

Changes to section Introduction

ISO/IEC 29500 specifies a family of XML schemas, collectively called *Office Open XML*, which define the XML vocabularies for word-processing, spreadsheet, and presentation documents, as well as the packaging of documents that conform to these schemas.

The goal is to enable the implementation of the Office Open XML formats by the widest set of tools and platforms, fostering interoperability across office productivity applications and line-of-business systems, as well as to support and strengthen document archival and preservation, all in a way that is fully compatible with the existing corpus of ~~Microsoft~~Microsoft[®] Office documents.

The intent of this Part of ISO/IEC 29500 is to enable a transitional period during which existing binary documents being migrated to ISO/IEC 29500 can make use of legacy features to preserve their fidelity, while noting that new documents should not use them. Part 1, §2.4, “Document Conformance”, notes that WML Strict, SML Strict and PML Strict documents do not use any of the features defined in Part 4.

This Part of ISO/IEC 29500 is normative for the current edition of ISO/IEC 29500, but is not guaranteed to be included in future revisions of that Standard. The intent is to enable the group responsible for maintenance of ISO/IEC 29500 to choose, at a later date, to remove this set of features from a revised version of that Standard.

In general, this Part of ISO/IEC 29500 augments Part 1, and inherits the provisions of that Part. Exceptions to this are indicated explicitly.

The following organizations have participated in the creation of ISO/IEC 29500 and their contributions are gratefully acknowledged:

Apple, Barclays Capital, BP, The British Library, Essilor, Intel, ~~Microsoft~~Microsoft[®], NextPage, Novell, Statoil, Toshiba, and the United States Library of Congress.

Changes to section 1. Scope

ISO/IEC 29500 defines a set of XML vocabularies for representing word-processing documents, spreadsheets and presentations. On the one hand, the goal of ISO/IEC 29500 is to represent faithfully the existing corpus of word-processing documents, spreadsheets and presentations that have been produced by ~~Microsoft~~Microsoft® Office applications (from ~~Microsoft~~Microsoft® Office 97 to ~~Microsoft~~Microsoft® Office 2008, inclusive). It also specifies requirements for Office Open XML consumers and producers. On the other hand, the goal is to facilitate extensibility and interoperability by enabling implementations by multiple vendors and on multiple platforms.

This Part of ISO/IEC 29500 defines features for backward-compatibility and that are useful for high-quality migration of existing binary documents to ISO/IEC 29500. These features are used only by documents of conformance class WML Transitional (**\$Error! Reference source not found.**), SML Transitional (**\$Error! Reference source not found.**), or PML Transitional (**\$Error! Reference source not found.**). These features are sometimes needed for high-quality migration of existing binary documents to ISO/IEC 29500.

Changes to section 14.7.3.52 usePrinterMetrics (Use Printer Metrics To Display Documents)

This element specifies whether applications shall use the printer metrics of the currently active printer when determining how to display the contents of a WordprocessingML document. *Printer metrics* are printer-specific settings which can be queried to tell an application how and where text shall be displayed on a printed page.

Typically, applications display the content of a document in a device independent manner - the application is therefore not changing the layout of a document based on the currently attached printer, and instead shall dictate to the printer where characters shall be presented on the page when printed. This element, when present with a val attribute value of true (or equivalent), specifies that the metrics of the current printer shall be used to display the document instead.

Specifically, when this setting is enabled, the printer metrics are used to determine the number of pixels per logical inch along the screen width and height. This should then be used to compute the pixel height of the fonts requested when displaying the document, as well as to scale between any logical units within the document (e.g. drawing object sizes) to the appropriate device units. Those units would then need to be scaled back into screen units for final display to a screen, but not scaled again when displayed to a printer.

[Note: On the ~~Windows~~Windows® platform, you can use the GetDeviceCaps function to retrieve device-specific information for the specified printer. For this specific setting, you can use GetDeviceCaps(hdc, LOGPIXELSX) and GetDeviceCaps(hdc, LOGPIXELSY) with a printer DC to retrieve the number of pixels per logical inch along the screen width and height. With this, you can then use those DPI metrics to compute a pixel value for the font request in the LOGFONT structure (the LOGFONT structure defines the attributes of a font). A common formula to do this is $S_{px} = S_{pts} * \frac{LOGPIXELSY}{72}$. end note]

[*Example*: Consider a WordprocessingML document. The default must use device-independent layout to present the contents of the page.

