Introducing the Office (2007) Open XML File Formats

By Archiveddocs

Archived: 2026-04-05 20:08:42 UTC

Summary: Learn the benefits of the Office Open XML Formats. Users can exchange data between Office applications and enterprise systems using XML and ZIP technologies. Documents are universally accessible. And, you reduce the risk of damaged files. (26 printed pages)

Frank Rice, Microsoft Corporation

May 2006

Applies to: 2007 Microsoft Office Suites, Microsoft Office Excel 2007, Microsoft Office PowerPoint 2007, Microsoft Office Word 2007

Contents

Introduction to the Office Open XML File Formats

Following the advent of XML in the 1990s, corporate computing customers began to realize the business value in adopting open formats and standardization in the computer products and applications that they relied on. IT professionals benefited from the common data format possible with XML because of its capacity to be read by applications, platforms, and Internet browsers.

Likewise, with the adoption of support for XML in Microsoft Office 2000, developers began to see the need to transition from the binary file formats seen in previous versions of Microsoft Office to the XML format. Binary files (.doc, .dot, .xls, and .ppt files), which for years did a great job of storing and transporting data, were not able to meet the new workplace challenges that included easily moving data between disparate applications, and allowing users to glean business insight from that data.

The 2007 Microsoft Office system continues with this transition by adopting an XML-based file format for Microsoft Office Excel 2007, Microsoft Office Word 2007, and Microsoft Office PowerPoint 2007. The new file format, called Office Open XML Formats, addresses these workplace issues with changes that affect the way that you approach solutions based on Microsoft Office documents.

The new formats improve file and data management, data recovery, and interoperability with line-of-business systems. They extend what is possible with the binary files of earlier versions. Any application that supports XML can access and work with data in the new file format. The application does not need to be part of the Microsoft Office system or even a Microsoft product. Users can also use standard transformations to extract or repurpose the data. In addition, security concerns are drastically reduced because the information is stored in XML, which is essentially plain text. Thus, the data can pass through corporate firewalls without hindrance.

Note

Do not confuse the Office XML Formats with the Microsoft Windows XML Paper Specification format. Office XML Formats use the Open Packaging Conventions, also used by the XML Paper Specification (XPS). However, the formats are different in several important ways. The XPS is a paginated, fixed document format introduced for the Microsoft Windows Vista operating system. Office XML Formats are fully editable file formats for Office Word 2007, Office Excel 2007, and Office PowerPoint 2007. Although they share similarities in their use of XML and ZIP compression, they are different in file format design and intended use. For more information, see the Open XML Formats Resource Center.

New File Format Scenarios

The Office XML Formats change the way that customers work with data. Consider some of the following scenarios, now possible with the new file formats:

- A server-side process can read one of the items out of a ZIP archive without having to extract them all. For example, you can extract only the XML file containing the document's data without opening the files that contain the document's formatting, properties, and other peripheral information.
- In most scenarios, the new file container is transparent to the user. It looks like a typical binary document file. However, the new XML format document is compressed and smaller, making it easier to send to other users through e-mail or on other media.
- Virus scanners and server applications can easily extract the container to view just the items affected directly, without the worry of corrupting the container or the contents inside.

- Files that are saved in the new file format cannot contain executable macro code. (This does not apply to macro-enabled files, discussed elsewhere in this article.) Therefore, they are considered "safe" and can pass through firewalls and be sent through e-mail applications without security concerns. This behavior is enforced by the Microsoft Office applications. If a macro part is added to an Office XML Format document, Office prevents opening the document.

  Note

  Some ZIP applications allow you to create encrypted files. The new file format does not create encrypted files. If you attempt to read an encrypted file, it stops reading the file and returns an error.

- If one item in the container becomes corrupted, the other items continue to be available. For example, if a user tries to open a file created from a damaged disk and the file is corrupt, the Office file-recovery mechanism rebuilds the central directory and fully recovers the contents of the file.

- Office developers can use Microsoft WinFX application programming interfaces (APIs) to create a valid XPS file. These APIs are the way that solutions interact with the content of the new file format. Office developers can also use the APIs to open a ZIP archive created by another ZIP application, assuming it does not use one of the ZIP features not supported by Office.

  Note

  Microsoft Office creates files conforming to the Open Packaging Conventions that the WinFX APIs can read, and conversely, WinFX can create packages that you can open using Microsoft Office applications. The WinFX APIs work with previous versions of Microsoft Windows, including the following: Microsoft Windows Vista, Microsoft Windows Server 2003, and Microsoft Windows XP. Additionally, converters are available for Microsoft Office 2000, Microsoft Office XP, and Microsoft Office 2003 Editions.

The new file format also changes the way that Office users work with the applications that they use in their daily tasks. The following scenarios illustrate ways that the new format changes the way that you accomplish work using the 2007 Office release:

- Howard, a curious user, finds that the new file format container in the 2007 Office release is a ZIP file. Howard wants to see this for himself, so he opens one of the documents he created in Word 2007 using a ZIP application. He sees some files that look like XML document content, and a few images that match what he had seen in the original document. Intrigued, the curious Howard opens one of the XML files in Notepad to see what sort of XML file Word generates. Satisfied at how easy it is to peer inside the document, he closes Notepad and the ZIP application and reopens the file in Word to continue working on the document.

- Jan needs to change a document property in her Word 2007 document. She opens the new format file in a ZIP product, extracts the XML item containing the document properties, makes the change, adds the item back into the container, and then saves and closes the container. She then opens the document in Word, and finds that all of her content is still there and that the only change is to the property that she modified.

- The logo for a consulting company changed to reflect its new mission. The IT department is given the task of changing the logo in the thousands of documents currently stored on a server. In previous versions of Microsoft Office, it was necessary to either open each document individually and delete the old logo and paste the new logo, or to create and test a complex custom application to automate the task. With the new file format, the IT department creates a batch process that navigates the file structure to locate the graphic in the media folder (which is the same for each document) and swaps out the new graphic. Now when the document opens, the new logo automatically displays.

- A law firm must protect the confidentiality of their customer's data and of their own data. Based on standard practices, the firm knows that documents, including those created in Word, store several properties—some visible and some hidden within the document. Some of those properties can contain information that is customer-sensitive and company proprietary. Through trial and error, the firm created a procedure to remove this information. However, the process is time-consuming and, because it is dependent on the user following the procedure exactly, subject to mistakes. With the new file format, the IT department develops a simple batch process that walks the folder structure to remove the targeted parts in the package. The firm can now be confident that all sensitive and propriety information is safely removed from the document.

- Denise is putting the final changes on the dissertation for her master's degree. After adding the bibliography, she prepares to perform the final save. Much to her horror, the save stops prematurely and she sees a message telling her that the document appears to be corrupt. After the shock subsides, Denise remembers that she is using the new file format. She imports the text into a new document, reattaches the pictures, and includes the formatting and style parts to reconstruct the document successfully. She then saves the new document.

