BS EN 62524:2011

BSI Standards Publication

Multimedia systems and equipment — Multimedia e-publishing and e-books — Reader's format for e-publishing

...making excellence a habit.™

National foreword

This British Standard is the UK implementation of EN 62524:2011. It is identical to IEC 62524:2009.

The UK participation in its preparation was entrusted to Technical Committee EPL/100, Audio, video and multimedia systems and equipment.

A list of organizations represented on this committee can be obtained on request to its secretary.

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

ISBN 978 0 580 60766 0

ICS 33.160.99; 35.240.30

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 31 March 2011.

Amendments issued since publication

Amd. No.	Date	Text affected

EUROPEAN STANDARD

NORME EUROPÉENNE

EUROPÄISCHE NORM

EN 62524

February 2011

ICS 33.160.99

English version

Multimedia systems and equipment - Multimedia e-publishing and e-books - Reader's format for e-publishing
(IEC 62524:2009)

Systèmes et appareils multimedia – Edition électronique multimedia et livres électroniques – Format du lecteur pour édition électronique
(CEI 62524:2009)

Multimediasysteme und -geräte - E-Publishing und E-Books für Multimedia-Anwendungen - Reader-Format für E-Publishing
(IEC 62524:2009)

This European Standard was approved by CENELEC on 2011-01-02. CENELEC members are bound to comply with the CEN/CENELEC Internal Regulations which stipulate the conditions for giving this European Standard the status of a national standard without any alteration.

Up-to-date lists and bibliographical references concerning such national standards may be obtained on application to the Central Secretariat or to any CENELEC member.

This European Standard exists in three official versions (English, French, German). A version in any other language made by translation under the responsibility of a CENELEC member into its own language and notified to the Central Secretariat has the same status as the official versions.

CENELEC members are the national electrotechnical committees of Austria, Belgium, Bulgaria, Croatia, Cyprus, the Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, the Netherlands, Norway, Poland, Portugal, Romania, Slovakia, Slovenia, Spain, Sweden, Switzerland and the United Kingdom.

CENELEC

European Committee for Electrotechnical Standardization
Comité Européen de Normalisation Electrotechnique
Europäisches Komitee für Elektrotechnische Normung

Management Centre: Avenue Marnix 17, B - 1000 Brussels

Foreword

The text of document 100/1376/CDV, future edition 1 of IEC 62524, prepared by technical area 10, Multimedia e-publishing and e-books, of IEC TC 100, Audio, video and multimedia systems and equipment, was submitted to the IEC-CENELEC parallel vote and was approved by CENELEC as EN 62524 on 2011-01-02.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. CEN and CENELEC shall not be held responsible for identifying any or all such patent rights.

The following dates were fixed:

– latest date by which the EN has to be implemented at national level by publication of an identical national standard or by endorsement (dop) 2011-10-02

– latest date by which the national standards conflicting with the EN have to be withdrawn (dow) 2014-01-02

Annex ZA has been added by CENELEC.

Endorsement notice

The text of the International Standard IEC 62524:2009 was approved by CENELEC as a European Standard without any modification.

Annex ZA
(normative)

Normative references to international publications with their corresponding European publications

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

NOTE When an international publication has been modified by common modifications, indicated by (mod), the relevant EN/HD applies.

Publication	Year	Title	EN/HD	Year
IEC/TS 62229	2006	Multimedia systems and equipment - Multimedia e-publishing and e-book - Conceptual model for multimedia e-publishing	-	-

CONTENTS

INTRODUCTION

Markets for multimedia e-book and e-publishing require standardization of formats for e-book data interchange among related parties; authors, data preparers, publishers and readers. The formats are classified into submission format, generic format and reader's format. The submission format has to support an interaction between authors and data preparers. The generic format has to provide an interchange format for data preparers and publishers and therefore should be reading-device-independent. The reader's format depends on e-publishing equipment.

MULTIMEDIA SYSTEMS AND EQUIPMENT – MULTIMEDIA E-PUBLISHING AND E-BOOKS – READER'S FORMAT FOR E-PUBLISHING

1 Scope

This International Standard specifies a reader's format for multimedia e-publishing employed for e-book data interchange among publishers and readers, satisfying a number of readers' requirements such as being non-revisable, equipment-adaptive and application-adaptive.

NOTE This International Standard does not address the following issues:

- elements necessary for final print reproduction only;
- rendering issues related to physical devices;
- metadata issues for document management;
- security issues such as DRM for document.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IEC/TS 62229:2006, *Multimedia systems and equipment – Multimedia e-publishing and e-book – Conceptual model for multimedia e-publishing*

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1
data preparer
organization or person that prepares an e-book

NOTE An editor is an example of preparer.

3.2
multimedia e-book
multimedia content consisting of text, graphics, sound and/or video data

3.3
publisher
organization or person that issues and distributes an e-book

3.4
reader
the final user who reads the e-book

3.5
reader's format
format for multimedia e-book contents rendered and presented by reading device

3.6
reading device
equipment or program used to render and display e-books

4 Position and requirements for reader’s format

4.1 Reader’s format in contents creation/distribution model

The conceptual model for multimedia e-publishing (IEC 62229) defines a contents creation/distribution model shown in Figure 1.

Author <--(1)--> **Data preparer** <--(2)--> **Publisher** --(3)--> **Reader**

IEC 190/09

Figure 1 – Contents creation/distribution model

In the third step of the distribution chain, the publisher creates the final version of the document in the reader’s format, based on data obtained from the generic format in the previous step, and finally distributes it to the reader. There can be several distinct reader’s formats, adapted to the various reading devices and distribution schemes. Because of reading devices’ potential limitations, some reader’s format may lack support for certain features of the corresponding generic format. It is the publisher’s role to determine how to convert an e-book to a less capable format.

