Overview of DatadiagramML

Microsoft Corporation

March 2004

Applies to:
    Microsoft® Office Visio 2003

Summary: This document describes the elements in the DatadiagramML Schema that are important to document developers and to application developers whose programs will read and write DatadiagramML documents. (8 printed pages)

Contents

A DatadiagramML file is an XML (Extensible Markup Language) document that contains all the components of a Microsoft Office Visio binary file. Much as you can access a Visio document by using Automation or from the ShapeSheet window, you can use XML and supporting tools to access data in a DatadiagramML file.

Note In previously published materials, including the Microsoft Office Visio 2003 SDK, the XML schema that is explained in this document was referred to as *XML for Visio*. From now on, that schema will be known as *DatadiagramML.*

What is XML?

XML (Extensible Markup Language) is the universal format for structured documents and data on the Web.

Just as HTML (Hypertext Markup Language) provides a way to display information on different hardware and software platforms by creating a standard set of fixed, non-customizable tags that a browser interprets and displays, XML provides a way to create documents with custom tags that allows the separation of document content from content presentation.

Documents called *schemas* define valid tags and their containment relationships. The schema is used to verify that the XML document complies with the content model and structure dictated by the schema.

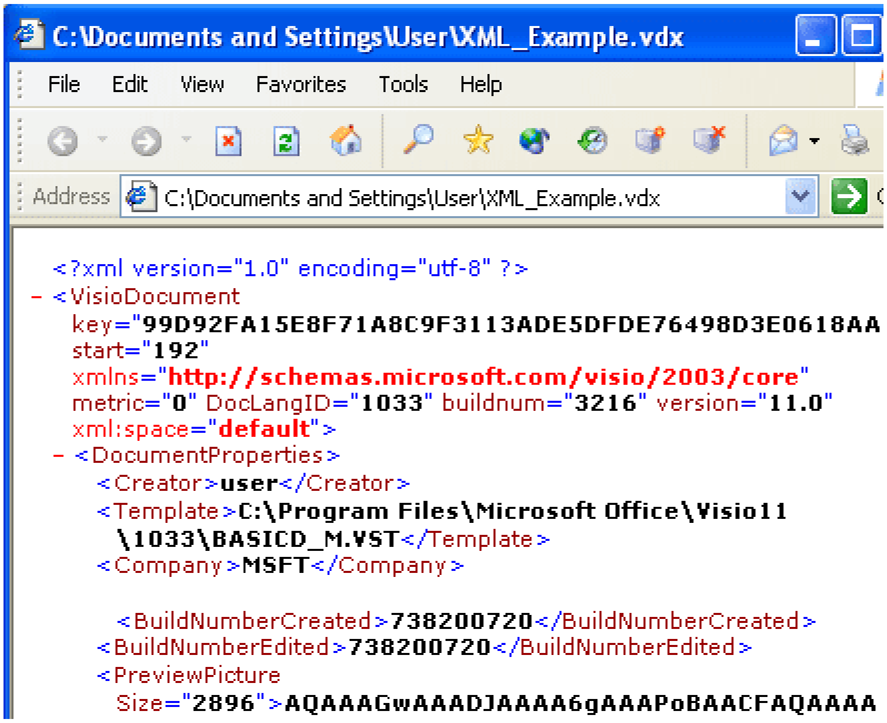
Because XML is simply text, you can use text-processing software to access the data. And because the definition follows standard rules, there are generic parsers that allow access to the internal data. As industry increasingly accepts XML as a standard (supported by W3C, the World Wide Web Consortium, www.w3c.org), XML is increasingly supported by a broad range of software platforms.

Visio and XML

Beginning with Microsoft Visio 2002, support was added for an XML file format that contains the same data as the Visio binary files.

For developers who use the Visio object model or program cells in the ShapeSheet window, the data structures will seem familiar. Most elements that define the data in the DatadiagramML file are named with the same or similar names to the cells in the ShapeSheet or the properties exposed in the object model. Because the document adheres to the XML format standard, you can use standard XSL (Extensible StyleSheet Language) or XSLT (XSL Transformations) languages for retrieving data from XML files.