- Elizabeth is a Windows 2000 user who runs Office 2000 with the converter installed that is included in the 2007 Office release. She can open a Word XML Format document. In addition, she can edit the document and re-save it as a document in the new format because the converters provide open and save support.

Benefits of the New File Formats

The new Office XML Formats introduce a number of benefits that help not only developers and the solutions they build, but also individual users and organizations of all sizes.

The following highlights are some of the additional overall benefits of the Office XML Formats:

- **Easily integrate business information with documents.** Office XML Formats enable rapid creation of documents from disparate data sources, accelerating document assembly, data mining, and content reuse. Exchanging data between Office applications and enterprise business systems is simplified. In addition, you can alter the information inside an Office document or create one from the document's components

without using the Office application. Employees can improve productivity by publishing, searching, and reusing information more quickly and accurately in the application they choose, as long as it supports reading and writing XML.

- **Open and royalty-free.** The Office XML Formats are based on XML and ZIP technologies, so they are universally accessible. The specification for the formats and schemas will be published and made available under the same royalty-free license that exists today for the Microsoft Office 2003 Reference Schemas, and that is openly offered and available for broad industry use.

- **Interoperable.** With industry-standard XML at the core of the Office XML Formats, exchanging data between Microsoft Office applications and enterprise business systems is simplified. Without requiring access to the Office applications, solutions can alter information inside an Office document or create a document by using standard tools and technologies capable of manipulating XML. The new formats enable you to build archives of documents without using Office code.

- **Robust.** The Office XML Formats are designed to be more robust than the binary formats, and, therefore, to help reduce the risk of lost information due to damaged or corrupted files. Even documents created or altered outside of Office are less likely to corrupt, because Office applications are designed to recover documents with improved reliability by using the new formats. With more and more documents traveling through e-mail attachments or removable storage, the chance of a network or storage failure increases the possibility of a document becoming corrupt.

  The new file formats improve data recovery by segmenting and separately storing each part within the file package. This has the potential to save companies tremendous amounts of money and time spent recovering lost data. When one file component is corrupt, the remainder of the file is still open within the application. For example, if a chart becomes damaged, this does not prevent the customer from opening the other parts of the document, without the chart. In addition, Office applications can detect these defects, and attempt to repair a document when opening it by restoring the proper data structure to the document.

- **Efficient.** The Office XML Formats use ZIP and compression technologies to store documents. A significant benefit of the new formats is substantially smaller file sizes—up to 75 percent smaller than comparable binary documents. This is one of the advantages of using the combination of XML and the ZIP technologies for storing files. Because XML is a text–based format that compresses very well, and the ZIP container supports compressing the contents, users can obtain significant reductions in file size. This type of file compression offers potential cost savings because it reduces the disk space required to store files and decreases the bandwidth needed to transport files through e-mail, over networks, and across the Web.

- **Secure.** The openness of the Office XML Formats translates to more secure and transparent files. You can share documents confidently because you can easily identify and remove personally identifiable information and business-sensitive information, such as user names, comments, and file paths. Similarly, you can identify files containing content, such as OLE objects or Microsoft Visual Basic for Applications (VBA) code, for special processing. The file formats also help to improve security against documents with embedded code or macros. By default, the new Word 2007, Excel 2007, and PowerPoint 2007 file formats do not execute embedded code. So, if a person receives an e-mail message with a Word document attached,

he or she could open the attachment knowing the document does not execute harmful code. The Office XML Formats include a special-purpose format with a separate extension for files with embedded code, enabling IT staff to quickly identify files that contain code.

- **Backward-compatible.** The 2007 Microsoft Office system is backward-compatible with these earlier versions: Microsoft Office 2000, Microsoft Office XP, and Microsoft Office 2003. Users of these versions can adopt the new format with little effort and continue to gain maximum benefit from existing files. Specifically, they can continue to use the older .doc, .xls, and .ppt binary formats, which are fully compatible with the 2007 file format. Users of earlier Office versions can download a free update that enables them to open and edit files in the new format from within their earlier versions. Users who install the 2007 Office release can set the default file formats to whichever format they choose. This helps ensure that users can continue to work with third-party solutions based on earlier versions, and work with their colleagues, suppliers, customers, and others who have upgraded.

Glossary for Office Open XML Formats

The following list defines terms used in this article and related content:

- **API** A set of functions or methods used to access software functionality. API is an acronym for application programming interface.
- **Converters** Free tools that open files created in the Office XML Formats in previous Office versions.

  > **Note**
  >
  > The converters read, honor, and apply information rights management (IRM) protection to documents. A document that includes IRM protection continues to be protected regardless of whether the recipient is using the 2007 Office release or a previous version.

- **Forward-compatibility** The ability of an earlier version of an application to open files from a later version and ignore features that are not implemented in the earlier version. For example, Word 2003 is forward-compatible with Word 2007 because it can successfully open Word 2007 files using a converter.
- **Office Open XML Formats** A structure of building blocks and relationships used for composing, packaging, distributing, and rendering document-centered content. These building blocks define a platform-independent framework for document formats that enable software applications to generate, exchange, and display documents reliably and consistently.
- **Package** The ZIP container that holds the components (parts) that comprise the document, as defined by the Open Packaging Conventions specification.
- **Part** Corresponds to one file in the package. For example, if a user right-clicks an Excel 2007 file and chooses to extract it, he or she sees files, such as a workbook.xml file and several sheet*n*.xml files. Each of those files is a part in the package.

- **Relationships** The method used to specify how the collection of parts come together to form a document. This method specifies the connection between a source part and a target resource. Relationships are stored within XML parts (for example, /_rels/.rels) in the document package.
- **XML** Extensible Markup Language (XML) is a simple, flexible text format designed for electronic publishing and the exchange of a wide variety of data on the Internet and elsewhere.
- **ZIP** An industry-standard compression archives format used to store and transport files between computers, through e-mail, or over the Internet.

Structure of the Office XML Formats

The new file format container is based on the simple and parts-based compressed ZIP file format specification. At the core of the new Office XML Formats is the use of XML reference schemas and a ZIP container. Each file is comprised of a collection of any number of parts; this collection defines the document.

Document parts are stored in the container file or package using the industry-standard ZIP format. Most parts are XML files that describe application data, metadata, and even customer data, stored inside the container file. Other non-XML parts may also be included within the container package, including such parts as binary files representing images or OLE objects embedded in the document. In addition, there are relationship parts that specify the relationships between parts; this design provides the structure for an Office file. While the parts make up the content of the file, the relationships describe how the pieces of content work together.

The results are XML file formats for Office documents that are tightly integrated but modular and highly flexible. The next few sections describe each component of the Office XML Formats in detail. They also describe the Office applications using the new file format.

Note

To understand the composition of an Office XML Formats file, you might want to extract a file. To open the file, it is assumed that you have a ZIP application, such as WinZip from WinZip Computing Corporation, installed on your computer.