However, if this compatibility setting is turned on:

```
<w:compat>
  <w:usePrinterMetrics />
</w:compat>
```

Then the printer metrics of the current active printer must be used to determine the display of the contents of the document instead, as needed. *end example*]

This element’s content model is defined by the common boolean property definition in Part 1, §17.17.4.

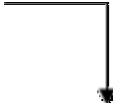
Changes to section 15.9.16 Changed attribute for pivotSelection element (Part 1, §18.3.1.69)

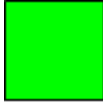


Attributes	Description
id (Relationship Id) Namespace: .../officeDocument /2006/relationships	Relationship Id pointing to the particular PivotTable PivotTable™ Part corresponding to this selection. The possible values for this attribute are defined by the ST_RelationshipId simple type (Part 1, §22.8.2.1).

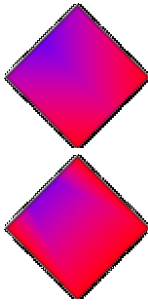
Changes to section 19.1.2.14 path (Shape Path)

This element defines the path that makes up the shape. This is done through a string that contains a rich set of pen movement commands. This element also describes the limo-stretch point, inscribed textbox rectangle locations and connection site locations. The limo-stretch definition and the formulas element (§**Error! Reference source not found.**) allow greater designer control of how the path scales. [*Example*: They allow, for example, definition of a true rounded corner rectangle where the corners remain circular even though the rectangle is scaled anisotropically. *end example*]


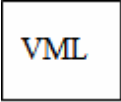
Attributes	Description
arrowok (Arrowhead Display Toggle)	Specifies whether arrowheads are allowed to be displayed. This attribute overrides all other arrowhead attributes in the parent or the stroke element (§ Error! Reference source not found.). Default is false. [<i>Example</i> : <pre><v:shape style="width:50;height:50"> <v:stroke endarrow="block"/> <v:path arrowok="true" v="m 0,0 l 1000,0 1000,1000 e"/> </v:shape></pre>


Attributes	Description
	 <p><i>end example]</i></p> <p>The possible values for this attribute are defined by the ST_TrueFalse simple type (\$Error! Reference source not found.).</p>
<p>connectangles (Connection Point Connect Angles)</p> <p>Namespace: urn:schemas-microsoft-com:office:office</p>	<p>Specifies the angle at which curves connect to a shape's connection points. The connection angles are defined by a string consisting of angle values delimited by commas. Default is no value.</p> <p>[<i>Example:</i> Connections are made along the horizontal and vertical axes:</p> <pre><v:path ... o:connectangles="0,90,180,270" ... > </v:path></pre> <p><i>end example]</i></p> <p>The possible values for this attribute are defined by the W3C XML Schema string datatype.</p>
<p>connectlocs (Connection Points)</p> <p>Namespace: urn:schemas-microsoft-com:office:office</p>	<p>Specifies the location of connection points on a path. The connection points are defined by a string consisting of pairs of x and y values, delimited by commas. This is used if connecttype is custom. Default is no value.</p> <p>[<i>Example:</i> Connection points exist at the midpoints of the sides of the square:</p> <pre><v:path ... v="m 0,0 l 100,0 100,100 0,100 x e" o:connectlocs="50,0;100,50;50,100;0,50" ... > </v:path></pre> <p><i>end example]</i></p> <p>The possible values for this attribute are defined by the W3C XML Schema string datatype.</p>
<p>connecttype (Connection Point Type)</p> <p>Namespace: urn:schemas-microsoft-com:office:office</p>	<p>Specifies the kind of connection points used for attaching shapes to other shapes. Default is none. If set to custom, connectlocs is used. Allowed values are:</p> <p>[<i>Example:</i></p> <pre><v:path ... o:connecttype="custom" o:connectlocs="50,0;100,50;50,100;0,50" ... > </v:path></pre> <p><i>end example]</i></p>