The following example shows the beginning statements from a DatadiagramML file, opened in Microsoft Internet Explorer.


C:\Documents and Settings\User\XML_Example.vdx
File
Edit
View
Favorites
Tools
Help
Address
C:\Documents and Settings\User\XML_Example.vdx

```
  <?xml version="1.0" encoding="utf-8" ?>
- <VisioDocument
   key="99D92FA15E8F71A8C9F3113ADE5DFDE76498D3E0618AA
   start="192"
   xmlns="http://schemas.microsoft.com/visio/2003/core"
   metric="0" DocLangID="1033" buildnum="3216" version="11.0"
   xml:space="default">
  - <DocumentProperties>
      <Creator>user</Creator>
      <Template>C:\Program Files\Microsoft Office\Visio11
        \1033\BASICD_M.VST</Template>
      <Company>MSFT</Company>

        <BuildNumberCreated>738200720</BuildNumberCreated>
      <BuildNumberEdited>738200720</BuildNumberEdited>
      <PreviewPicture
        Size="2896">AQAAAGwAAADJAAAA6gAAAPoBAACFAQAAAA
```

DatadiagramML statements

The first line in the XML file is a processing instruction that identifies the file as an XML file.

The second line contains the opening tag for the top-level DatadiagramML container, the **VisioDocument** element. This element is described in more detail in the section "DatadiagramML schema."

The following lines contain several attributes of the **VisioDocument** element, including two, **key** and **start**, which Visio uses to verify that the VDX document has not been modified outside of Visio since the last time it was saved in Visio. These attributes take the place of the `<?integrity ?>` processing-instruction tag that served the same purpose in Visio 2002.

For documents that you create outside of Visio, it is best to omit the **key** and **start** attributes in order to improve performance. If the **key** and **start** attributes are present, Visio will do a full test of the contents of the file. For documents created outside of Visio, this test will produce the same results, regardless of whether the **key** and **start** attributes are present.

For details about validating XML files against the DatadiagramML schema, see *Loading Files that were Created or Edited Outside of Visio* in the DatadiagramML Schema Reference.

Also notice the required **xmlns** attribute in the **VisioDocument** element. Every VDX document created in Microsoft Office Visio 2003 must have a namespace declaration of "http://schemas.microsoft.com/visio/2003/core". If there is no namespace declaration, or if the namespace declaration matches that used in Microsoft Visio 2002, Microsoft Office Visio 2003 will assume the VDX file contains a Microsoft Visio 2002 document and convert the document to the file format of the previous version before loading it.

DatadiagramML file formats

Following are the three extensions for DatadiagramML files:

- .vdx for drawing files
- .vtx for template files
- .vsx for stencil files

Files in .vdx, .vtx, or .vsx format are designed to be peers of the Visio document binary formats— .vsd, .vst, or .vss, respectively. For example, a .vsd file can be opened and saved as a .vdx file with no loss of information.

You can use Automation interface calls to manage an XML file that is open in Visio (for example, you can open, save, export, or import DatadiagramML files).

> **Note** You can use Automation calls to manipulate objects in a DatadiagramML document only while it is open in Visio. To retain changes you make by using Automation, you must save the document before closing it.

DatadiagramML schema

Visio provides an XML vocabulary or *schema* that is the set of XML tags that define a Visio document's data elements and attributes and their containment relationships. Developers can refer to the schema to see how to access the data within the DatadiagramML document structure.

The XML statements that follow show the top-level container, **VisioDocument**. This contains the **DocumentProperties** container, which in turn contains **Creator**, **Template**, **Company**, and other elements.

```
<VisioDocument
```

```
   ...>
   <DocumentProperties>
      <Template>C:\Program Files\Microsoft
Office\Visio11\1033\BASICD_M.VST</Template>
      <Company>MSFT</Company>
      ...
```