To open a Word 2007 XML file

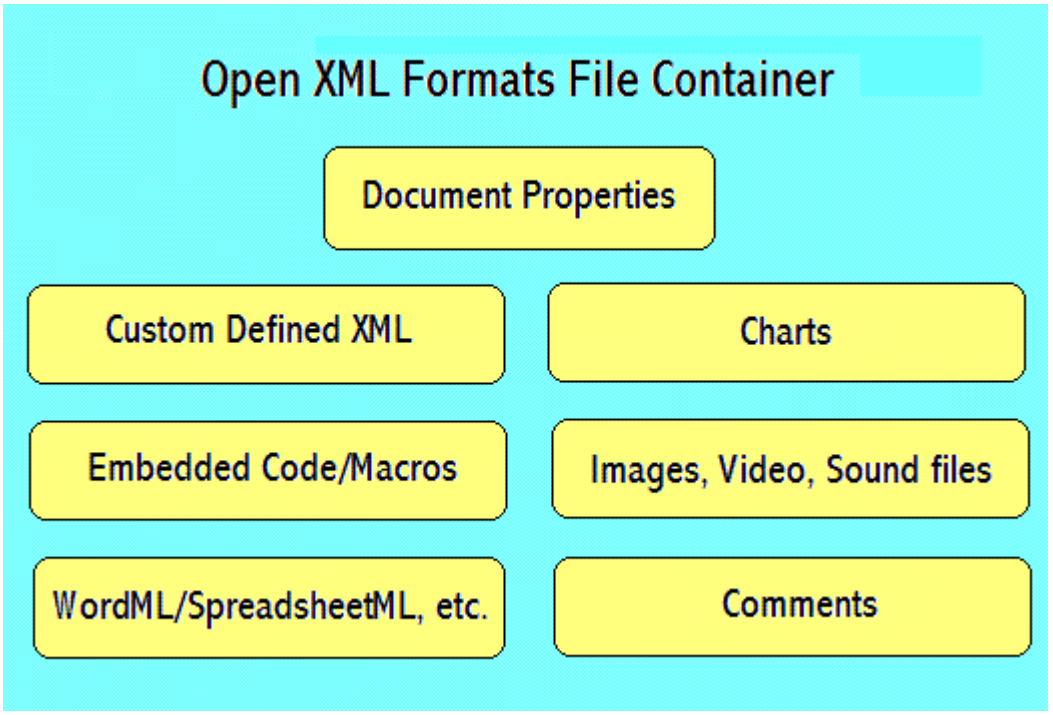
1. Create a temporary folder in which to store the file and its parts.
2. Save a Word 2007 document, containing text, pictures, and other elements, as a .docx file.
3. Add a .zip extension to the end of the file name.
4. Double-click the file. It will open in the ZIP application. You can see the parts that comprise the file.
5. Extract the parts to the folder that you created previously.

ZIP Package

Many elements go into creating a Microsoft Office document. Some of these are commonly shared across all the Office applications, for example, document properties, style sheets, charts, hyperlinks, diagrams, and drawings. Other elements are specific to each application, like worksheets in Excel, slides in PowerPoint, or headers and footers in Word.

When users save a document with Office 2003 or previous versions of Microsoft Office, a single file is written to disk, which you can easily open. This metaphor is important to how documents are stored, managed, and shared in practice. By wrapping the individual parts of a 2007 Microsoft Office system file in a ZIP container, a document remains as a single file instance. The use of a single package file to represent the entity of a single document means users have the same experience as with previous Office versions when saving and opening Office (2007) documents. They can continue to work with just a single file.

Figure 1. The file format container in the 2007 release



With previous Office versions, developers who wanted to manipulate the content of an Office document needed to know how to read and write data according to the structured storage defined within the binary file. This process is complex and challenging, notably because the Office binary file formats were designed to be accessed primarily through the Office applications. The formats reflected the in-memory structures of the applications and could run on low-memory computers with slow hard drives. In addition, altering Office binary files programmatically without the Office applications was identified as a leading cause of file corruption. This deterred some developers from even attempting to alter the files.

ZIP was chosen as the package format for the Office XML Formats because it is a well-understood industry standard. There are many tools available today to work with the ZIP format, and using ZIP provides a flexible, modular structure that allows for an expansion of functionality, going forward. Therefore, you have access to the

complete contents of 2007 Microsoft Office system documents by using any of the numerous tools and technologies that work with industry-standard ZIP files. After you open a container file, you can manipulate any of the document parts found within the package that define the document. For example, you can open a Word 2007 document that uses Office XML Formats, locate the XML part that represents the body of the Word document, alter the part by using any technology capable of editing XML, and return the XML part to the container package to create an updated Office document.

Parts

Within an Office XML Formats package, many logical parts of the file are stored as individual files or parts. This modularity is one of the important characteristics of the file format. Modularity enables you to locate a specific part quickly and to work directly with just that part. You can edit, exchange, or even remove document parts depending on the preferred outcome of a specific business need.

All the Office applications share some types of parts, such as the thumbnail, metadata, media, and relationships parts. Others exist consistently within all files as a specific part, such as document properties. Many parts, however, are unique to the application document type they represent. For example, a worksheet part is only found in an Excel document, while a slide master part only appears in a PowerPoint document.

It is important to note that, with a few exceptions defined within the Open Packaging Conventions, the actual file directory structure is arbitrary. The relationships of the files within the package, not the file structure, are what determine file validity. You can rearrange and rename the parts of an Office (2007) XML Formats file inside its ZIP container, provided that you update the relationships properly so that the document parts continue to relate to one another as designed. If the relationships are accurate, the file opens without error. The initial file structure in an Office XML Formats file is the default structure that is created. This default structure enables you to determine the composition of Office XML Formats files easily. Provided that you keep the relationships current, you can change this default file structure. For more information about this, see Walkthrough: Word 2007 Open XML File Format.

Parts can be of different content types. Parts used to describe Microsoft Office applications data are stored as XML. These parts conform to the XML reference schema that defines the associated Office feature or object. For example, in an Excel 2007 file, the data that represents a worksheet is found in an XML part that adheres to the Office schema for an Excel worksheet. Additionally, if there were multiple worksheets in the workbook, there is a corresponding XML part stored in the package file for each worksheet. All of the schemas that represent default Office document parts will be fully documented and made available from Microsoft with a royalty-free usage license. Then, by using any standard XML-based technologies, you can apply your knowledge of the Office schemas to parse and create 2007 Microsoft Office system documents easily.

In many scenarios, it is advantageous to store parts in their native content type. These parts are not stored as XML. Images in an Office document in the 2007 release, for example, are stored as binary files (.png, .jpg, and other file types) within the document package. Therefore, you can open the package container by using a ZIP application and immediately view, edit, or replace the image in its native format. Not only is this storage approach more accessible, it also requires less internal processing and disk space than storing an image as encoded XML. Other notable parts stored as binary parts are VBA projects and embedded OLE objects. (Embedded OLE objects are

binary only if the associated OLE server provides only a binary representation. 2007 Microsoft Office system–embedded documents, for example, embed their contents as another package.) For developers, accessibility makes many scenarios more attractive. For example, you can build a solution that iterates a collection of 2007 Microsoft Office system documents to update an existing OLE object with a newer version. You can accomplish this and any number of other scenarios without having to use the Office applications or alter the document-specific XML. The next sections briefly describe some of the parts that are common to all of the Office products that support the Office XML Formats. The types and numbers of parts depend on the application that creates the ZIP container file. For example, Word 2007 creates document-related parts, but PowerPoint 2007 creates parts related to slide presentations.