Attributes	Description
<p>extrusionok (Extrusion Toggle)</p> <p>Namespace: urn:schemas- microsoft- com:office:office</p>	<p>The possible values for this attribute are defined by the ST_ConnectType simple type (\$Error! Reference source not found.).</p> <p>Specifies whether an extrusion is allowed to be displayed. This attribute overrides all other extrusion attributes in the parent or the extrusion element (\$Error! Reference source not found.). Default is true.</p> <p>[Example:</p> <pre><v:rect fillcolor="lime" style="width:50;height:50"> <v:extrusion on="true"/> <v:path o:extrusionok="false"/> </v:rect></pre>  <p><v:path o:extrusionok="false"/></p>  <p><v:path o:extrusionok="true"/></p> <p>end example]</p> <p>The possible values for this attribute are defined by the ST_TrueFalse simple type (\$Error! Reference source not found.).</p>
<p>fillok (Shape Fill Toggle)</p>	<p>Specifies whether a fill is allowed to be displayed. This attribute overrides all other fill attributes in the parent or fill element (\$Error! Reference source not found.). Default is true.</p> <p>[Example:</p> <pre><v:shape style="width:50;height:50" fillcolor="red"> <v:path filllok="false" v="m 0,0 l 0,1000, 1000,1000, 1000,0 x e"/> </v:shape></pre>  <p>end example]</p> <p>The possible values for this attribute are defined by the ST_TrueFalse simple type</p>

Attributes	Description
gradientshapeok (Gradient Shape Toggle)	<p>(Error! Reference source not found.)</p> <p>Specifies whether a gradient path is made up of repeated concentric paths. Default is false.</p> <p>If true, a gradient fill can be produced by repeated drawing of scaled versions of the path - this shall only be set if it is possible to scale the path in such a way that a fill is always contained in the original path. This controls the interpretation of the type="gradientradial" attribute of the fill element (Error! Reference source not found.).</p> <p>[<i>Example:</i> In the first case, the radial gradient is aligned irrespective of the shape's path:</p> <pre data-bbox="451 653 1208 852"> <v:shape style="width:50;height:50;rotation:45" path="m 0,0 l 0,1000, 1000,1000, 1000,0 x e"> <v:path gradientshapeok="false"/> <v:fill type="gradientradial" color="red" color2="blue"/> </v:shape> </pre> <div data-bbox="451 888 980 1192">  <p style="text-align: center;">gradientshapeok="false"</p> <p style="text-align: center;">gradientshapeok="true"</p> </div> <p><i>end example]</i></p> <p>The possible values for this attribute are defined by the ST_TrueFalse simple type (Error! Reference source not found.).</p>
id (Unique Identifier)	<p>Specifies a unique identifier that can be used to reference a VML object.</p> <p>Default is no value.</p> <p>[<i>Example:</i></p> <pre data-bbox="451 1598 886 1661"> <v:shape ... id="myShape" ... > </v:shape> </pre> <p><i>end example]</i></p> <p>The possible values for this attribute are defined by the W3C XML Schema string datatype.</p>
insetpenok (Inset	<p>Specifies whether the stroke can be inset from the path. If this is false, it overrides the</p>

Attributes	Description
Stroke From Path Flag)	<p>insetpen attribute and prevents the stroke from being inset.</p> <p>[Example: The stroke is not inset:</p> <pre data-bbox="451 394 967 491"><v:shape ... insetpen="true"> <v:path ... insetpenok="false"/> </v:shape></pre> <p><i>end example]</i></p> <p>The possible values for this attribute are defined by the ST_TrueFalse simple type (§Error! Reference source not found.).</p>
limo (Limo Stretch Point)	<p>Specifies a stretch point on the shape's edge that defines where and how a shape is allowed to be stretched by a user in a graphical editor. Default is "0,0".</p> <p>[Example:</p> <pre data-bbox="451 863 1110 959"><v:line from="20pt,20pt" to="100pt,20pt"> <v:path limo="60pt,20pt"/> </v:line></pre> <p><i>end example]</i></p> <p>The possible values for this attribute are defined by the W3C XML Schema string datatype.</p>
shadowok (Shadow Toggle)	<p>Specifies whether a shadow is allowed to be displayed. This attribute overrides all other shadow attributes in the parent or the shadow element (§Error! Reference source not found.). Default is true.</p> <p>[Example: The shape has no shadow:</p> <pre data-bbox="451 1367 1269 1535"><v:shape style="width:50;height:50"> <v:path v="m 0,0 l 0,1000, 1000,1000, 1000,0 x e" shadowok="false"/> <v:shadow on="true"/> </v:shape></pre> <p><i>end example]</i></p> <p>The possible values for this attribute are defined by the ST_TrueFalse simple type (§Error! Reference source not found.).</p>
strokeok (Stroke Toggle)	<p>Specifies whether a stroke is displayed. This attribute overrides all other stroke attributes in the parent or the stroke element (§Error! Reference source not found.). Default is true.</p>

