



madcap
BLAZETM

Key Features Guide

Version 3.0

Copyrights

Copyright 2010 MadCap Software. All rights reserved.

Information in this document is subject to change without notice. The software described in this document is furnished under a license agreement or nondisclosure agreement. The software may be used or copied only in accordance with the terms of those agreements. No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or any means electronic or mechanical, including photocopying and recording for any purpose other than the purchaser's personal use without the written permission of MadCap Software.

MadCap Software
7777 Fay Avenue
La Jolla, California 92037
858-320-0387
www.madcapsoftware.com

Contents

Key Features	1
General.....	1
Interface.....	2
XML Editor.....	3
Single-Source Publishing.....	5
Language Support.....	7
Page Layouts.....	8
Importing.....	9
Output.....	11
Project Management.....	13
Other Key Features.....	14
Index	15

Key Features

Following are some of the key features that make Blaze unique. See the online Help for more information about each feature.

General

Following are some general key features in Blaze:

- **XML authoring** Even if you do not know anything about Extensible Markup Language (XML), you can use Blaze to create XML content (in Blaze's main content editor, the XML Editor). If you are experienced with XML, you will be able to take advantage of all of the benefits of this structured language in Blaze.
- **XML everywhere = open file architecture** Topics are not the only Blaze documents that use XML. All Blaze files are separate XML documents—topics, outlines, targets, snippets, glossaries, destinations, condition tag sets, variable sets, and more. This means that Blaze projects are completely open, transparent, and accessible.
- **W3C schema compliance** Blaze content conforms to industry standard schema requirements from the World Wide Web Consortium (W3C). Therefore, Blaze content can be edited in the Blaze XML Editor, in external editors, or transferred back and forth between the two at will. Blaze's code adheres to the W3C XHTML Schema specification, making it easy to integrate with other XML or XHTML applications.
- **Multiple topics versus one long document** Topics are individual chunks of content about a particular subject. In Blaze, you can create long topics (i.e., each topic might be one chapter in the output), or you can create many small topics that can be combined in the output to form the chapters. In the output, topics are strung together in the order that you specify in an outline. Although you can use either long or short topics in your project, creating small topics is highly recommended. Small topics are easier to manage and send out to others for review. In addition, small topics let you take full advantage of Blaze's many single-sourcing features (e.g., you can easily pick and choose which topics to include in a particular manual).

Interface

Following are some key features of the Blaze user interface:

- **Multiple documents open simultaneously** Blaze is unique in that you can have multiple documents open in the interface simultaneously.

EXAMPLE

If you want to work on four topics, two outlines, and one style sheet all at the same time, you do not have to close one document before working on another. You can have all seven documents open at the same time. You can even "float" and move them so that they are placed side-by-side. If you have access to dual monitors, you can have one document displayed on one monitor and another document displayed on the second monitor.

- **Floating and dockable window panes** You can "float" window panes and editors, which lets you then click and drag them wherever you want in the interface. A window pane can be attached (or "docked") to the sides, top, or bottom of the program interface.
- **Resizable dialogs and drop-down menus** All dialogs and even drop-down menus can be resized by clicking and dragging their edges.
- **Window layouts** You can save a custom configuration (layout) of your workspace.

EXAMPLE

You might want to use Blaze's default layout most of the time. However, when you are working on creating an index, you want to open the Index window pane. Furthermore, you may want to move that window pane to a certain place in the workspace. After you have this element just where you want it, you can then save that configuration of the workspace as a layout so that you do not have to move things around each time you want to work on indexing.

- **Enhanced undo and redo** Blaze's undo and redo functions are enhanced as follows:
 - **Unlimited** You can undo or redo an unlimited number of times, back to the last time you opened the document, even after you have saved it.
 - **Multiple buffers** Each document in the Blaze interface has its own Undo/Redo buffer. This means that you can undo actions in one document and then switch to another document and undo actions specific to that file.
 - **User-friendly interface** The name of the action is displayed on the Undo or Redo button. In addition, you can use the drop-down arrow next to the button to select multiple steps to undo or redo at once, even resizing the drop-down box as necessary.