4.2 Requirements for reader’s format

The reader’s format can satisfy the following requirements of readers:

a) non-revisable

As the reader’s format is the final form of the document, which will be used only for viewing, there is little point in it being revisable. What matters most is that the file should be easy to process, even if this makes editing the data more difficult. Being revisable can even be considered a problem, since it makes the format needlessly complex.

b) equipment-adaptive, application-adaptive

The reader’s format is directly processed while the reader browses the book. For that reason, to maximize the reader’s comfort, the format should be specifically designed to match the capabilities of the device, in terms of CPU power, memory foot-print, display size, etc. For example, a format targeted at a device with weak processing abilities should, in order to keep the memory and CPU requirements low: (1) Use a light special purpose binary structure, rather than processing-intensive formats like XML, (2) store the pre-calculated position of the elements, rather than compute the layout on the fly... On the other hand, if the target reading device is a high end processing system like a PC, a format allowing for rich multimedia effects would be preferred, since it can easily be handled.

c) legibility

To achieve a sufficient level of reading comfort, it is important that the reader’s format pays attention to legibility on the reading device. In that regard, the following implementation methods can be considered.

- Fixed page layout

  The format defines the document so that each page of the document may be rendered identically on any reading device. In this type of layout, it is common to record directly in the file the actual position of all displayable elements. Generally, as the publisher can specify precisely the final aspect, he will set the layout that is deemed to be the most comfortable for the reader. Consequently, complex designs can be achieved, as long as the display is as large as, or maybe larger, than the designer expected, making it possible to reach excellent levels of legibility. On the other hand, if the actual display is smaller than the one the document was designed for, it must be zoomed out, possibly making some characters unreadable, or the reader has to scroll around the document, reducing the reading comfort.

- Flowing layout

  In this type of layout, the reading device dynamically determines, according to the screen size, where line breaks should be inserted, and compute the resulting position of the various elements. The final layout therefore depends on the screen size, font type, font size, etc. The reading device usually has a set of rules to handle word wrapping or hyphenation (ends of lines). Even though this layout model cannot achieve designs as sophisticated as fixed page layouts, it can guarantee that the text will remain clearly readable, whatever the screen size. It also usually gives more customization options to the reader, letting him set parameters as the font size or colour, making it more easily adaptable to individual readers' preferences. On the other hand, the publisher somewhat loses control over the final appearance of the document.

- Others

  Intermediate solutions also exist. For example, a format could specify what is essentially a fixed layout when the screen is large enough to display the defined layout at the current zoom level, but also allow, when zooming in, to change the layout (by collapsing the margins or changing the paragraphs size, for example), so that the characters can indeed be zoomed in, without making the page larger than the screen.

4.3 File format

A reader's format may have a specific data structure depending on the reading device. When rendering functionality is supported by reading devices, both logical structure and style specification are recommended for flexibility of presentation. When no rendering functionality is supported by reading devices, the reader's format should have a final form structure.

The format may also be adapted to the mode of distribution.

- Complete single download: A whole e-book is copied or downloaded at a time to the reading device. In that case, the e-book can usually be stored in a single file.
- Continuous download: Chunks are downloaded on demand, during the rendering. This is useful for device with readily available connectivity, but limited storage capacity. This may be achieved by splitting the document in several small files.

4.4 Features of the reader's Format

4.4.1 General

The features of the reader's format may vastly vary with the targeted reading devices, depending on their capabilities. For this reason, while creating contents for a particular reading device, the publisher may have to omit some features, or, on the contrary, add others to compensate.

4.4.2 Types of displayable elements

Actual rendering capacities of the rendering device may vary, but reader's formats should at least support text and static images. In addition, animations made of a sequence of static images, sound, movies and other multimedia data may be supported too.

4.4.3 Layout and styling

Reader's format, as the final document, shall contain all the styling information needed for proper screen rendering. As stated in 4.2 c), there are mainly two types of layout: the fixed page layout, and the flowing layout. In each case, the way to specify the style may be quite different. When opting for a fixed page layout, the most common solution is to store the final position and style of each displayable element. On the other hand, formats with a flowing layout will have settings such as line spacing, character spacing, indentation and margins, which will be used for computing the layout on the fly by the reading device.

4.4.4 Fonts

In most formats, it is possible to set various text properties, such as font, size, colour, bold or italic, etc.

Moreover, in the case of fixed layout, to ensure that the page is rendered as intended, the font itself may be embedded in the document. On the other hand, in the case of flowing layout, while it is common to use the fonts bundled with the system, font data can sometimes be embedded in the document to ensure that all the needed glyphs will be available at read-time.

4.4.5 E-book specific features

- Link jump

  Allows to jump to a predefine position within the document or to a web site, upon clicking on, or otherwise activating a certain part of the document.

- Effects

  Specifies special visual effects for certain parts of the document, like fade-in or wipe.

5 Conformance level

Generally, reader's formats are designed to closely match the reproduction capabilities of the reading devices. As reading devices can be quite diverse in reproduction capabilities, it is to be expected that the capabilities of each reader's formats differ accordingly. To ease communication and understanding between the various actors of the e-book publishing market, this International Standard establishes a 3-level classification, namely, minimum, medium and rich conformance levels, to help categorize the different reader's formats.

This International Standard requires that each format define such 3-tier classification in itself.

While this standard stipulates a very rudimentary capability for minimum conformance level, other details of the 3 levels are left to the description of each format to allow for diversity among different formats.

a) Minimum conformance level

   Targeted at devices with low reproduction capabilities.

   This standard requires that at least a line of text data be displayed for a viewer to satisfy minimum conformance level.

b) Medium conformance level

   Intermediate level between minimum and rich conformance levels.

c) Rich conformance level

   Targeted at devices with high reproduction capabilities.

Annex A
(normative)

C-XMDF reader's format

A.1 About compact XMDF

The present International Standard describes an e-book format targeted at mobile phones. Similarly to HTML, text is not divided into pages, but dynamically arranged according to the screen and font size. This type of layout will be called flowing layout.