_rels Folder

This folder contains a .rels file that defines the root relationships within the package. This is the first place you should go to start parsing through the package.

.rels File

Contains relationships based on the start part (the virtual start part). Relationships are defined with the following format:

```
<Relationship Id="someID" Type="relationshipType" Target="targetPart"/>
```

where **Id** is any string, as long as it is unique in the .rels file.

**Type **The type of the relationship that distinguishes relationships from one another and provides a hint as to the relationship's purpose. It points to the schema that defines Office XML Formats types.

**Target **Points to the folder and file that contain the target of the relationship (another part).

Table 1. Relationship types

Built-in Relationship Types
http://schemas.microsoft.com/office/2006/relationships/officeDocument
http://schemas.microsoft.com/office/2006/relationships/vbaProject
http://schemas.microsoft.com/office/2006/relationships/userXmlData

Built-in Relationship Types
http://schemas.microsoft.com/office/2006/relationships/hyperlink
http://schemas.microsoft.com/office/2006/relationships/styleSheet
http://schemas.microsoft.com/office/2006/relationships/comments
http://schemas.microsoft.com/office/2006/relationships/oleObject
http://schemas.microsoft.com/office/2006/relationships/e2Object
http://schemas.microsoft.com/office/2006/relationships/e1Object
http://schemas.microsoft.com/office/2006/relationships/image
http://schemas.microsoft.com/office/2006/relationships/sound
http://schemas.microsoft.com/office/2006/relationships/movie
http://schemas.microsoft.com/office/2006/relationships/slide
http://schemas.microsoft.com/office/2006/relationships/layout
http://schemas.microsoft.com/office/2006/relationships/notesslide
http://schemas.microsoft.com/office/2006/relationships/slidemaster
http://schemas.microsoft.com/office/2006/relationships/glossaryDoc

Built-in Relationship Types
http://schemas.microsoft.com/office/2006/relationships/cfChunk
http://schemas.microsoft.com/office/2006/relationships/dataStoreItem
http://schemas.microsoft.com/office/2006/relationships/embeddedFont
http://schemas.microsoft.com/office/2006/relationships/embeddedMetroObject
http://schemas.microsoft.com/office/2006/relationships/chart
http://schemas.microsoft.com/office/2006/relationships/activeXControl
http://schemas.microsoft.com/office/2005/relationships/diagram
http://schemas.microsoft.com/office/2005/relationships/diagramData
http://schemas.microsoft.com/office/2005/relationships/diagramStyle
http://schemas.microsoft.com/office/2005/relationships/diagramColorTrans
http://schemas.microsoft.com/office/2005/relationships/diagramDefinition
http://schemas.microsoft.com/package/2005/02/md/core-properties
http://schemas.microsoft.com/office/2006/relationships/docPropsApp
http://schemas.microsoft.com/office/2006/relationships/docPropsCustom

Built-in Relationship Types
http://schemas.microsoft.com/ office/2006/relationships/documentThumbnail
http://schemas.microsoft.com/office/2006/relationships/glossaryDoc

Main Document Part

The target of the http://schemas.microsoft.com/office/2006/relationships/officeDocument relationship is the main part defining the document (the presentation part for PowerPoint, the workbook part for Excel, or the document part for Word). All other relationships are based on the main document part.

Application Folder (such as Word)

Contains application-specific, document component files such as (for Word):

- **wordDocument.xml** Contains the data (text) in the document, plus styles and font settings.
- **footer.xml** Contains information about footers in the document, such as what page they are on and some styles information.
- **header.xml** Contains information similar to that found in the footer.xml file, but regarding headers.
- **wordDocument.doc** Is a copy of the original document.
- **styles.xml** Contains information about the styles found in the document, such as font sizes, table styles, and bulleted lists.

Audio File

Contains any audio-type files, such as .mid, .mp3, or .wav files.

Content_Types.xml File

Provides a listing of the content types for the other parts that are contained in the package. Content types are defined as the types of parts that can be stored in a package:

Table 2. Content types in a ZIP container

Built-in Content Types
application/vnd.ms.powerpoint.template.macroEnabled.12application/x-font
application/vnd.ms-excel.12application/x-font
application/vnd.ms-excel.addin.12application/xml
application/vnd.ms-excel.binary.12audio/aiff
application/vnd.ms-excel.macroEnabled.12audio/basic
application/vnd.ms-excel.macroEnabledTemplate.12audio/midi
application/vnd.ms-excel.template.12audio/mp3
application/vnd.ms-metro.core-properties+xmlaudio/mpegurl
application/vnd.ms-metro.relationships+xmlaudio/wav
application/vnd.ms-office.activeX+xmlaudio/x-ms-wax
application/vnd.ms-office.chartaudio/x-ms-wma
application/vnd.ms-office.vbaProjectimage/bmp
application/vnd.ms-powerpoint.image/gif
application/vnd.ms-powerpoint.macroEnabled.12image/jpeg

Built-in Content Types
application/vnd.ms-powerpoint.main.12+xmlimage/png
application/vnd.ms-powerpoint.presentation.12image/tiff
application/vnd.ms-powerpoint.show.12image/xbm
application/vnd.ms-powerpoint.show.macroEnabled.12image/x-icon
application/vnd.ms-powerpoint.template.12video/avi
application/vnd.ms-word.document.12video/mpeg
application/vnd.ms-word.document.macroEnabled.12video/mpg
application/vnd.ms-word.document.macroEnabled.main+xmlvideo/x-ivf
application/vnd.ms-word.document.main+xmlvideo/x-ms-asf
application/vnd.ms-word.fontTable+xmlvideo/x-ms-asf-plugin
application/vnd.ms-word.listDefs+xmlvideo/x-ms-wm
application/vnd.ms-word.settings+xmlvideo/x-ms-wmv
application/vnd.ms-word.styles+xmlvideo/x-ms-wmx
application/vnd.ms-word.subDoc+xmlvideo/x-ms-wvx

Built-in Content Types
application/vnd.ms-word.template.12
application/vnd.ms-word.template.macroEnabled.12
application/vnd.ms-word.template.macroEnabled.main+xml
application/vnd.ms-word.template.main+xml

Document Properties Part

Contains the core document properties defined for all files conforming to XPS format, such as:

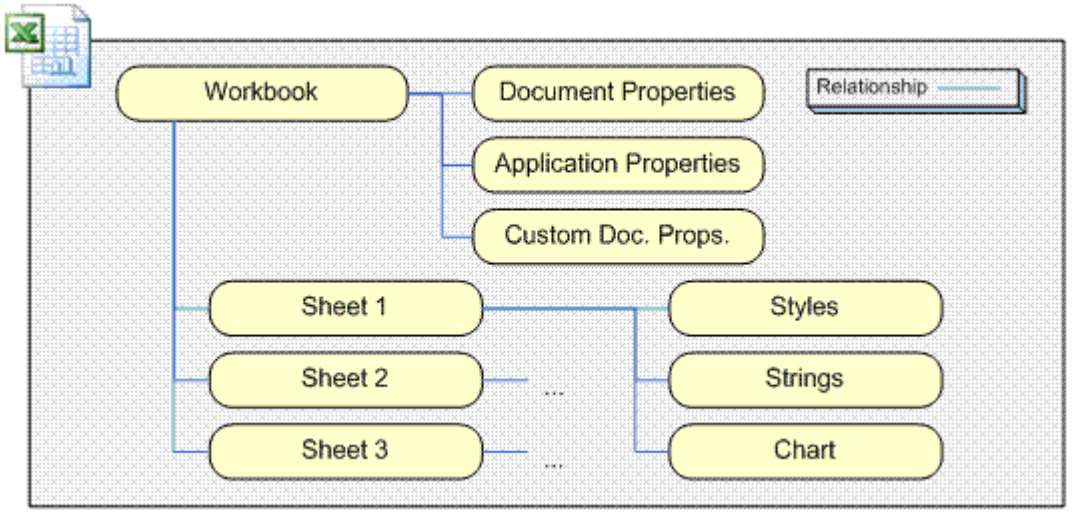
- Author
- Title
- Subject
- Comments
- Last Saved Date
- Created Date

Relationships

Parts are the individual elements that make up an Office document in the 2007 release. Relationships are the method used to specify how the collection of parts relate together to form the actual document. You use XML to define relationships. Relationships specify the connection between a source part and a target resource. For example, you can identify the connection between a slide and an image that appears on that slide by a relationship. Relationships are stored within XML parts or "relationship parts" in the document container. If a source part has multiple relationships, all subsequent relationships are listed in the same XML relationship part.

Relationships play an important role in Office XML Formats. Every document part is referred to by at least one relationship. The use of relationships makes it possible to discover how one part relates to another part without having to look within the content of parts. Within parts, all references to relationships are represented using a Relationship ID, which allows all connections between parts to stay independent of content-specific schema.

Figure 2. High-level relationship diagram of an Excel 2007 workbook



The following is an example of a relationship part in an Excel 2007 workbook containing two worksheets:

```
 <Relationships xmlns="http://schemas.microsoft.com/package/2005/06/relationships">
<Relationship ID="rId3"
   Type="http://schemas.microsoft.com/office/2005/8/relationships/xlStyles"
      Target="styles.xml"/>
   <Relationship ID="rId2"
Type="http://schemas.microsoft.com/office/2005/8/relationships/xlWorksheet"
         Target="worksheets/Sheet2.xml"/>
   <Relationship ID="rId1"
      Type="http://schemas.microsoft.com/office/2005/8/relationships/xlWorksheet"
         Target="worksheets/Sheet1.xml"/>
   <Relationship ID="rId5"
Type="http://schemas.microsoft.com/office/2005/8/relationships/xlMetadata"
         Target="metadata.xml"/>
   <Relationship ID="rId4"
Type="http://schemas.microsoft.com/office/2005/8/relationships/xlSharedStrings"
         Target="strings.xml"/>
</Relationships>
```

It is also important to note that relationships represent not only internal document references but also external resources. For example, if a document contains linked pictures or objects, these are also represented using relationships. This makes links in a document to external sources easy to locate, inspect, and change. It offers you the opportunity to repair broken external links, validate unfamiliar sources, or remove potentially harmful links.

The use of relationships in Office XML Formats benefits developers in a number of ways. Relationships simplify the process of locating content within a document because you do not need to parse document-specific XML to find parts—you also do not need to parse document-specific XML to find internal and external document resources. Relationships enable you to take inventory of all the content within a document quickly. For example, if you need to count the number of worksheets in an Excel workbook, you can inspect the relationships for how many sheet parts exist. You can also use relationships to examine the type of content in a document. This is

helpful in situations where you need to identify if a document contains a particular type of content that may be harmful (such as an OLE object that is suspect) or helpful (such as in a scenario where you want to extract all JPEG images from a document for reuse elsewhere).

Relationships also enable you to manipulate documents without having to learn application-specific syntax or content markup. For example, without any knowledge of how to program PowerPoint, a developer solution could easily remove extraneous slides for a presentation by editing the document's relationships.

Macro-Enabled Files vs. Macro-Free Files

Default 2007 Microsoft Office system documents saved in Office XML Formats are intended to be macro-free files, and therefore cannot contain code. This behavior ensures that malicious code, residing in a default document, can never be executed unexpectedly. While 2007 Microsoft Office system documents can still contain and use macros, the user or developer must save these documents as a macro-enabled document type. This safeguard will not affect your ability to build solutions, but allows organizations to use documents with more confidence.

Macro-enabled files have the exact same file format as macro-free files, but contain additional parts that macro-free files do not. The additional parts depend on the type of automation found in the document. A macro-enabled file that uses VBA contains a binary part that stores the VBA project. Any Excel workbook that uses Excel 4.0–style macros (XLM macros) or any PowerPoint presentation that contains command buttons are also saved as macro-enabled files. If a code-specific part is found in a macro-free file, whether placed there accidentally or maliciously, the Office applications will not allow the code to execute—without exception.

You can now determine if any code exists within a 2007 Microsoft Office system document before opening it. Previously this "advance notice" was not something that could be easily accomplished outside Office. You can inspect the package file for the existence of any code-based parts and relationships without running Office and without running potentially risky code. If a file looks suspicious, you can remove any parts capable of executing code from the file, so that the code can cause no harm.

File Name Extensions

2007 Microsoft Office system documents saved by using Office XML Formats have new file name extensions that allow Office to differentiate these file format documents from binary documents used by previous Office versions. The new extensions borrow from the existing binary file name extensions by appending a letter to the end of the suffix. The default extensions for documents created in Word 2007, Excel 2007, and PowerPoint 2007 using the new file formats append the letter "x" and are .docx, .xlsx, and .pptx, respectively. Other Office document types that use the new file formats (including templates, add-ins, and PowerPoint shows) also receive new extensions.

Another new change introduced in the 2007 Office release is that there are different extensions for files that are macro-enabled versus those that are macro-free. Macro-enabled documents include a file name extension that ends with the letter "m" instead of an "x." For example, a macro-enabled Word 2007 document has the .docm extension, and thereby allows any users or software applications, before a document opens, to identify that it contains code.