XML Editor

The XML Editor is the primary editor that is used in Blaze. It provides access to the underlying XML structure of documents in a comfortable visual authoring environment. This editor is used to enter, modify, and format the content for topics that users see in the output. Not only is this editor used for topics, but it is also used for working with snippets. Although this editor lets you produce XML files, you do not need to know anything about XML to use it. Following are some of the key features of this editor:

- **Print and Web layout modes** The XML Editor lets you work in two different layout modes—Web or Print. The Web mode is the traditional layout for online use. The Print mode displays topics integrated with page layouts that you can create. This allows you to see the margins, headers, footers, etc. that will be used in the print output.
- **Structure bars** The XML Editor in Blaze contains four kinds of "structure bars," which (as the name suggests) are bars around the topic content that show its structure. Not only do structure bars let you see the tags and spans in a topic, but you can also perform numerous tasks by clicking on the individual bars and making a selection from the context menu. These bars might seem unusual when you first start working in Blaze (especially if you are used to working in another tool), but you will soon find that they can be extremely useful and an integral part of creating and designing content.
- **Markers** Blaze uses industry-driven markup in content. A marker is a highlighted "flag" that shows the insertion of certain features in the XML Editor—such as variables, bookmarks, index keywords, and concepts. A major benefit of using markers is that they are deleted when you remove the content containing them. This means that orphan keywords are not left in the project without a point of reference.

EXAMPLE

Let's say you insert an index keyword into a topic. Later, if you decide to delete that topic, the index keyword is also deleted. If markup was not used in the application, but rather a separate database was used to hold all index keywords, you might forget to remove the index keyword after you delete the topic. Then, when a user clicks the index keyword in the output, the application will be unable to find the topic.

- **Add notes and comments in lists** The "Make Paragraph Item" feature lets you add a paragraph tag at the end of a numbered or bulleted list item. This allows you to easily add a comment after the item without interrupting the flow of the list. To use this feature, simply click on the list item, select **Format>List>List Actions>Make Paragraph Item(s)**, and then press **Enter**.
- **Merge lists** There may be times when you are creating lists and find that you have two or more ordered lists (tags) or unordered lists (tags) next to each other. If necessary, you can quickly merge the lists together into one list.

- **Drag and drop table rows and columns** The XML Editor allows you to drag and drop content as necessary. One unique aspect to this is the ability to click individual rows or columns in a table and drag them to new locations in the table.
- **Color selection tool** When you need to apply color to content, you can easily select and customize colors by using Blaze's Color Picker dialog. You can even use the "Screen" tool to hover over any area of your computer screen and click to select the precise color beneath it.
- **HTML to XHTML conversion** When you attempt to open an HTML file in the XML Editor, a dialog opens, which guides you through the process of converting that file to XHTML so that it can be modified in the XML Editor. You can also select **Project>Import HTML Files** to import and convert multiple HTML files to XHTML at once.
- **Tag-sensitive cursor** In Blaze's XML Editor, the cursor knows when it is inside or outside of a tag. By default, the cursor is a bracket when it is located between tags (e.g., inline formatting). This bracket cursor faces right (]) or left ([), depending on whether it is inside a particular tag.

E X A M P L E

Let's say you have written a sentence, and at the end of the sentence you have typed the word "Close" in bold font, followed by a period in regular font. Now you want to add the word "button" immediately after the word "Close," but you do not want the word "button" to take on the bold font. You hover the cursor at the end of the word "Close." The bracket faces left ([), which tells you that if you were to begin typing at that location, the next text would also be bold. So you move the cursor slightly toward the right and the bracket cursor faces right (]), which tells you that the next text at this location would take on the regular font of the period. You click the mouse button and type the new text in regular font.

Single-Source Publishing

"Single-sourcing" is a fancy term that means something very simple—to produce multiple results from one source. In Blaze, you can make use of single-sourcing in many different ways.

- **Global Project Linking** You can import content and project files contained in another Blaze project, thus allowing you to maintain the information in one location but reuse it in any other project. When you use this feature to import files, you can include or exclude particular types of files (e.g., topics, snippets, style sheets, glossaries, targets), specific individual files, or files that have certain condition tags applied. Simply use the include/exclude methods that work best for you.

This is different than a simple import process, because in this case, the imported files remain linked to the source project. This allows you to make future updates to those files in just one place—in the source project file. When you perform ongoing imports using your previous settings, Blaze recognizes changes to the source files. Therefore, the new files can be brought over, replacing the outdated files.