Compact XMDF documents may be divided into several parts, for distribution or retail purposes.

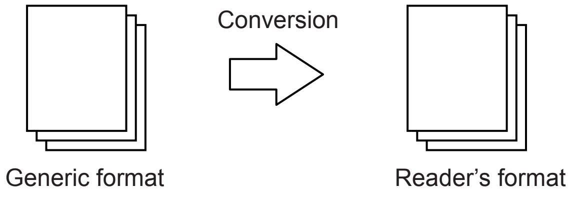
A compact XMDF document is usually made of several flows. These data structures are meant to match a meaningful division of the document, such as chapters. Flows are displayed in the order specified in the content data. The viewer should do a page break when moving from a flow to the next one. The specific way a document is to be divided into flows is not specified, and is left to the document preparer’s discretion. Chapters of a novel or articles of a newspaper are good examples of possible divisions. Note that it is perfectly acceptable not to organize the document in several flows, and store every thing in only one flow. There are two types of flows: formatted text flows, and image base cell flows. In this text, when the type of the flow is obvious from the context, or not relevant to the discussion at hand, they may be simply referred to as flows.

To meet both the requirements of ease of edition during the preparation phase, and ease of processing by the software viewer, there are two corresponding format, as described in Table A.1.

Table A.1 – File types

Type	Details
Generic format	Format used for creation and edition of the document. It is meant to be easy to modify with a text editor, and is based on XML and CSV files.
Reader’s format	Format used for distribution purposes. It is designed to be easy to process by the viewer software, and is a binary format.

Both formats have equivalent semantics, and the generic format is to be translated to the reader’s format prior to distribution, possibly using conversion software.



IEC 191/09

Figure A.1 – Relation between generic and reader's formats

This standard describes the binary reader’s format.

A.2 Terms and definitions

For the purposes of this annex, the following terms and definitions apply.

A.2.1
checksum
sum one by one of all the bytes of the file (the checksum itself excluded) stored as a 4 byte field

A.2.2
cxmdf_char type
an integer ranging from 0 to 255, stored in one byte

A.2.3
cxmdf_int type
an integer ranging from 0 to 4 294 967 295, stored in four bytes in big-endian order

A.2.4
cxmdf_short type
an integer ranging from 0 to 65 535, stored in two bytes in big-endian order

A.2.5
cxmdf_Sshort type
an integer ranging from −32 768 to 32 767, stored in two bytes in big-endian order

A.2.6
cxmdf_string type
data type used to store character strings

See Table A.2 for details.

Table A.2 – Cxmdf_string type

Type/number of bytes	Content	Details
Cxmdf_char	Length of the string	Records the number of bytes used to store the string.
n	String	The number of bytes of this field is determined by the previous field. If 0, then this field does not exist. The string is to be interpreted according to the encoding specified in the character encoding flag of the root file (see A.4.1).

A.2.7
flow number
flows are numbered in order, starting at 0; the first flow is 0, the second is 1; the third is 2 and so on

A.2.8
half width and full width characters
characters called half width are the usual range of Latin characters, and similarly sized characters; full width characters, primarily used by CJK languages, are typically twice as large, even in mono space fonts

NOTE Most half width characters, including the space character, have a full width version, visually distinct, used for typographic purposes.

A.2.9
horizontal and vertical writing
text can be displayed in horizontal, left-to-right lines, beginning at the top of the page and going down, or in vertical top-to-bottom lines, beginning at the right of the page, and going left

NOTE The vertical mode is primarily used for CJK languages.

A.2.10
image number
images are numbered in their order of registration starting from 0, within the range specified in the root file as number of image data (see A.4.1)

A.2.11
in-line note character
in-line note characters are characters used to denote the reading, or pronunciation of other data or for other purposes (in-line notes)

NOTE Characters used in in-line notes are language and application dependent. This International Standard distinguishes in-line note characters from standard characters due to the possibility that the viewer needs special processing to handle in-line notes because of their character size, layout etc.

A.2.12
sound number
sounds are numbered in their order of registration starting from 0, within the range specified in the root file as number of sound data (see A.4.1)

A.2.13
standard character
standard characters are characters of the set specified in the root file (see A.4.1) with the standard character string flag

A.2.14
standard character string
strings made of standard characters

A.2.15
tag
basic unit of control information, used to carry the display settings to be applied to flows' text, such as paragraphs, font settings, pictures to be inserted, and so on

NOTE Tags are composed of a number identifying the feature they are related to, and parameters to give the setting's details.

A.2.16
unused region
region not used in this standard, which the C-XMDF viewer program is to ignore

A.3 File organization

A.3.1 Compact XMDF format

The compact XMDF format is made of several files. This clause introduces the different file types, and the naming conventions.

A.3.2 File types

Compact XMDF documents are made of files of types shown in Table A.3.

Table A.3 – File types

File type	Content
Root file	The first file that the viewer software must access. It contains bibliographic data, determines whether there is an index or not, how many flows, images or sound are used, and so on. There is always one root file.
Text flow body file	The file storing the unformatted text of a text flow. There is one such file per text flow. All formatting information is stored separately in the text flow control file.
Text flow control file	Used when displaying a text flow, this file contains formatting information to be applied to the raw text of the text flow body file. It provides information on paragraphs, font style, inserted picture, links, and so on. There is one such file per text flow.
Cell flow control file	The file used to describe the content of cell flows. It stores information on which picture to use, on the transition effects, hyperlinks, and so on. There is one such file per cell flow. Unlike text flows, this is the only file needed.
Index body file	File storing the unformatted text of the index, similarly to text flow body files.
Index control file	File storing the formatting information of the index. It has the same structure as text flow control files.
Picture file	Files storing the images used in text of cell flows. As of publication of this standard, JPEG, PBM, GIF and MIG (see A.6) can be used.
Sound file	Files storing the sounds used as background music in the flows. As of the current specification, MFi (not supported on all devices), SMF and SP-MIDI (not supported on all devices) can be used.