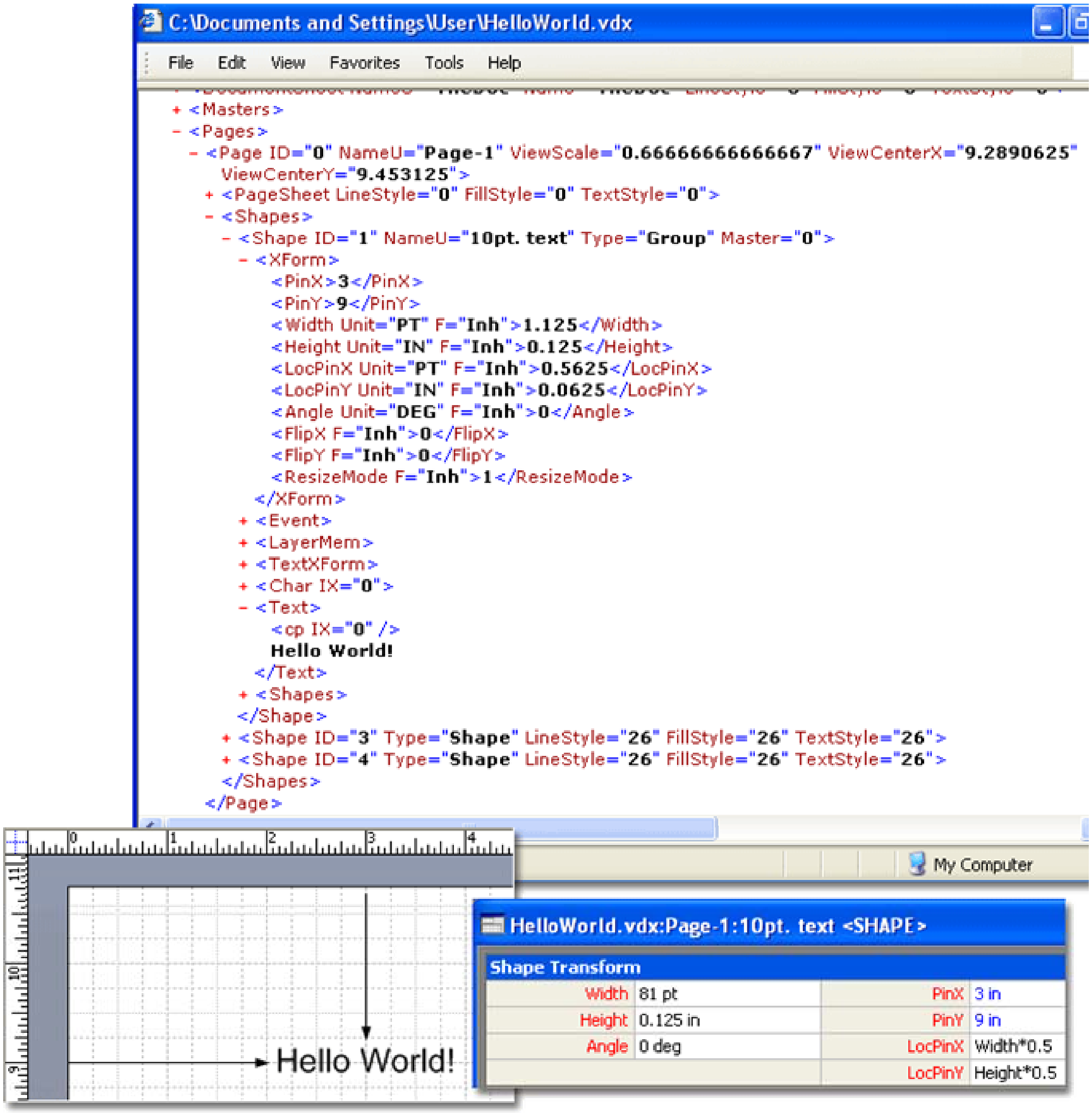
For more information about the DatadiagramML hierarchical tag structure, see *About the DatadiagramML Schema* in the DatadiagramML Schema Reference. For a representation of the schema itself, see *DatadiagramML Schema* in the DatadiagramML Schema Reference. You can also download the DatadiagramML schema from the Microsoft Office Developer Center on MSDN, the Microsoft Developer Network.

Using DatadiagramML

Because DatadiagramML is a text-based format, you can take advantage of all the text-based tools available for document management. Any text utilities for archiving, differencing, searching, indexing, or versioning can be used to process your DatadiagramML files. You can:

- Create a multifile utility to search all the text in your DatadiagramML documents without running Visio.
- Index your files and search on more than just document properties.
- Post DatadiagramML files on the Web to easily share data.
- Retrieve data embedded in the documents and run offline data processing and data analyzing applications.