Attributes	Description
	<p>[<i>Example</i>: The shape's red stroke is not shown:</p> <pre data-bbox="451 321 1271 485"><v:shape style="width:50;height:50" fillcolor="blue" strokecolor="red"> <v:path v="m 0,0 l 0,1000, 1000,1000, 1000,0 x e" strokeok="false"/> </v:shape></pre>  <p><i>end example</i>]</p> <p>The possible values for this attribute are defined by the ST_TrueFalse simple type (§Error! Reference source not found.).</p>
textboxrect (Text Box Bounding Box)	<p>Specifies one or more text boxes inside a shape. Default is the same as the geometry's bounding box.</p> <p>A textbox is defined by one or more sets of numbers specifying (in order) the left, top, right, and bottom points of the rectangle. Multiple sets are delimited by a semicolon. The default value is the same dimension value as the containing rectangle. If more than one textbox is defined, the comma-delimited quadruple sets that define each textbox are separated by semicolons. Normally textboxes come in sets of 1, 2, 3, or 6 rectangles on a shape. The textboxrect dimensions clip any text that extends beyond its region.</p> <p>[<i>Example</i>: The textbox is 25% down from the top and the exclamation point is clipped:</p> <pre data-bbox="451 1247 1271 1411"><v:shape style="width:60;height:50"> <v:path v="m 0,0 l 0,1000, 1000,1000, 1000,0 x e" textboxrect="0,250,850,1000"/> <v:textbox>VML!</v:textbox> </v:shape></pre>  <p><i>end example</i>]</p> <p>The possible values for this attribute are defined by the W3C XML Schema string datatype.</p>
textpathok (Text Path Toggle)	<p>Specifies whether a text path is displayed. Default is false.</p> <p>If true, this indicates that the path is an appropriate warping path for the textpath element (§Error! Reference source not found.). Otherwise, the textpath element shall</p>

Attributes	Description								
	<p>be ignored.</p> <p>[<i>Example</i>: The defined textpath is ignored:</p> <pre data-bbox="451 394 1256 590"> <v:curve from="50,100" to="400,100" control1="200,200" control2="300,200"> <v:path textpathok="false"/> <v:textpath on="false" style="font:normal normal normal 36pt Arial" string="textpath"/> </v:curve> </pre>  <p><i>end example</i>]</p> <p>The possible values for this attribute are defined by the ST_TrueFalse simple type (\$Error! Reference source not found.).</p>								
<p>v (Path Definition)</p>	<p>Specifies a string containing the commands that define the shape's path. This value consists of commands followed by zero or more parameters. Default is no value.</p> <p>The following rules apply to path strings:</p> <ul data-bbox="464 1125 1479 1514" style="list-style-type: none"> • Commas or spaces delimit parameters for each command. Both "m 0,0" and "m0 0" are acceptable. • A parameter that is omitted using commas is treated as having a value of zero. Thus, "c 10,10,0,0,25,13" and "c 10,10,,,25,13" are equivalent. • Parameterized paths are also allowed. In this case, the shape shall also have a formulas element (\$Error! Reference source not found.) with a list of formulas that are substituted into the path using the @ symbol followed by the number of the formula. The adj property of the shape contains the input parameters for these formulas. [<i>Example</i>: For example, "moveto @1@4". <i>end example</i>] The evaluations of the formulas are substituted into the appropriate positions. The @ character also serves as a delimiter. <p>The allowed commands are given below. An asterisk (*) indicates that the command is allowed to be repeated. For the qb command, the controlpoint parameter is also allowed to be repeated.</p> <table border="1" data-bbox="418 1692 1479 1858"> <thead> <tr> <th>Command</th> <th>Name</th> <th>Parameters</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>m</td> <td>moveto</td> <td>2</td> <td>Start a new sub-path at the given (x,y) coordinate.</td> </tr> </tbody> </table>	Command	Name	Parameters	Description	m	moveto	2	Start a new sub-path at the given (x,y) coordinate.
Command	Name	Parameters	Description						
m	moveto	2	Start a new sub-path at the given (x,y) coordinate.						