A.3.3 File naming conventions

All files composing a compact XMDF document are named according to the convention detailed in Table A.4. As explained in A.4.1, there is no field in a compact XMDF document to record the name of the different files used. File names, which are needed to access them or download them from the server, are determined by convention instead.

Table A.4 – File naming conventions

File type	Naming convention	File name example	Abbreviation	Notes
Root file	root.cxf (fixed)	Only the preceding name	none	
Text flow body file	f+ flow number +.txt	File name for the flow number 2: f2.txt	t+ flow number	See A.2 about flow numbers
Text flow control file	f+ flow number +.ctl	File name for the flow number 0: f0.ctl	c+ flow number	See A.2 about flow numbers
Cell flow control file	f+ flow number +.kom	File name for the flow number 3: f3.kom	k+ flow number	See A.2 about flow numbers
Index body file	index.txt (fixed)	Only the preceding name	none	
Index control file	index.ctl (fixed)	Only the preceding name	none	
Picture file	i+ image number + extension	File name for the picture number 15, of jpeg type: i15.jpg	[JPEG] j+ image number [PBM] p+ image number [MIG] i+ image number [GIF] g+ image number	See A.2 about image numbers The extension depends on the format in which the image is encoded. JPEG : .jpg PBM : .pbm MIG : .mig GIF : .gif
Sound file	s+ sound number + extension	File name for the sound number 15, of mld type: s15.mld	[MFi] m+ sound number [SMF/SP-MIDI] s+ sound number	See A.2 about sound numbers The extension depends on the format in which the sound is encoded. As of publication of this standard, the following formats can be used. MFi: .mld SMF/SP-MIDI: .mid

Based on these rules, to determine the name of all files, the only information needed is: the number of flows, the type of each flow, the number of images, the format of each image, the number of sounds, the format of each sound and the availability of the index. For example, a document defined as having

- 3 text flows (in the order : text, text, cell),
- 3 images, all of which are jpegs,
- 2 sounds, the first being MFI, and the second SMF,
- an index,

would be composed of the following 12 files.

> root.cxf, f0.txt, f1.txt, f0.ctl, f1.ctl, f 2.kom, index.txt, index.ctl, i0.jpg, i1.jpg, i2.jpg, s0.mld, s1.mid

Note that as all compact XMDF documents use the same convention, there will be duplicated file names between different documents. Care shall be taken not to mix up the files.

A.3.4 Media types of the files

The different files that compose a compact XMDF document have media types shown in Table A.5. They are used, for example, when downloaded form a contents server.

Table A.5 – Media types

File type	media type	notes
Root file	application/x-cxmdf	
Text flow body file	text/plain	
Text flow control file	application/x-cxmdf-ctl	
Cell flow control file	application/x-cxmdf-koma	
Index body file	text/plain	
Index control file	application/x-cxmdf-ctl	Identical to text flow control file
Picture file	JPEG : image/jpeg PBM : image/pbm MIG : image/x-cxmdf-mig GIF : image/gif	Set to the appropriate media type, depending on the actual image format.
Sound file	MFi: audio/mfi SMF/SP-MIDI: audio/midi	Set to the appropriate media type, depending on the actual sound format.

A.4 Details of the reader’s format

A.4.1 Root file

This is the file that the viewer software must read first. It contains the number and type of flows, the availability of an index, the number and type of images and sounds, bibliographic data, and so on. Any compact XMDF document has one and only one root file.

Its data structure is shown in Table A.6

Table A.6 – The data structure of root file

Type/Number of bytes	Content	Details
4	Compact XMDF ID number	Identifies the file as a compact XMDF document. It is defined as the four characters: CMDF. On any other value, the viewer software should abort reading the file.
4	Compact XMDF version number	Identifies the version of the compact XMDF format used by the current document, as a sequence of four characters. As of publication of this standard, it is set to 1,40. On any non-supported value, the viewer software should abort reading the file.
Cxmdf_char	Number of standard character sets	The number of character sets that can be used by the standard character strings. Cannot be 0. If more than 1 character set is specified, their union must be used.
Cxmdf_char	Standard character set flag	Indicates the character set used by the standard character strings. As of the current specification, the following values can be used: (See A.7.1 for other values) 0x00: JIS X 0201 and JIS X 0208: 1997) 0x01: The Big5 character set.