Table 3. List of file name extensions for Word 2007 document types

Word 2007 File Type	**Extension**
Word 2007 XML Document	.docx
Word 2007 XML Macro-Enabled Document	.docm
Word 2007 XML Template	.dotx
Word 2007 XML Macro-Enabled Template	.dotm

Table 4. List of file name extensions for Excel 2007 document types

Excel 2007 File Type	**Extension**
Excel 2007 XML Workbook	.xlsx
Excel 2007 XML Macro-Enabled Workbook	.xlsm
Excel 2007 XML Template	.xltx
Excel 2007 XML Macro-Enabled Template	.xltm
Excel 2007 Binary Workbook	.xlsb
Excel 2007 XML Macro-Enabled Add-In	.xlam

Table 5. List of file name extensions for PowerPoint 2007 document types

PowerPoint 2007 File Type	Extension
PowerPoint 2007 XML Presentation	.pptx
PowerPoint 2007 Macro-Enabled XML Presentation	.pptm
PowerPoint 2007 XML Template	.potx
PowerPoint 2007 Macro-Enabled XML Template	.potm
PowerPoint 2007 Macro-Enabled XML Add-In	.ppam
PowerPoint 2007 XML Show	.ppsx
PowerPoint 2007 Macro-Enabled XML Show	.ppsm

Developing Solutions Using the Office XML Formats

The Office XML Formats introduce or improve on many types of solutions involving documents that you can build. You can access the contents of an Office document in the new file formats by using any tool or technology capable of working with ZIP archives. You can then manipulate the document content using any standard XML processing techniques, or, for parts that exist as embedded native formats (such as images), process using any appropriate tool for that object type.

In addition, being able to open the container file of a 2007 Microsoft Office system document manually as a ZIP archive has some interesting benefits for developers. For example, when building Office-based solutions, you can examine the contents and structure of a document without having to write any code. This facility can be very helpful in solution design and when building prototypes.

After you are inside a 2007 Microsoft Office system document, the structure makes it easy to navigate a document's parts and its relationships, whether it is to locate information, change content, or remove elements from a document. Having the use of XML, along with the published Office reference schemas, means you can easily create additional documents, add data to existing documents, or search for specific content in a body of documents.

The rest of this article explores some scenarios in which Office XML Formats enable document-based solutions. These few are only part of an almost endless list of possibilities:

- Data Interoperability
- Content Manipulation
- Content Sharing and Reuse
- Document Assembly
- Document Security
- Managing Sensitive Information
- Document Styling
- Document Profiling

Data Interoperability

The emergence of XML as a popular standard for data exchange means the new Office XML Formats make document-based data more accessible among heterogeneous systems. Whether users are sharing document data across a department, or two organizations are trading business data, XML as a default file format for Microsoft Office documents means Office applications can participate in business processes without the limitations previously imposed by the binary formats.

The openness of the new file formats unlocks data and introduces a broad, new level of integration beyond the desktop. For example, you could refer to the published specification of the new file formats to create data-rich documents without using an Office application. Server-side applications could process documents in bulk to enable large-scale solutions that mesh enterprise data within the familiar, flexible Office applications. You could use standard XML protocols, such as XPath (a common XML query language) and Extensible Stylesheet Language Transformations (XSLT) to retrieve data from documents or to update the contents inside a document from external data.

One such scenario could involve personalizing thousands of documents to distribute to customers. You could insert information programmatically into a standard document template by using a server application that uses XML that you extracted from an enterprise database or customer relationship management (CRM) application. Creating these documents is highly efficient because there is no requirement to run Office applications; yet the capability still exists for producing high-quality, rich Office documents.

The use of custom schemas in Office is another way you can use documents to share data. Information that was once locked in a binary format is now easily accessible and, therefore, documents can serve as openly exchangeable data sources. Custom schemas not only make insertion or extraction of data simple, they also add structure to documents and are capable of enforcing data validation.

Content Manipulation

Editing the contents of existing Office documents is another valuable example where Office XML Formats enhance a process. The edit could involve updating small amounts of data, swapping entire parts, removing parts, or adding new parts altogether. By using relationships and parts, the new file formats make content easier to find and manipulate. The use of XML and XML schema means you can use common XML technologies, such as XPath and XSLT, to edit data within document parts in virtually endless ways.

One scenario might involve the need to edit text in the header of a Word document. Of course, it is not logical to automate that task for one document. But, in another scenario, what if a company merged and needed to update their new company name in the header of hundreds of different pieces of documentation? A developer could write code that loops through all the documents, locates the header part in the Word file structure, and performs an XPath query to find the old text. Then it could insert the new text, replace the header part, and repeat the process until every document is updated. Automation could save a lot of time, enable a process that might otherwise not be attempted, and prevent potential errors that might occur during a manual process.

Another scenario might be one in which an existing Office document must be updated—by changing only an entire part. In an Excel 2007 workbook, you could replace an entire worksheet that contained old data or outdated calculation models with a new one by overwriting its part. This kind of updating also applies to binary parts. You could swap an existing image, or even an OLE object, out for a new one, as necessary. You could update a Microsoft Office Visio drawing embedded as an OLE object in Office documents, for example, by overwriting that binary part. You could update URLs in hyperlinks to point to new locations.

Following are some additional application-specific scenarios.

Content Manipulation in Word 2007

It is a common business practice to incorporate "boilerplate" text inside a Word document. For example, an official legal disclaimer or a disclosure of terms and conditions can be required in every public document created by an organization. Another typical example of boilerplate is a "Company Overview" section that is used in authoring sales proposals or public releases of company announcements. Word offers features, such as AutoText, that are capable of accomplishing the insertion of formatted text, but this feature is limited in scale because it requires either Word automation or direct user interaction.

Word 2007 offers a very flexible alternative for you to insert content into a document. The Word XML Format allows you to add document parts, called document building blocks, that are referred to by the overall document when it opens in Word. This means you can build a library of document building blocks, which you can derive from document formats that Word is capable of rendering, and programmatically reuse them as needed in Word document solutions.

This broader ability to manipulate Word content offers some interesting scenarios, such as server-side document assembly. Going back to the example given previously, you can automatically insert a legal disclaimer into a document created on a server. Imagine a multinational company that requires that all of its documents contain a legal disclaimer in local languages. The company could create the appropriate language-specific disclaimers as .html files and save them on a server. An application that is constructing documents can insert the corresponding document fragment for the language required as a part inside the document container. This fragment is then rendered as a seamless part of a Word document.

Content Manipulation in Excel 2007

To optimize loading and saving performance and file size, Excel 2007 stores only one copy of repetitive text within the Excel file. To do so, Excel implements a shared string table in a document part specified by the target of the http://schemas.microsoft.com/office/2005/8/relationships/xlSharedStrings relationship. Each unique text value found within a workbook is listed once in this part. Individual worksheet cells then reference the string table to derive their values.

While this process optimizes the Excel XML Format, it also introduces some interesting opportunities for additional content manipulation solutions. Developers in a multinational organization could use the shared string table to offer a level of multilanguage support. Instead of building unique workbooks for each language supported, a single workbook could use string tables that correspond to different languages. Another possibility is to use string tables to search for keyword terms inside a collection of workbooks. Processing a single, text-only XML document of strings is faster and simpler than having to manipulate the Excel object model over many worksheets and workbooks.