EXAMPLE

Let's say you are working on three different Blaze projects. Within those projects, you might have 35 topics and 50 images that are identical in the three projects. In addition, you might use the same style sheet in each project. Rather than maintaining three different sets of identical files, you can store one set of those files and import them into the individual projects when needed. Here are a couple of options: (1) One option is that you could consider one of your three Blaze projects as the "global parent" for those shared files. (2) Another option is that you could create a new Blaze project (perhaps naming it "global"); this project could have no other purpose than to serve as a repository for the shared files across your projects. In other words, you would not necessarily generate any output from this parent project, but simply use it as a place to hold your shared information.

When you want to use any of the shared topic, image, or style sheet files from the global project, you would import them into the child project. This creates a link between the imported files and those in the global project. Therefore, when you edit those files in the future, you would do so from the global project and then re-import the changes (either manually or automatically) to the other child projects.

- **Multiple outputs from one project** Blaze allows you to generate output in a variety of formats, creating as many different targets as you want from the same Blaze project.
- **Condition tags at all levels** You can apply condition tags at all levels in Blaze—character, paragraph, file, and more.

- **Snippets** In Blaze, there are multiple ways to reuse content in different places. One method is to create a snippet. Snippets are pre-set chunks of content that you can use in your project over and over. Snippets are used for longer pieces of content that you can format just as you would any other content in a topic. In snippets, you can also insert tables, pictures, and whatever else can be included in a normal topic.
- **Snippet conditions** Snippet conditions are condition tags that you can apply to content within snippets. With snippet conditions, you can separate certain snippet content so that it displays in some topics but not in others. This allows you to use one snippet for many purposes, rather than having to create multiple snippets. Whereas regular conditions are included or excluded at the target level, snippet conditions are included or excluded at the topic level.
- **Variables** Another way to reuse content is to create variables. Variables are pre-set terms that you can use in your project over and over. They are stored in "variable sets," which can hold multiple variables. Blaze provides you with an initial variable set, but you can add as many additional variable sets as you like. Variables are used for brief, non-formatted pieces of content (such as the name of your company's product or your company's phone number). There are different kinds of variables: (1) those you create, (2) system variables (e.g., date and time; Chapter, Section, and Volume numbers), (3) Heading variables, and (4) Running Head variables. Some of these are especially useful for page headers and footers.
- **Style sheets** You can take advantage of cascading style sheets (CSS files) to control the look of your output in one place. You can apply style sheets to individual topics, or you can use a "master" style sheet, applying it to all files at the target level or project level. In addition to using style sheets for topics, you can use separate style sheets in Blaze specifically for tables inserted into topics.
- **Mediums for topic styles** Let's say you want one style setting (e.g., underline font) to be used for a particular output and another setting (e.g., do not underline font) to be used for a different output. You can use a medium in your style sheet to create different settings for the same style. When you apply a particular medium to a target, it will be used for that output.

Language Support

Following are some of the ways that you can take advantage of language support in Blaze:

- **Unicode support for all left-right languages** Blaze is fully Unicode capable, making it possible to handle the entire world's Unicode language characters. Blaze not only supports Western European languages, but also double-byte Asian languages, Eastern European languages, and more.
- **MadCap Lingo integration** One of the easiest ways to translate a Blaze project is for a translator to open that project within MadCap Lingo, which is tightly integrated with Blaze. Because of this integration, there is no need to transfer localized files outside of the actual project, which helps prevent content and formatting corruption. In addition, translators can leverage all previous translations created in other tools by importing Translation Memory eXchange (TMX) files.

After opening your project in Lingo, a translator can immediately see a list of all of the files (e.g., topics, snippets, variables), index markers, and concepts that need to be localized. Then, after translating the content in the Lingo interface, the translator can export the results to a new Blaze project in that language. For more information, please refer to the documentation provided with MadCap Lingo.

Page Layouts

A page layout is an element that you can create in your project in order to determine page specifications (e.g., size, margins) and to apply certain content (e.g., headers, footers, page numbers) to many (or all) topics in print-based output. Page layouts allow for easy configuration through the use of content frames, a snap-to grid, dragging and dropping, alignment features, and more.