Type/Number of bytes	Content	Details
		0x02: The GB2312 character set 0x03: The ISO-IR-149 character set 0x04: The US-ASCII character set 0x05 The ISO-8859-15 character set On unaccepted values, the viewer software should abort reading the file.
…	…	“Standard character set flag” repeated, according to “Number of standard character sets”
Cxmdf_char	Character encoding flag	Defines the character encoding used in the document. It applies to all character strings in the document, including bibliographic data, the body of text flows or of the index. As of the current specification, the following value can be used. 0x01: UTF-16 (Big-Endian without BOM) On any other value, the viewer software should abort reading the file.
Cxmdf_char	Content type	Defines which types of flow can be used in the document bit7(MSB) Text flows 0: Text flows are not used 1: Text flows are used bit6: Cell flows 0: Cell flows are not used 1: Cell flows are used bit5 - 0: These bits are not used, and are always set to 0.
Cxmdf_short	Unused region	Reserved for future extensions. Set to 0x0000. On any other value, the viewer software should abort reading the file.
Cxmdf_int	Total size	The total size of this Compact XMDF document, in bytes, consisting of the sum of the size of all its files: root file, index and text flow body and control files, cell flow files, images and sounds.
Cxmdf_int	Flow body total size	The total size of all the text flow body files used in this compact XMDF document, in bytes.
Cxmdf_int	Total image size	The total size of all the image files used in this compact XMDF document, in bytes.
Cxmdf_int	Total sound size	The total size of all the sound files used in this compact XMDF document, in bytes.
Cxmdf_int	Recommended Download size	To guarantee a comfortable reading experience, at least this size should be available to the viewer software to download the flows immediately preceding and following the current flow.
Cxmdf_char	Screen size settings flag	Determines whether the intended screen size at the time of content creation is stored or not. 0x00: not stored, 0x01: stored
Cxmdf_short /0	Recommended x size	The intended screen width at the time of content creation. Only stored in screen size settings flag is set to 0x01.
Cxmdf_short /0	Recommended x size	The intended screen height at the time of content creation. Only stored in screen size settings flag is set to 0x01.
Cxmdf_short	Number of flows	Records the number of flows in the document.
Cxmdf_char	Type of the flow 0	Records the type of the first flow (flow number 0). The following values can be used. 0x00: Text flow, 0x01: Cell flow
Cxmdf_char	Flow 0 dead-end flag	Records the dead-end settings for the flow number 0. bit7(MSB) Defines whether it is allowed to move from this flow to the previous one. 0: Allowed 1: Disallowed bit6: Defines whether it is allowed to move from this flow to the next one. 0: Allowed 1: disallowed bit5 - 0: These bits are not used, and are always set to 0.

Type/Number of bytes	Content	Details
Cxmdf_short/0	Size of flow 0 body file	Records the size, in bytes, of the body file of the flow number 0. This field exists only if the flow type is text.
Cxmdf_int/0	Checksum of flow 0 body file	Records the checksum of the body file of the flow number 0. This field exists only if the flow type is text.
Cxmdf_short	Size of flow 0 control file	Records the size, in bytes, of the control file of the flow number 0.
Cxmdf_int	Checksum of flow 0 control file	Records the checksum of the control file of the flow number 0.
Cxmdf_short Cxmdf_short Cxmdf_short …	Number of images used by flow 0 Number of image 0 Number of image 1 …	Records the number of images used in the flow number 0. The document-global image number of the first (number 0) picture used in this flow. The document-global image number of the second (number 1) picture used in this flow. When the number of images used by flow … field is 0, subsequent information is skipped.
Cxmdf_short Cxmdf_short Cxmdf_short …	Number of sounds used by flow 0 Number of sound 0 Number of sound 1 …	Records the number of sounds used in the flow number 0. The document-global sound number of the first (number 0) picture used in this flow. The document-global sound number of the second (number 1) picture used in this flow. When the number of sound used by flow … field is 0, subsequent information is skipped.
Cxmdf_char Cxmdf_char Cxmdf_short Cxmdf_int Cxmdf_short Cxmdf_int Cxmdf_short Cxmdf_short Cxmdf_short … Cxmdf_short Cxmdf_short Cxmdf_short …	Type of the flow 1 Flow 1 dead- end flag Size of flow 1 body file Checksum of flow 1 body file Size of flow 1 control file Checksum of flow 1 control file Number of images used by flow 1 Number of image 0 Number of image 1 … Number of sounds used by flow 0 Number of sound 0 Number of sound 1 …	Records the information regarding the flow number 1 (second flow) in the same format as the flow number 0.
…	…	Repeated for each flow,
Cxmdf_char	Index availability flag	Determines whether the document has an index or not. If available, the index body file and index control file exist. The following values can be used. bit7(MSB): 0: not available, 1: available bit6 – 0: These bits are not used, and are always set to 0.
Cxmdf_short/0	Index body file size	Records the size, in bytes, of the body file of the index.
Cxmdf_int/0	Index body file checksum	Records the checksum of the body file of the index.
Cxmdf_short/0	Index control file size	Records the size, in bytes, of the control file of the index.
Cxmdf_int/0	Index control file checksum	Records the checksum of the control file of the index.
Cxmdf_int/0	Total size of images used in index	Records the total size in bytes of images used only in the index.
Cxmdf_int/0	Total size of sounds used in index	Records the total size in bytes of sounds used only in the index.
Cxmdf_short Cxmdf_short Cxmdf_short …	Number of images used in index Number of image 0 Number of image 1 …	Records the number of images used in the index. The document-global image number of the first (number 0) picture used in the index. The document-global image number of the second (number 1) picture used in the index. When the “number of images used by index” field is 0, subsequent information is skipped.
Cxmdf_short Cxmdf_short Cxmdf_short …	Number of sound data used in index Number of sound 0 Number of sound 1 …	Records the number of sound data used in the index. The document-global sound number of the first (number 0) picture used in the index. The document-global sound number of the second (number 1) picture used in the index. When the number of sound used by index field is 0,

Type/Number of bytes	Content	Details
		subsequent information is skipped.
		Note that these fields only exist if the index availability flag is set to 1.
Cxmdf_short	Number of images	Records the number of images used in the whole document. The following fields are repeated the number of times specified here. The order in which they are recorded defines the image number.
Cxmdf_char	File flag for Image 0	The various bits of this flag field give general information about this image's usage. bit7(MSB) Determines whether the image is only used in the index, or in other parts too. 1: Index only 0: Main text only, or main text and index. bit6: Determines whether the image is used as a background or not. 1: Used as a background. 0: Not used as a background. bit5: Determines whether the image is used a cell image or not. 1: Used as a cell image. 0: Not used as a cell image bit4: This bit is not used, and is always set to 0. bit3 – 2: These bits determine the copy protection of the image. The following values can be used. 11, 00: Copy is forbidden. 10: Copy is allowed only to areas of the device under full control of the viewer software. If the viewer software cannot prevent these copied files to be exported from the device, then no copy at all, even internal, is allowed. 01: Copy is allowed. bit1: Determines whether the image is used in an image tag (see A.5.9) or not. 1: Used in an image tag. 0: Not used in an image tag. bit0: Determines whether the image is used in an external character tag (see A.5.8) or not. 1: Used in an external character tag. 0: Not used in an external character tag.
Cxmdf_char	Encoding format of image 0	Specifies the encoding format of the image. The following values can be used. 0x00 : jpeg 0x01 : pbm 0x02 : mig 0x03 : gif
Cxmdf_short	Height of image 0	Records the height of the first image (image 0) in pixels.
Cxmdf_short	Width of image 0	Records the width of the first image (image 0) in pixels.
Cxmdf_int	File size of image 0	Records the size of the first image (image 0) in bytes
Cxmdf_int	Checksum of image 0	Records the checksum of the first image (image 0).
Cxmdf_char Cxmdf_char Cxmdf_short Cxmdf_short Cxmdf_int Cxmdf_int	File flag for Image 1 Encoding format of image 1 Height of image 1 Width of image 1 File size of image 1 Checksum of image 1	
…	…	Repeated for each image.
Cxmdf_short	Number of sounds	Records the number of sounds used in the whole document. The following fields are repeated the number of times specified here. The order in which they are recorded defines the sound number.