Content Manipulation in PowerPoint 2007

When a PowerPoint 2007 presentation is stored using the PowerPoint XML Format, the content remains highly accessible. Because this is the first version of PowerPoint to offer an XML format, it opens up many scenarios not possible in previous versions. You now have full access to slides and slide notes as text. Solutions that require searching, indexing, and creating presentation content are now possible. You can easily produce data-driven presentations using XML. And, you can access slide masters and slide layouts through XML parts to programmatically format existing or new PowerPoint presentations.

You can take a different approach to assembling or reusing content from PowerPoint presentations by building an application that uses a catalog of slides stored independently of existing presentations. Slides are represented as individual XML parts, therefore, a solution can optimize the way an organization stores and manages PowerPoint 2007 slides as data. You can even write a slide "viewer" that allows a user to discover and select slides to build a presentation from outside PowerPoint. The application can even be Web-based to allow centralized management.

Content Sharing and Reuse

The modularity of Office XML Formats opens up the possibility for generating content once and then repurposing it in a number of other documents. As a developer, you can imagine building a number of core templates and reusing portions as building blocks for other documents. You could use a table created in one Word document, for example, in other Word documents. You can build charts (which have a common schema across 2007 Microsoft Office system applications) once and reuse them a number of times in different document types. The accessibility of the format lends itself to unlimited content-sharing opportunities.

One such scenario could be one in which there is a need to build a repository of images used in documents. You can create a solution that extracts images out of a collection of Office documents and allow users to reuse them from a single point of access. Because Office documents in the 2007 release store images intact as binary parts, you can build the solution and maintain a library of images easily. Then, users who want to incorporate previously used images do not need to browse through an entire collection of documents, opening and closing each

individually, to find images. They can use the custom application to find images in the repository and immediately insert them into the document with which they are working.

You can build a similar application that reuses document "thumbnail" images extracted from documents, and add a visual aspect to a document management process.

Document Assembly

One of the most common requests from developers is for the ability to create Microsoft Office documents on a server without automating the Microsoft Office applications. Organizations needing to produce complex, data-enriched documents or to assemble documents in mass quantities want more efficient processing for high-end purposes. Technically, Office applications were not written and were not supported to run from a server.

In the Microsoft Office 2003 Editions, the introduction of XML document formats that could be produced according to the Office 2003 XML Reference Schemas helped overcome this limitation. Any technology capable of assembling XML can build a Word or Excel document, as long as it adheres to the Office schemas. It was a tremendous advance at the time, but, unfortunately, it only applied to Excel and Word, and only Word truly offered full fidelity in its XML file support. The 2007 Office release builds on this effort by adding PowerPoint XML files and ensuring that both PowerPoint XML files and Excel XML files are also full-fidelity.

This advance in technology means that, with the 2007 Office release, you can build an Office solution that produces Excel, Word, and PowerPoint documents without ever opening Office. The solution must create XML according to the schemas in the 2007 release and build the package contents as defined by the Office XML Formats. Although the Office schemas are quite extensive, to fully represent the rich feature sets that the Microsoft Office applications provide, all structures defined by the format are not required to generate a document. Each of the Office applications is capable of opening the file with a minimal amount of items defined, thereby making it easy to create many documents.

Note that document assembly does not pertain to only new documents, either. Of course, by following the rules of Office XML Formats, you can build documents programmatically without using Office. But often document assembly often means building documents by using portions of existing documents, data, and other content. The new Office XML Formats fit well in this scenario because they have a modular architecture and their content is XML-based.

A document assembly example also applies to PowerPoint presentations. Many organizations have vast collections of PowerPoint files that have reusable value. Users often borrow slides from several pre-existing presentations to create an additional or related presentation. Finding, coordinating, and integrating (copying and pasting) slides is typically a time-consuming, redundant process that many organizations want automated for customer-facing presentations. With the 2007 Office release, individual slides within a PowerPoint presentation file are readily accessible because each one is self-contained in its own XML part within the presentation container package. A custom solution can use this architecture to automate the assembly process for presentations completely. You can use custom XML to hold metadata pertaining to individual slides, thus allowing users to search them easily by using predefined keywords. After a user selects a slide, the solution inserts the slide's XML part into the assembled presentation and creates the reference relationship.

Document Security

Security is very important today in information technology. Office XML Formats help you to be more confident about working with Office documents and delivering solutions that consider document security. With the new file formats, you can build solutions that search for and remove any identified potential vulnerabilities before they cause issues.

For example, if a company needs a solution to prepare documents either for storage in an archive library, where they never need to run custom code, or for sending macro-free documents to a customer. You can write an application that removes all VBA code from a body of Office documents by iterating through the documents and removing the part specified by the target of the http://schemas.microsoft.com/office/2006/relationships/vbaProject relationship. The result is a collection of higher-quality documents.

In addition, the Office XML Formats provide one file type only for each product in the 2007 release (.docm for Word, .xlsm for Excel, and .pptm for PowerPoint) in which you can execute macro code. Any file types that do not end with the "m" suffix, even those that do contain macro code, will not execute that code. This helps guarantee that users are safe from malicious software when working with those file types.

Unfortunately, macro code is not the only potential security threat for Office users. Recently, security vulnerabilities were detected in binary .jpg files. You can circumvent potential risks from binary files, such as OLE objects and images, by interrogating Office documents and removing any exposures that arise. For example, if a specific OLE object is identified as a known security threat, you can create an application to locate and cleanse or quarantine any documents containing the object. Likewise, you can readily identify any external references made from a 2007 Microsoft Office system document by examining the relationship parts. This identification enables solution developers to decide if external resources referred to from a document are trustworthy or require corrective action.

You can block parts by content-type and relationships. For example, if an IT administrator becomes aware that .gif images can contain a security threat, group policy settings can be used to disallow the loading of image/gif content in 2007 Microsoft Office system documents. After deploying the proper security patches, the setting can be turned off, and the IT administrator can be confident that users were protected quickly.

You can also block parts through format policies. For example, if the IT administrator discovers a vulnerability in which malformed comments can cause PowerPoint to fail with an exploitable buffer overrun, users and their computers can be protected by the deployment of an Office file format policy. This time, rather than blocking by content type (because in this case, it is an XML containing text), comments are blocked specifically, by blocking the schemas.microsoft.com/office/2006/relationships/comments relationship type. To do this, administrators can use Microsoft Office policy templates (.adm files) to set these options. For more information about using Office policy templates, see the Microsoft Knowledge Base article Administrators Can Use Office Policy Templates with the Group Policy Settings of Windows.

Managing Sensitive Information

As you seek to protect users from malicious content, you can also help protect users from accidentally sharing data inappropriately. This inappropriate data might be in the form of personally identifiable information (PII)

stored within a document, or tracked changes, comments, and annotations so marked that they should not leave the department or organization. You can programmatically remove both types of information directly without having to scour an entire document. To remove document comments, for example, you can check for the existence of a comment part relationship and, if found, remove the associated comment part.