EXAMPLE

Let's say you are creating a manual that consists of front matter (e.g., title page, copyright page, and table of contents), 10 chapters, and an index. Perhaps you want all of the pages in the manual to measure 8 inches in height and 6 inches in width. Furthermore, you might want some pages (e.g., title and copyright pages) to contain no headers or footers, while you want the other parts of the manual to contain header text and page numbers at the bottom. In a situation such as this, you might create one page layout for your title and copyright pages, a second page layout for your TOC, a third page layout to be used by all of the chapters, and a fourth page layout to be used by the index. Each page layout might contain the same page size settings, but different page headers and footers.

Like all other files in Blaze, a page layout is an XML file. It has an .flpgl extension and is stored in the Content Explorer under the Resources\PageLayouts folder.

Importing

Following are some of the ways that you can import files into Blaze projects:

- **Import legacy documentation** Blaze allows you to import existing documentation in the following formats:
 - **Word documents** You can import Microsoft Word documents, including DOC, DOCX, and RTF files. Blaze tightly integrates with Word (including Word 2007), using modern XML data flow techniques and leveraging the Microsoft XML Schema for Office documents. This allows for superior content fidelity during import.
 - **FrameMaker documents** You can import Adobe FrameMaker documents, including BOOK, FM, or MIF files. Because you can import the source FrameMaker BOOK and FM files (rather than just MIF files), Blaze has full access to FrameMaker variables, conditionals, auto-numbering, and so on. This means that those features are converted to Blaze seamlessly.
 - **DITA file content** In Blaze, you can import content from files that have either a .dita or .ditamap extension. You can start a new project by importing DITA file content, or you can import DITA file content into an existing project. When DITA file content is imported, the DITA elements are converted to their equivalents in the Blaze XHTML environment. For example, formatting elements are converted to style classes.
 - **HTML files** You can import HTML files and convert them to XHTML by using the **Project>Import HTML Files** option.
 - **XHTML files** You can import XHTML files.
- **Import projects from source control** If you are using a source control application that is integrated with Blaze, you can import an existing Blaze project from the source control tool. You might use this method, for example, if you are working on a multi-author project and another member of the team has placed the Blaze project in a source control tool such as Microsoft Visual SourceSafe or Microsoft Team Foundation Server.
- **Easy Sync** If you import files from another project, copies of those files are placed in the current project and a link exists between the imported files and the source files in the parent project. Also, if you import Word, FrameMaker, or DITA file content into Blaze, you can specify whether a similar link ties those imported files to the source Word, FrameMaker, or DITA files. These links mean that future changes to the imported files can be made at the source (i.e., in your parent project, or in the source Word, FrameMaker, or DITA files). When you make future changes to the source documents, those files can be re-imported into the project so that they are included in the current project's output. You have the option of re-importing these files manually. However, you can also tell Blaze to do this for you automatically when you attempt to generate the output. This is known as "Easy Sync."

The Easy Sync option is labeled **Auto reimport before 'Generate Output'** and is located in the Project Import Editor (for Global Project Linking); Import Microsoft Word Wizard and Word Import Editor (for Word imports); Import FrameMaker Wizard and Frame Import

Editor (for FrameMaker imports); and Import DITA Wizard and DITA Import Editor (for DITA imports).

- **Full style mapping** Blaze has full style mapping capabilities. For example, if you import six Word documents, you don't have to worry about six different CSS files being created.

Output

Following are the output options available in Blaze.