Type/Number of bytes	Content	Details
Cxmdf_char	File flag for sound 0	The various bits of this flag field give general information about this sound's usage. bit7(MSB) Determines whether the sound is only used in the index, or in other parts too. 1: Index only 0: Main text only, or main text and index. bit6: Determines whether the sound is used as background music in text flows or not. 1: Used as background music. 0: Not used as background music. bit5: Determines whether the sound is used as sound effect in cell flows or not. 1: Used as sound effect. 0: Not used as sound effect. bit4: This bit is not used, and is always set to 0. bit3 – 2: These bits determine the copy protection of the image. The following values can be used. 11, 00: Copy is forbidden. 10: Copy is allowed only to areas of the device under full control of the viewer software. If the viewer software cannot prevent these copied files to be exported from the device, then no copy at all, even internal, is allowed. 01: Copy is allowed. bit1-0: These bits are not used, and are always set to 0.
Cxmdf_char	Encoding format of sound data 0	Specifies the encoding format of the sound data. The following values can be used. 0x00: MFi 0x01: SMF/SP-MIDI
Cxmdf_int	File size of sound data 0	Records the size of the first sound data (sound data 0) in bytes
Cxmdf_int	Checksum of sound data 0	Records the checksum of the first sound data (sound data 0).
…	…	Repeated for each sound.
Cxmdf_short	Bibliographic data flag	This field determines which bibliographic data is stored in the document, for the various possible. The signification of each bit is as follows. bit15(MSB）-8: These bits are not used, and are always set to 0. bit7: Determines whether the title is stored or not. The following value can be used. 1: Stored. bit6: Determines whether the title's reading (pronunciation) is stored or not. The following values can be used. 0: Not stored, 1: Stored bit5: Determines whether the subtitle is stored or not. The following values can be used. 0: Not stored, 1: Stored bit4: Determines whether the book's ID is stored or not. The following value can be used. 1: Stored bit3: Determines whether the name of the author is stored or not. The following values can be used. 0: Not stored, 1: Stored bit2: Determines whether the reading (pronunciation) or the name of the author is stored or not. The following values can be used. 0: Not stored, 1: Stored bit1: Determines whether the name of the publisher is stored

Type/Number of bytes	Content	Details
		or not. The following values can be used. 0: Not stored, 1: Stored bit0: Determines whether the cover picture is stored or not. The following values can be used. 0: Not stored, 1: Stored
Cxmdf_string /0	Bibliography – Title	Records the title of the document, using a standard character string. Maximum 160 bytes. Only stored if the relevant bit is set to 1 in the bibliographic data flag.
Cxmdf_string /0	Bibliography – Title reading	Records the reading of the title of the document, using a standard character string. May be used for sorting purposes. Maximum 160 bytes. Only stored if the relevant bit is set to 1 in the bibliographic data flag.
Cxmdf_string /0	Bibliography – Subtitle	Records the subtitle of the document, using a standard character string. Maximum 160 bytes. Only stored if the relevant bit is set to 1 in the bibliographic data flag.
Cxmdf_string /0	Bibliography – ID	Records the ID of the document, using a standard character string. Maximum 80 bytes. Only stored if the relevant bit is set to 1 in the bibliographic data flag.
Cxmdf_string /0	Bibliography – Author	Records the name of the author of the document, using a standard character string. If there are several authors, they are combined in this string. Maximum 160 bytes. Only stored if the relevant bit is set to 1 in the bibliographic data flag.
Cxmdf_string /0	Bibliography – Author reading	Records the reading of the name of the author of the document, using a standard character string. If there are several authors, they are combined in this string. Maximum 160 bytes. Only stored if the relevant bit is set to 1 in the bibliographic data flag.
Cxmdf_string /0	Bibliography – Publisher	Records the name of the publisher of the document, using a standard character string. Maximum 160 bytes. Only stored if the relevant bit is set to 1 in the bibliographic data flag.
Cxmdf_short /0	Cover image number	The image number of the picture used as a cover. Only stored if the relevant bit is set to 1 in the bibliographic data flag.
1	Unused region	Reserved for future extensions. Set to 0x00. On any other value, the viewer software should abort reading the file.
Cxmdf_int	Checksum of the root file	The checksum of this root file, from the beginning up to the field immediately preceding the checksum itself.

A.4.2 Text flow body file

This is the file that carries in plain text the body of each flow. There is one such file per text flow. As explained in A.3.2, to properly display the text, both this file and the text flow control file, which carries formatting information, are needed.

Attention should be paid to the following points.