Attributes		Description	
l	lineto	2*	Draw a line from the current point to the given (x,y) coordinate which becomes the new current point. Specifying a number of coordinate pairs forms a polyline.
c	curveto	6*	Draw a cubic bézier curve from the current point to the coordinate given by the final two parameters. The control points are given by the first four parameters.
x	close	0	Close the current sub-path by drawing a straight line from the current point to the original moveto point.
e	end	0	End the current set of sub-paths. A given set of sub-paths (as delimited by end) is filled. Subsequent sets of sub-paths are filled independently and superimposed on existing ones.
t	rmoveto	2*	Start a new sub-path at a coordinate relative to the current point, cp (cp _x +x, cp _y +y).
r	rlineto	2*	Draw a line from the current point to the given relative coordinate (cp _x +x, cp _y +y).
v	rcurveto	6*	Cubic bézier curve using the given coordinate relative to the current point.
nf	nofill	0	The current set of sub-paths (delimited by e) is not filled.
ns	nostroke	0	The current set of sub-paths (delimited by e) is not stroked.
ae	angleellipseto	6*	Draws a segment of an ellipse as described using these parameters. A straight line is drawn from the current point to the start point of the segment. The parameters are: center (x,y), size(w,h), start angle, end angle.
a1	angleellipse	6*	Same as angleellipseto except that there is an implied moveto the starting point of the segment.

Attributes	Description			
	at	arcto	8*	A segment of the ellipse is drawn which starts at the angle defined by the start radius vector and ends at the angle defined by the end vector. A straight line is drawn from the current point to the start of the arc. The arc is always drawn in a counterclockwise direction. The parameters are: left, top, right, bottom, start(x,y), end(x,y). The first four values define the bounding box of an ellipse. The last four define two radial vectors.
	ar	arc	8*	Same as arcto except there is an implied moveto the start point of the arc.
	wa	clockwisearco	8*	Same as arcto but the arc is drawn in a clockwise direction.
	wr	clockwisearc	8*	Same as arc but the arc is drawn in a clockwise direction
	qx	ellipticalquadrant x	2*	A quarter ellipse is drawn from the current point to the given end point. The elliptical segment is initially tangential to a line parallel to the x-axis. (i.e. the segment starts out horizontal). The parameters are: end(x,y).
	qy	ellipticalquadrant y	2*	Same as ellipticalquadrantx except that the elliptical segment is initially tangential to a line parallel to the y-axis (i.e. the segment starts out vertical).
	qb	quadraticbezier	2+2*	Defines one or more quadratic bézier curves by means of control points and an end point. Intermediate (on-curve) points are obtained by interpolation between successive control points as in the OpenTypeOpenType® font specification. The sub-path need not be started in which case the sub-path is closed. In this case the last point of the sub-path defines the start point of the quadratic bézier. The parameters are: controlpoint(x,y)*, end(x,y).

Attributes	Description
	The possible values for this attribute are defined by the W3C XML Schema string datatype.

[Note: The W3C XML Schema definition of this element's content model (CT_Path) is located in §Error! Reference source not found.. end note]

Changes to section 19.2.2.3 clippath (Shape Clipping Path)

This element specifies the path of the clipping polygon for the shape.

[Example:

```
<v:rect ... wrapcoords="-207 -433 -207 21925 21807 21925 21807 -433 -207 -433"
o:clip="t" o:cliptowrap="t">
  <o:clippath o:v="m-207,-433r,22358121807,21925r,-223581-207,-433xe"/>
</v:rect>
```

end example]

Attributes	Description								
v (Path Definition)	<p>Specifies a string containing the commands that define the shape's path. This value consists of commands followed by zero or more parameters. Default is no value.</p> <p>The following rules apply to path strings:</p> <ul style="list-style-type: none"> Commas or spaces delimit parameters for each command. Both "m 0,0" and "m0 0" are acceptable. A parameter that is omitted using commas is treated as having a value of zero. Thus, "c 10,10,0,0,25,13" and "c 10,10,,,25,13" are equivalent. Parameterized paths are also allowed. In this case, the shape shall also have a formulas element (§Error! Reference source not found.) with a list of formulas that are substituted into the path using the @ symbol followed by the number of the formula. The adj property of the shape contains the input parameters for these formulas. For example, "moveto @1@4". The evaluations of the formulas are substituted into the appropriate positions. @ also serves as a delimiter. <p>The allowed commands are given below. An asterisk (*) indicates that the command is allowed to be repeated. For the qb command, the controlpoint parameter is also allowed to be repeated.</p> <table border="1" data-bbox="415 1745 1485 1879"> <thead> <tr> <th>Command</th> <th>Name</th> <th>Parameters</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>m</td> <td>moveto</td> <td>2</td> <td>Start a new sub-path at the given (x,y) coordinate.</td> </tr> </tbody> </table>	Command	Name	Parameters	Description	m	moveto	2	Start a new sub-path at the given (x,y) coordinate.
Command	Name	Parameters	Description						
m	moveto	2	Start a new sub-path at the given (x,y) coordinate.						