- **PDF** Short for "Portable Document Format," PDF is an open file format created by Adobe. PDF files represent two-dimensional documents in a device-independent and resolution-independent fixed-layout document format.
- **XPS** Microsoft's XML Paper Specification (XPS) is a document format with a markup language that is a subset of XAML for Windows Presentation Foundation. XPS is an alternative to Adobe's Portable Document Format (PDF).
- **XHTML book** XHTML is a browser-based output type that consolidates project content in an XML file. It can be viewed online or printed. The output appears as one long book, even if the project consists of hundreds of topics. You can view and print the XHTML output.
- **Microsoft Word** The output is exported to Microsoft Word in one of the following file formats.
 - **XML** This is the default document format created (if you do not select one of the other formats in this list).
 - **DOC** You can export Word to the standard DOC format.
 - **DOCX** This is Microsoft Word's platform-independent, open XML format.
 - **XPS** In addition to sending output directly to XPS (as described above), you can generate an XPS file automatically when building Word output. You can do this by installing a free add-in download from Microsoft.
 - **PDF** In addition to sending output directly to PDF (as described above), you can generate a PDF file automatically when building Word output. Because Blaze supports Microsoft Vista and Word 2007, you can send Word output to PDF format, even if you do not have the Adobe Distiller installed.
- **Adobe FrameMaker** The output is exported to Adobe FrameMaker in one of the following formats.
 - **BOOK** This is FrameMaker's native book (or "document collection") file.
 - **FM** This is FrameMaker's native single document file.
 - **PDF** In addition to sending output directly to PDF (as described above), you can generate a PDF file automatically when building FrameMaker output.
- **DITA** Darwin Information Typing Architecture (DITA) file content is supported in Blaze. DITA is an XML-based markup language with its own schema for authoring, producing, and delivering technical information. It is a standard of the Organization for the Advancement of Structured Information Standards (OASIS), and it consists of a set of design principles for creating "information-typed" modules at a topic level and for using that content in various delivery modes. In Blaze, you can generate output that produces DITA files. When you build this type of output, a DITA map file is generated, with multiple DITA files in it. The XHTML tags are converted to DITA elements. In other words, although it is considered an "output" from

the standpoint of the Blaze process, the end result is actually a collection of "source" files, which you can later use in another tool (or import back into Blaze) to produce the final output.

Online output via Flare: Blaze is designed solely for print output. However, because Blaze uses the same files and structure as MadCap Flare, you can easily open a Blaze project in Flare and take advantage of all of its features to produce online output. To do this, simply change the extension of the main Blaze project file from .blprj to .flprj.

Project Management

Blaze provides the following features, which can be used to manage your project and enhance team authoring:

- **File tagging** You can assign "tags" to topics and any other files in Blaze, even folders. You can use file tags for many different purposes, such as assigning authors or milestones to topics. Blaze lets you generate reports based on the tags that are assigned. This makes project development easier to track, manage, and schedule.
- **Source control** Because all content and project-level files are stored as separate XML files, Blaze projects are compatible with all source control systems. All files in a project are independent of one another, which means that there are no file dependencies that hinder multiple authors from accessing project files.

Blaze also provides integrated support for version control applications. Built-in support is available for Microsoft SourceSafe and Microsoft Team Foundation Server. In addition, the Microsoft Source Code Control API (SCC API) allows you to configure your project for integration with other version control tools. One of the highlights of this integration is an instant message/email system, which can be used to quickly send requests to other authors to check in needed files. This same system allows individuals to check in and check out files on the fly, saving valuable time for multi-author teams.

- **Topic reviews and contributions** Blaze provides for close collaboration between authors and others through the use of topic reviews and contributions.
 - **Reviews** You can quickly email topics to other individuals for their feedback and/or changes. After inserting annotations (comments) or making changes to a topic with MadCap X-Edit or X-Edit Review, reviewers can send the topic back to you. You can then accept those comments or changes so that they are added to the source file.
 - **Contributions** Authors, developers, or other individuals in your company can use MadCap X-Edit or X-Edit Contribute to create new documents and files, which can be incorporated into your Blaze project.

Other Key Features

Following are some additional key features of Blaze:

- **Auto-numbering** This is just what it sounds like—a feature where content is numbered automatically. Of course, if you want to create simple numbered lists in topics, you can always use Blaze's quick list drop-down options. But if you want an alternative that is more advanced and powerful, you can use auto-numbering.
- **Cross-references** A cross-reference is a navigation link that lets you connect text in one topic to another topic (or a bookmark within a topic). This is somewhat similar to a text hyperlink. However, cross-references are more powerful in that the links can automatically be updated based on commands. You can also convert cross-references to elements such as page numbers for printed output.

You can also take advantage of context-sensitive cross-references, which are especially designed for generating print-based output. When you use a context-sensitive cross-reference, the text automatically changes based on the relationship of the link and the target location if they are on the same page or only one page away.

EXAMPLE

Let's say you have a cross-reference designed to display the text "See Figure 2.1." If the link and the target fall on the same page, the cross-reference is updated to display the text, "See Figure 2.1 above" or "See Figure 2.1 below." If the link and the target are on adjacent pages, the cross-reference is updated to display the text, "See Figure 2.1 on previous page" or "See Figure 2.1 on next page." If the document is double-sided with the link on the left page and the target on the right page, the cross-reference displays the text, "See Figure 2.1 on the facing page."