- The text flow body file contains standard character strings.
- The encoding used is the one specified in the character encoding flag of the root file.
- The viewer must show a page break when switching between flows.
- Characters having a ruby string (see A.5.6) attached to them are stored in this file; while the ruby string is stored in the control file corresponding to this flow.
- Information on the alternative image used for external characters (see A.5.8) is stored in the control file, while the character string to display when not using the image is directly stored in the text flow body file.
- Among the characters allowed in the text flow body file, special characters shown in Table A.7 are to be handled as described.

Table A.7 – Special characters

Character	Behavior
Half width space (0x20)	Displayed as is
Line breaks (0x0D, 0x0A, 0x0D0A)	Ignored
Tabulation (0x09)	Displayed as a half width space (0x20)

- When displaying text in Latin characters, the line break should be between words, where a space (0x20) or tabulation (0x09) character is found. In case there a space-less and tab-less part of the text is too long to fit a line, the viewer should split it. The point where such a split occurs is implementation dependant.

- The text flow body file can not be longer than 65 535 bytes.

A.4.3 Text flow control file

A.4.3.1 Control file

The formatting information needed to properly compute the layout of the text in a flow body file is stored in the control file. Similarly to the body file, there is one control file per flow. Its content shall be interpreted together with the flow body file.

The details of the organization of the control file are given in the following subclauses.

A.4.3.2 About the division in blocks

Usually, the user reads the document from the beginning, scrolling through the flow in order. Computing the layout in this case poses no major problem, as the position of new elements can be calculated relatively to those already displayed. However, by following links or using bookmarks, it may happen that the user jumps directly to the middle of a flow. In that case, to correctly determine the position of the elements to be displayed, the position of everything before it would need to be computed, even for parts that would not be displayed at all. This would have an impact on the speed of the layout algorithm for long flows. Moreover, it would require keeping in memory the layout information for a full flow, which is not desirable.

On the other hand, it is not a good solution to ignore everything that precedes what you want to display. For example, consider the case where we want to display the text illustrated in Figure A.2 from the character at position A. If we simply ignore what’s before it, the character at position B will not appear at the beginning of a line when the user scrolls backwards from position A.

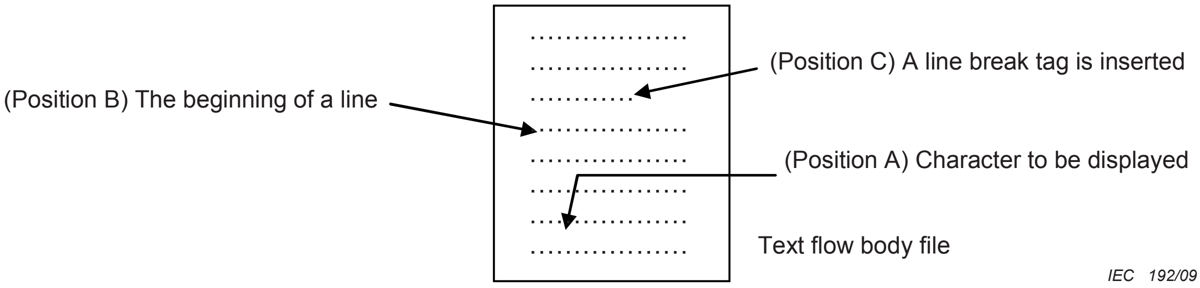

(Position C) A line break tag is inserted
(Position B) The beginning of a line
(Position A) Character to be displayed
Text flow body file
IEC 192/09

Figure A.2 – A visual example of a text flow

The solution adopted by the present format is based on the idea that any character (such as B in the previous example) following a tag which causes a line break will necessarily be at the beginning of a line. The flow is divided into logical blocks, each beginning necessarily on a new line. The formatting tags (paragraph margin size, font color and so on) are grouped block by block and stored in the text flow control file.

In the following subclauses, the structure of the text flow control file will be detailed.

A.4.3.3 Text flow control file

The data structure of the text flow control file is detailed in Table A.8.

Table A.8 – Data structure of text flow control file

Type / number of bytes	Content	Details
2	Text flow control file ID	Identifies the file as a text flow control file. It is defined as the two characters: FC. On any other value, the viewer software should abort reading the file.
Cxmdf_char	Flow attributes 1	Records attributes of the flow. The following fields are defined. bit7(MSB) - 6: Determines the direction of the text. The following values can be used. 00: Horizontal, left to right. Vertical is also allowed, if demanded by the user. 01: Vertical, top to bottom. Horizontal is also allowed, if demanded by the user. 10: Horizontal left to right. Vertical is forbidden. 11: Vertical, top to bottom. Horizontal is forbidden. bit5: Determines whether the text body's font size is specified or not. The following values can be used. 0: Not specified. 1: Specified bit4 - 3: Determines whether ruby text should be displayed or not. The following values can be used. 00: Depends on the viewer software's setting 10: Ruby is not displayed 11: Ruby is displayed bit2 - 0: These bits are not used, and are always set to 0.
Cxmdf_char	Flow attributes 2	bit7(MSB): Determines whether there is a background picture or not. The following values can be used. 0: There is no background picture 1: There is a background picture bit6: Determines whether there is a background music or not. The following values can be used. 0: There is no background music 1: There is a background music bit5 - 4: These bits are not used, and are always set to 0. bit3 - 2: Determines whether the flow has a default font color or not. The following values can be used. 00: Not set (use the device's default) 01: Set, in monochrome 10: Set, in color bit1 - 0: Determines whether the flow has a default background color not. The following values can be used. 00: Not set (use the device's default) 01: Set, in monochrome 10: Set, in color
Cxmdf_char/0	Text size	Records the text size to use when displaying the text of the flow body. If this size is set, the user can not change it via the preferences menu of the viewer software. Only stored if the relevant bit is set to 1 in the flow attributes 1 data flag. 0x00 : TINY 0x01 : SMALL 0x02 : MEDIUM 0x03 : LARGE
Cxmdf_short/0	Background picture number	Records the number of the picture to use as background. Only stored if the relevant bit is set to 1 in the flow attributes 2 data flag.

