifiers	= & h1000	Omit all qualifiers.

Note: Not yet implemented in the XMP SDK.

See also:

- 2.11.1 Constructor 120
- 2.11.1 Constructor(meta as XMPMetaMBS, options as integer=0) 120
- 2.11.1 Constructor(meta as XMPMetaMBS, schemaNS as string, propName as string, options as integer=0) 122
- 2.11.1 Constructor(schemaNS as string, propName as string, options as integer) 122

Constructor(meta as XMPMetaMBS, schemaNS as string, propName as string, options as integer=0)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Construct an iterator for the global tables of the XMP toolkit.

Notes:

schemaNS: Optional schema namespace URI to restrict the iteration. Omitted (visit all schema) by passing "".

propName: Optional property name to restrict the iteration. May be an arbitrary path expression. Omitted (visit all properties) by passing "". If not empty a schema URI must also be provided.

options: Option flags to control the iteration.

The available option flags are:

kXMP_ IterJustChildren	= & h100	Just visit the immediate children of the root, default is subtree.
kXMP_ IterJustLeafNodes	= & h200	Just visit the leaf nodes, default visits all nodes.
kXMP_ IterJustLeafName	= & h400	Return just the leaf part of the path, default is the full path.
kXMP_ IterOmitQualifiers	= & h1000	Omit all qualifiers.

Note: Not yet implemented in the XMP SDK.

See also:

- 2.11.1 Constructor 120
- 2.11.1 Constructor(meta as XMPMetaMBS, options as integer=0) 120
- 2.11.1 Constructor(meta as XMPMetaMBS, schemaNS as string, options as integer=0) 121
- 2.11.1 Constructor(schemaNS as string, propName as string, options as integer) 122

Constructor(schemaNS as string, propName as string, options as integer)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Construct an iterator for the properties within an XMP object.

Notes:

On success handle is not 0.

schemaNS: Optional schema namespace URI to restrict the iteration. Omitted (visit all schema) by passing "".

propName: Optional property name to restrict the iteration. May be an arbitrary path expression. Omitted

(visit all properties) by passing "". If not empty a schema URI must also be provided.
options: Option flags to control the iteration.

The available option flags are:

kXMP_ IterJustChildren	= & h100	Just visit the immediate children of the root, default is subtree.
kXMP_ IterJustLeafNodes	= & h200	Just visit the leaf nodes, default visits all nodes.
kXMP_ IterJustLeafName	= & h400	Return just the leaf part of the path, default is the full path.
kXMP_ IterOmitQualifiers	= & h1000	Omit all qualifiers.

See also:

- 2.11.1 Constructor 120
- 2.11.1 Constructor(meta as XMPMetaMBS, options as integer=0) 120
- 2.11.1 Constructor(meta as XMPMetaMBS, schemaNS as string, options as integer=0) 121
- 2.11.1 Constructor(meta as XMPMetaMBS, schemaNS as string, propName as string, options as integer=0) 122

NextItem() as boolean

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Visit the next node in the iteration.

See also:

- 2.11.1 NextItem(byref schemaNS as string) as boolean 123
- 2.11.1 NextItem(byref schemaNS as string, byref propPath as string) as boolean 124
- 2.11.1 NextItem(byref schemaNS as string, byref propPath as string, byref propValue as string) as boolean 125
- 2.11.1 NextItem(byref schemaNS as string, byref propPath as string, byref propValue as string, byref options as integer) as boolean 125

NextItem(byref schemaNS as string) as boolean

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Visit the next node in the iteration.

Notes:

Parameters:

schemaNS	A string that is assigned the schema namespace URI of the current property.
propPath	A string that is assigned the XPath name of the current property.
propValue	A string that is assigned the value of the current property.
options	An integer that is assigned the flags describing the current property.

See also:

- 2.11.1 NextItem() as boolean 123
- 2.11.1 NextItem(byref schemaNS as string, byref propPath as string) as boolean 124
- 2.11.1 NextItem(byref schemaNS as string, byref propPath as string, byref propValue as string) as boolean 125
- 2.11.1 NextItem(byref schemaNS as string, byref propPath as string, byref propValue as string, byref options as integer) as boolean 125

NextItem(byref schemaNS as string, byref propPath as string) as boolean

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Visit the next node in the iteration.

Notes:

Parameters:

schemaNS	A string that is assigned the schema namespace URI of the current property.
propPath	A string that is assigned the XPath name of the current property.

See also:

- 2.11.1 NextItem() as boolean 123
- 2.11.1 NextItem(byref schemaNS as string) as boolean 123
- 2.11.1 NextItem(byref schemaNS as string, byref propPath as string, byref propValue as string) as boolean 125
- 2.11.1 NextItem(byref schemaNS as string, byref propPath as string, byref propValue as string, byref options as integer) as boolean 125

NextItem(byref schemaNS as string, byref propPath as string, byref propValue as string) as boolean

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Visit the next node in the iteration.

Notes:

Parameters:

schemaNS	A string that is assigned the schema namespace URI of the current property.
propPath	A string that is assigned the XPath name of the current property.
propValue	A string that is assigned the value of the current property.

See also:

- 2.11.1 NextItem() as boolean 123
- 2.11.1 NextItem(byref schemaNS as string) as boolean 123
- 2.11.1 NextItem(byref schemaNS as string, byref propPath as string) as boolean 124
- 2.11.1 NextItem(byref schemaNS as string, byref propPath as string, byref propValue as string, byref options as integer) as boolean 125

NextItem(byref schemaNS as string, byref propPath as string, byref propValue as string, byref options as integer) as boolean

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Visit the next node in the iteration.

Notes:

Parameters:

schemaNS	A string that is assigned the schema namespace URI of the current property.
propPath	A string that is assigned the XPath name of the current property.
propValue	A string that is assigned the value of the current property.
options	An integer that is assigned the flags describing the current property.

See also:

- 2.11.1 NextItem() as boolean 123

- 2.11.1 NextItem(byref schemaNS as string) as boolean 123
- 2.11.1 NextItem(byref schemaNS as string, byref propPath as string) as boolean 124
- 2.11.1 NextItem(byref schemaNS as string, byref propPath as string, byref propValue as string) as boolean 125

Skip(options as integer)

Plugin Version: 6.4 Console & Web: Yes Mac: Yes, Win: Yes, Linux: Yes, . **Function:** Skip some portion of the remaining iterations.

Notes:

The available option flags are:

- kXMP_ IterSkipSubtree = 1 Skip the subtree below the current node.
- kXMP_ IterSkipSiblings = 2 Skip the subtree below and remaining siblings of the current node.

Chapter 3

List of all classes

• XMPAssertNotifyMBS	31
• XMPDateTimeMBS	24
• XMPExceptionMBS	32
• XMPFilesMBS	32
• XMPIteratorMBS	119
• XMPMetaMBS	57
• XMPPacketInfoMBS	15
• XMPScannerMBS	16
• XMPSnippetMBS	22
• XMPTextOutputMBS	21
• XMPVersionInfoMBS	19