- **Link Viewer** In Blaze, there are many ways that you can "link" one file with another, thus creating a dependency. Link dependencies are created when you perform tasks such as: inserting a text hyperlink, inserting a picture, applying a style sheet to a topic, and more. The Link Viewer window pane lets you see what other files a particular file is linked to and from.

Index

A	
Adobe FrameMaker	
importing	9
output	11
Adobe PDF	11
Auto-numbers	14
B	
BOOK file	9
Bulleted lists	See <i>Lists</i>
C	
Cascading style sheets	6
Chapters	
variables	6
Classes	See <i>Styles</i>
Color	
screen	4
Comments	
lists	3
Condition tags	
content	5
snippets	6
Cross-references	14
CSS file	6, 10
Cursors	
tag-sensitive	4
D	
DITA	
DITA file	9
DITAMAP file	9
importing	9
output	11
DOC file	9, 11
Dockable windows	2
DOCX file	9, 11
E	
Easy Sync	9
Extensible Markup Language	See <i>XML</i>
F	
Files	
tags	13
Floating	
windows	2
FLPGL file	8
FM file	9
Footers	8
Chapter Number variables	6
Heading variables	6
Running Head variables	6
Section Number variables	6
Volume Number variables	6
FrameMaker	See <i>Adobe FrameMaker</i>
G	
Global Project Linking	5
H	
Headers	8
Chapter Number variables	6
Heading variables	6
Running Head variables	6

"Heading" through "Printed output"

Section Number variables	6	Microsoft Visual SourceSafe	9
Volume Number variables	6	Microsoft Word	
Heading		importing	9
variables	6	output	11
HTML file	4	Microsoft XPS	11
importing	9	MIF file	9
I		Moving	
Importing	9	table columns	4
DITA files	9	table rows	4
files from projects	5	Multi-authoring	9, 13
FrameMaker documents	9	N	
HTML files	9	Navigation links	
source control projects	9	cross-references	14
topics	9	Numbered lists	See Lists
Word documents	9	O	
XHTML files	9	Output	
L		DITA	11
Language support		FrameMaker	11
Unicode	7	Microsoft Word	11
Layouts	See Page layouts	PDF	11
Links		types	5
viewing	14	XHTML	11
Lists		XPS	11
adding notes or comments	3	P	
merging	3	Page footers	8
Localization	See Language support	Page headers	8
M		Page layouts	
MadCap X-Edit	13	Chapter Number variables	6
Mapping		Heading variables	6
styles	10	Running Head variables	6
Markers		Section Number variables	6
tags	3	Volume Number variables	6
Master pages	See Page layouts	Page numbering	8
Mediums	6	Paragraphs	
Merging		lists	3
lists	3	Printed output	
Microsoft Team Foundation Server	9	DOC	11

DOCX	11	Team Foundation Server	See <i>Microsoft Team Foundation Server</i>
XML	11		
Projects		Topics	
files	5	snippet conditions	6
Global Project Linking	5		
importing	5		
R		U	
Redo	2	Undo	2
RTF file	9	Unicode	7
Running Head variables	6	V	
S		Variables	6
Schemas		Chapter Number variables	6
DITA	11	Heading variables	6
W3C	1	Running Head variables	6
Sections		Section Number variables	6
variables	6	Volume Number variables	6
Selectors	See <i>Styles</i>	Version control	See <i>Source control</i>
Single-sourcing	5	Viewing	
Snippets	6	links	14
condition tags	6	Visual SourceSafe	See <i>Microsoft Visual SourceSafe</i>
Source control	13	Volumes	
importing projects	9	variables	6
Microsoft Team Foundation Server	9	W	
Microsoft Visual SourceSafe	9	W3C	1
SourceSafe	See <i>Microsoft Visual SourceSafe</i>	Windows	
Structure bars	3	layouts	2
Styles	6	Word	See <i>Microsoft Word</i>
mapping	10	WYSIWYG Editor	See <i>XML Editor</i>
mediums	6	X	
style sheets	6	X-Edit	See <i>MadCap X-Edit</i>
T		XHTML	
Tables		file	1, 4, 9
columns	4	output	11
rows	4	XML	1
Tags		file	11
files	13	XML Editor	1

"XPS" through "XPS"

XPS

See *Microsoft XPS*