Besides securing PII and comments, Office XML Formats enable access to this information that may be useful in other ways. You could create a solution that uses PII data to return a list of documents authored by an individual person or from a specific organization. With the new file formats, you can produce this list without having to open Office or use its object model. Similarly, an application could loop through a folder or volume of Office documents and aggregate all of the comments within the documents. You can apply additional criteria to qualify the comments and help users better manage the collaboration process as they create documents.

Document Styling

Like so many other aspects of Office documents using Office XML Formats, document styles, formatting, and fonts are maintained in separate XML parts within the container package. So, once again, you can create solutions that take advantage of this separation. Some organizations have very specific document standards, and managing these can be quite time-consuming. However, you can, for example, modify or replace fonts in documents without opening Office.

Also, it is a common practice to have a document or collection of documents that contain the same content, but that were formatted differently by another department, location, subsidiary, or targeted customer. You can maintain the content within a single set of documents, and then apply a new set of styles, as necessary. To do this, you exchange the part specified by the target of the http://schemas.microsoft.com/office/2006/relationships/styleSheet relationship with another part. This ability to exchange simplifies the process of controlling a document's presentation without having to manage content in numerous documents.

Document Profiling

Managing documents effectively has been a long-standing issue in information technology practices. In Microsoft Office 2003, you have access to the traditional Office document properties, such as Author, Title, Subject, and other properties, using OLE. In the new Office XML Formats, document properties are also readily accessible, because they reside in their own part within a document. The following is an example of a Document Properties part in a Word .docx file.

```
 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CoreProperties xmlns="http://schemas.microsoft.com/package/2005/06/md/core-properties">
   <Title>Word Document Sample</Title>
   <Subject>Microsoft Office Word 2007</Subject>
   <Creator>2007 Microsoft Office System User</Creator>
   <Keywords/>
   <Description>2007 Microsoft Office system .docx file</Description>
   <LastModifiedBy>2007 Microsoft Office System User</LastModifiedBy>
   <Revision>2</Revision>
   <DateCreated>2005-05-05T20:01:00Z</DateCreated>
```

```
    <DateModified>2005-05-05T20:02:00Z</DateModified>
</CoreProperties>
```

However, Office documents using the new file formats allow you to add your own data and content beyond what Office-based properties offer, for example, for advanced document profiling. You can create your own custom-defined XML and place it in the file as just another part. You can then use this XML with any tool or application capable of accessing Office XML Formats.

Using the New File Formats in the Office Products

While there are many parts that are common to the Office products that implement the new file formats, there are also components specific to each product.

The File Format Structure in Excel 2007

In addition to the parts that each Office product has in common (such as XML data parts, relationships parts, and media parts), Excel also provides separate parts for a workbook, sheets, and such entities as charts, PivotTable dynamic views, and so forth.

The File Format Structure in PowerPoint 2007

PowerPoint uses many of the same parts as the other products in its file format, with the addition of those objects specific to slide presentation, such as individual slide parts, a master slide part, presentation data, and so forth.

The File Format Structure in Word 2007

Likewise, Word users can expect to see additional parts relating to document properties, styles and formatting, footers, headers, endnotes, and so forth.

Modifying Office XML Formats Files to Customize the Ribbon User Interface

The following steps illustrate creating a custom Ribbon user interface (UI) in Excel 2007 that contains the components to call a custom macro by modifying a macro-enabled workbook file. In this sample, you do the following:

- Create an Excel workbook with one macro and save the workbook as an Office XML Formats macro-enabled file (.xlsm).
- Create a Ribbon extensibility customization file with one tab, one group, and one button.
- Specify a callback event in the button to call the macro you created in the document.
- Modify the contents of the macro-enabled document container file to point to the Ribbon extensibility customization file.
- Save the macro-enabled file and open it in Excel.

To create a macro-enabled Office XML Formats file in Excel

1. Start Excel 2007.
2. Click the **Developer** tab, and then click **Visual Basic**.

> **Note**
>
> If you do not see the **Developer** tab, you need to identify yourself as a developer. To do this, in your application, click the **Microsoft Office Button**, click **Excel Options**, click **Personalize**, and then select **Show Developer tab in the Ribbon**. This is a global setting and identifies you as a developer in all other Office applications that implement the Ribbon UI.

3. In the Visual Basic Editor, double-click **ThisWorkbook** to open the code window.
4. Type the following VBA subroutine and then close the Visual Basic Editor:

```
 Sub MyButtonMacro(ByVal ControlID As IRibbonControl)
   Msgbox("Hello world")
End Sub
```

5. Save the workbook as an Office XML Formats macro-enabled file (.xlsm).

To create the XML file that contains the markup to modify the UI

1. Create a folder on your desktop called *customUI*.
2. Open a new file in a text editor and save it as *customUI.xml* in the *customUI* folder.
3. Add the following code to the file:

```
<customUI xmlns="http://schemas.microsoft.com/office/2006/01/customui">
  <ribbon startFromScratch="true">
     <tabs>
        <tab id=":CustomTab" label="My Tab">
           <group id="SimpleControls" label="Sample Group">
              <button id="Button1" size="large" label="Large Button"
                 onAction="ThisWorkbook.MyButtonMacro" />
           </group>
        </tab>
     </tabs>
  </ribbon>
</customUI>
```

Next, you modify some of the files contained in the macro-enabled file container that you created previously:

1. Add a .zip extension to the workbook file name and double-click to open the file
2. Add the customization file to the container by dragging the **customUI** folder from the desktop to the ZIP file.
3. Extract the .rels file to your desktop. A _rels folder containing the .rels file is copied to your desktop.
4. Open the .rels file and add the following line between the last **Relationship** tag and the **Relationships** tag. This creates a relationship between the workbook file and the customization file:

```
 <Relationship Id="someID" Type="http://schemas.microsoft.com/office/2006/relationships/ui
/extensibility" Target="customUI/customUI.xml" />
```

5. Close and save the file.
6. Add the _rels folder back to the container file by dragging it from the desktop, overwriting the existing file.
7. Rename the workbook file to its original name by removing the .zip extension.
8. Open the workbook and notice that the Ribbon UI now displays **My Tab**.
9. Click the button and the message box is displayed.

Conclusion

Users, organizations, and developers benefit from the advantages of the Office XML Formats in the 2007 release of the Microsoft Office system. As open, default file formats based on XML, the new file formats unlock the possibilities for many new solution types and scenarios that you can build. You can access documents as sources of data, manipulate them without the Office applications, and process them in enterprise solutions. Organizations that combine existing business system investments with the Microsoft Office system platform, the 2007 Office release, and the new XML-based file formats can only benefit.

Additional Resources

For more information, see the following resources:

- Open XML Formats Resource Center
- Ecma Office Open XML File Formats overview
- Microsoft Office Developer Center