Extracting data from various container-level elements is relatively straightforward. For example, the following illustration shows a Visio drawing that contains two arrows and the text "Hello world!", the corresponding Shape Transform section in the ShapeSheet window, and an excerpt from the resulting .vdx file.



C:\Documents and Settings\User\HelloWorld.vdx
File
Edit
View
Favorites
Tools
Help
My Computer
0
1
2
3
4
11
10
9
Hello World!
HelloWorld.vdx:Page-1:10pt. text <SHAPE>

```
+ <Masters>
- <Pages>
  - <Page ID="0" NameU="Page-1" ViewScale="0.66666666666667" ViewCenterX="9.2890625"
      ViewCenterY="9.453125">
    + <PageSheet LineStyle="0" FillStyle="0" TextStyle="0">
    - <Shapes>
      - <Shape ID="1" NameU="10pt. text" Type="Group" Master="0">
        - <XForm>
            <PinX>3</PinX>
            <PinY>9</PinY>
            <Width Unit="PT" F="Inh">1.125</Width>
            <Height Unit="IN" F="Inh">0.125</Height>
            <LocPinX Unit="PT" F="Inh">0.5625</LocPinX>
            <LocPinY Unit="IN" F="Inh">0.0625</LocPinY>
            <Angle Unit="DEG" F="Inh">0</Angle>
            <FlipX F="Inh">0</FlipX>
            <FlipY F="Inh">0</FlipY>
            <ResizeMode F="Inh">1</ResizeMode>
          </XForm>
        + <Event>
        + <LayerMem>
        + <TextXForm>
        + <Char IX="0">
        - <Text>
            <cp IX="0" />
            Hello World!
          </Text>
        + <Shapes>
        </Shape>
      + <Shape ID="3" Type="Shape" LineStyle="26" FillStyle="26" TextStyle="26">
      + <Shape ID="4" Type="Shape" LineStyle="26" FillStyle="26" TextStyle="26">
      </Shapes>
    </Page>
```

Shape Transform			
Width	81 pt	PinX	3 in
Height	0.125 in	PinY	9 in
Angle	0 deg	LocPinX	Width*0.5
		LocPinY	Height*0.5

DatadiagramML file format excerpt showing text position

The "Hello world!" text in the drawing is centered at X=3.0 inches and Y=9.0 inches. You can see these values in the drawing view on the rulers and in the PinX and PinY cells. You can also find these values in the XML tags for the **PinX** and **PinY** elements, and the text itself after the `<Text>` tag highlighted in bold by Microsoft Internet Explorer. The Internet Explorer window shows a portion of the XML data created by saving the drawing in DatadiagramML drawing file format (.vdx).

To view and edit the XML text of a DatadiagramML file, open a .vdx file in Notepad or in an XML editing tool.

Indents are disabled by default in .vdx files to reduce file size. To enable indents for DatadiagramML file output, see the section about white space in *Round-tripping DatadiagramML Files* in the DatadiagramML Schema Reference.

To view a .vdx file in Internet Explorer (version 5.0 or later), change the file extension from ".vdx" to ".xml," and then open the file in Internet Explorer. However, you must change the file extension back to .vdx if you want to open the file again in Visio.

For complete information about traversing the DatadiagramML tag structure, see *About the DatadiagramML Schema* in the DatadiagramML Schema Reference.

Validation of third-party DatadiagramML files

When you save a Visio document in XML file format, the first line in the XML document (`<?xml version='1.0' encoding='utf-8' ?>`) is a processing instruction that identifies the document as XML.

The next few lines contain the opening tag for the top-level DatadiagramML container, the **VisioDocument** element, and several attributes of the **VisioDocument** element, including two, **key** and **start**, which Visio uses to verify that the VDX document has not been modified outside of Visio since the last time it was saved in Visio. These attributes are only present in files that have been saved by Visio, and they can be thought of as a "stamp" of file integrity, allowing Visio to skip the time-consuming step of recalculating the values of all cell elements that contain formulas when you open the document in Visio. Files such as these that Visio has saved are called *trusted* files.