Attributes	Description			
	l	lineto	2*	Draw a line from the current point to the given (x,y) coordinate which becomes the new current point. Specifying a number of coordinate pairs forms a polyline.
	c	curveto	6*	Draw a cubic bézier curve from the current point to the coordinate given by the final two parameters. The control points are given by the first four parameters.
	x	close	0	Close the current sub-path by drawing a straight line from the current point to the original moveto point.
	e	end	0	End the current set of sub-paths. A given set of sub-paths (as delimited by end) is filled. Subsequent sets of sub-paths are filled independently and superimposed on existing ones.
	t	rmoveto	2*	Start a new sub-path at a coordinate relative to the current point, cp (cpx+x, cpy+y).
	r	rlineto	2*	Draw a line from the current point to the given relative coordinate (cpx+x, cpy+y).
	v	rcurveto	6*	Cubic bézier curve using the given coordinate relative to the current point.
	nf	nofill	0	The current set of sub-paths (delimited by e) is not filled.
	ns	nostroke	0	The current set of sub-paths (delimited by e) is not stroked.
	ae	angleellipseto	6*	Draws a segment of an ellipse as described using these parameters. A straight line is drawn from the current point to the start point of the segment. The parameters are: center (x,y), size(w,h), start angle, end angle.
	a1	angleellipse	6*	Same as angleellipseto except that there is an implied moveto the starting point of the segment.

Attributes	Description			
	at	arcto	8*	A segment of the ellipse is drawn which starts at the angle defined by the start radius vector and ends at the angle defined by the end vector. A straight line is drawn from the current point to the start of the arc. The arc is always drawn in a counterclockwise direction. The parameters are: left, top, right, bottom, start(x,y), end(x,y). The first four values define the bounding box of an ellipse. The last four define two radial vectors.
	ar	arc	8*	Same as arcto except there is an implied moveto the start point of the arc.
	wa	clockwisearco	8*	Same as arcto but the arc is drawn in a clockwise direction.
	wr	clockwisearc	8*	Same as arc but the arc is drawn in a clockwise direction
	qx	ellipticalquadrantx	2*	A quarter ellipse is drawn from the current point to the given end point. The elliptical segment is initially tangential to a line parallel to the x-axis. (i.e. the segment starts out horizontal). The parameters are: end(x,y).
	qy	ellipticalquadranty	2*	Same as ellipticalquadrantx except that the elliptical segment is initially tangential to a line parallel to the y-axis (i.e. the segment starts out vertical).
	qb	quadraticbezier	2+2*	Defines one or more quadratic bézier curves by means of control points and an end point. Intermediate (on-curve) points are obtained by interpolation between successive control points as in the OpenTypeOpenType® font specification. The sub-path need not be started in which case the sub-path is closed. In this case the last point of the sub-path defines the start point of the quadratic bézier. The parameters are: controlpoint(x,y)*, end(x,y).

Attributes	Description
	The possible values for this attribute are defined by the W3C XML Schema string datatype.

[Note: The W3C XML Schema definition of this element’s content model (CT_ClipPath) is located in §Error! Reference source not found.. end note]

Changes to section 19.4.2.53 ScriptLanguage (HTML Script Language)

This element specifies the language of the custom function. If the document contains no HTML script, this element should be ignored. Allowed values are:

Value	Description
1	Java
2	Visual Basic <u>Visual Basic®</u>
3	ASP
4	Other

[Example:

```
<x:ClientData> ...
  <x:ScriptLanguage>1</x:ScriptLanguage>
</x:ClientData>
```

end example]

The possible values for this element are defined by the W3C XML Schema nonNegativeInteger datatype.