Third parties can also create and edit XML files outside of Visio that follow the DatadiagramML schema. These files are called *untrusted* files. If an untrusted file does not carry the stamp, or if the file carries a stamp that indicates the file has been modified outside of Visio, Visio checks that the XML is well-formed and valid, which can increase the time it takes to open an untrusted file. If you then save the file in Visio as a DatadiagramML document, the file will thereafter contain a stamp that indicates the document is a trusted file.

> **Note** Visio will not load files that are not well-formed or that will not pass validation against the Visio schema.

For more details about working with untrusted files, see *Loading Files that were Created or Edited Outside of Visio* in the DatadiagramML Schema Reference.

Round-tripping documents

Opening a Visio document, saving it as a DatadiagramML document, and then reopening the DatadiagramML document shows that you can *round-trip* all the Visio data in the document. That is, no information is lost between the binary version and the DatadiagramML version of the document.

XML files that are created or edited outside of Visio, and then opened and resaved in Visio, do not necessarily appear identical, but the information should be close to identical. For example, white space may or may not be preserved (depending on the element), and some elements might get reordered (or added in some cases) in the file in a way that is preferred by Visio.

For more details about some of the DatadiagramML file differences that may occur in round-tripping, see *Round-tripping DatadiagramML Files* in the DatadiagramML Schema Reference.

Including custom data within a DatadiagramML file

You can include custom XML data within your DatadiagramML file, provided the data contains well-formed XML that complies with the DatadiagramML schema and the internal rules of Visio.

You can embed custom XML in a DatadiagramML file in two ways:

- As *unknown XML*, which is XML that Visio does not understand but will round-trip in a DatadiagramML file when contained in particular elements.
- As *solution XML*, which is well-formed XML contained within a **SolutionXML** element that provides a standardized means of persisting solution data.

At the document level, you can store XML data by using the **SolutionXML** element contained immediately within the **VisioDocument** element. At the cell level, you can store well-formed XML data in selected cells by using a string value within a **SolutionXML** element. You can then manipulate the XML data in your solution at run time by using Automation.

Beginning with Microsoft Office Visio 2003, you can test the validity of changes you make to **SolutionXML** elements without having to reopen the file in Visio by turning on the **XMLValidateOnSave** registry key.

For information about embedding and extracting solution-specific data placed in drawings, stencils, or templates by a third-party solution provider, and about enabling validation on save, see *Embedding Custom XML in a DatadiagramML File* in the DatadiagramML Schema Reference.

Working with Microsoft Visio 2002 XML files in Microsoft Office Visio 2003

You can open a .vdx file created in Microsoft Visio 2002 in Microsoft Office Visio 2003. Visio will automatically convert the file to the newer format. However, once you make changes to the file and save it in Microsoft Office Visio 2003, you will no longer be able to open the file in Microsoft Visio 2002. Nor can you save the modified file in Microsoft Visio 2002 .vdx format from Microsoft Office Visio 2003.

To be able to open such a file in Microsoft Visio 2002 once again, save it in .vsd format. Then you can open it in Microsoft Visio 2002 and, if you wish, save it again in .vdx file format.

XML resources

You can use a variety of tools and information to work with XML in Visio, including the following resources:

- The Visio schema files, articles, and samples at the Microsoft Office Developer Center on MSDN, the Microsoft Developer Network, can offer specific information for working with DatadiagramML files.

- To parse XML text and retrieve the data within the tags, you can use the Microsoft Document Object Model (DOM) or Simple API for XML (SAX2) provided with Microsoft's 5.0 XML Parser library (msXML5.dll), which is distributed with Visio, or a later version, available on the Microsoft Developer Network Web site. Alternatively, you can use any XML parser that supports the use of namespaces and of XML Schema definition language (XSD).
- The languages related to XSL (Extensible StyleSheet Language) and XSLT (XSL Transformations) enable you to display or further process the data, including creating other HTML or XML documents.
- For information about working with XML as a solution across Microsoft Office 2003 products, see the XML reference material in the MSDN Library and the information on Office XML in the documentation for the respective Office applications.