Type / number of bytes	Content	Details
Cxmdf_short/0	Background music number	Records the number of the sound to use as background music. Only stored if the relevant bit is set to 1 in the flow attributes 2 data flag.
Cxmdf_char/0	Font color info 1	Stores the R component of the flow's default font color if bits 3-2 of flow attributes 2 are set to 10, or the grayscale level if they are set to 01.
Cxmdf_char/0	Font color info 2	Stores the G component of the flow's default font color if bits 3-2 of flow attributes 2 are set to 10. Omitted for other values.
Cxmdf_char/0	Font color info 3	Stores the G component of the flow's default font color if bits 3-2 of flow attributes 2 are set to 10. Omitted for other values.
Cxmdf_char/0	Background color info 1	Stores the R component of the flow's default background color if bits 1-0 of flow attributes 2 are set to 10, or the grayscale level if they are set to 01.
Cxmdf_char/0	Background color info 2	Stores the G component of the flow's default background color if bits 1-0 of flow attributes 2 are set to 10. Omitted for other values.
Cxmdf_char/0	Background color info 3	Stores the B component of the flow's default background color if bits 1-0 of flow attributes 2 are set to 10. Omitted for other values.
Cxmdf_short	Number of blocks	Records the number of blocks composing the flow. The following data structure is repeated accordingly.
 Cxmdf_short Cxmdf_short Cxmdf_short	 Block 1 start offset in the body file Block 1 control information start offset Block 1 control information data size	Records the following information of the first block. Offset in bytes in the text flow body file of the first character included in the block. Offset in bytes in this file (text flow control file) of the beginning of the control information of this block. Size in bytes of the control information of this block.
 Cxmdf_short Cxmdf_short Cxmdf_short	 Block 1 start offset in the body file Block 1 control information start offset Block 1 control information data size	Records offset and field size information about the second block.
…	…	Repeated for each block.
1	Unused region	Set to 0x00 as of the present specification
n	Control information of block 1	Records the control information of the first block. The details of this data structure are explained in A.4.3.4. The starting point and size of this field correspond to the information store in the block 1 control information start offset and block 1 control information data size fields.
n	Control information of block 2	Records the control information of the second block.
…	…	

A.4.3.4 Block control information

The tag information of each block of a text flow is stored in block control information whose data structure, is described in Table A.9.

Table A.9 – Data structure of block control information

Type / number of bytes	Content	Details
2	Block control information ID	An identification number marking the block control information data structure. It is defined as the two characters: BC. On any other value, the viewer software should abort reading the file.
Cxmdf_short	Number of tags	Records the number of tags used in this block. The tags are stored in the order of appearance in the flow.
		Information about the first tag
Cxmdf_short	Tag 1 start offset	Records the offset (in bytes) from the beginning of the text flow body file where the tag applies.
Cxmdf_char	Tag 1 tag type number	Records the number identifying the tag. Each tag type has a different number, see A.5 for details.
Cxmdf_short	Tag 1 parameter data offset	Records the offset (in bytes) from the beginning of this file (text flow control file) where the tag's parameters are stored. If the tag has no parameter, it is set to 0xFFFF.
		Information about the second tag
Cxmdf_short Cxmdf_char Cxmdf_short	Tag 2 start offset Tag 2 Tag type number Tag 2 parameter data offset	
…	…	Repeated for each tag
1	Unused region	Set to 0x00 as of the present specification
n	Tag parameter 1	• Stores the parameters associated with a tag. The meaning of the information stored here depends on the type of tag, as stored in the tag n tag type number. For details of the data structure in each case, see A.5. • Note that this tag parameter 1 is not necessarily related to the first tag of the block. Indeed, some tags do not have any parameters. Therefore, this field should always be located by reading the tag n parameter data offset field of the tag information.
n	Tag parameter 2	
…	…	

To make things clear, here is a detailed explanation of how offset works for tags. As said above, the tag n start offset records the position in the text flow body file where the tag applies. But depending on the type of tag, it can have slightly different meanings. If the type does something punctual, like a line break, or an image insertion, then its effect happens right before the indicated offset. For example, if a line break tag's start offset is 25, the line break must be inserted between the 24th and 25th byte of the text flow body file. Some other tags apply on a range of characters, like the font tag, used amongst other things to change the text color. In this case, the offset designates the first character to which the effect applies. The tag parameter indicates where the effect ends.

It may happen that several tags point to the same offset. In that case they apply in the order of appearance in the text flow control file. We give here two examples, but other tag combinations work similarly.

For example, if a picture tag and a mask tag point at the same position, depending on the order in which these tags appear in the text flow control file, the result changes as follows.

a) Image tag followed by mask tag: As the image comes first, it is inserted before the mask beings, and therefore is not covered by it.

b) Mask tag followed by image tag: Since the mask tag comes first, the image is inserted inside the mask, and will be covered by it.

Consider now having two font tags at the same position, specifying different colors. The reasoning is similar.

c) Blue font tag, followed by red font tag: Characters are first set to be displayed blue, but before any has been shown, they are set to red, so the final color used for display is red.

d) Red font tag, followed by blue font tag: Characters are first set to be displayed red, but before any has been shown, they are set to blue, so the final color used for display is blue.

Normally, tags stored within a block control information data structure only point to offset within the block. However, it may happen that they point to block boundary offsets. This can only happen for four tags. For each, Table A.10 lists in which of the current and the following blocks they should be recorded.

Table A.10 – Treatment of block boundary

Tag name	Block to be recorded in
Paragraph tag (see A.5.2)	Next block
Line break tag (see A.5.3)	Current block
Horizontal line tag (see A.5.4)	Next block
Image tag (see A.5.9)	Next block

A.4.4 Cell flow control file

A.4.4.1 Cell control flow

The cell flow control file record details on how to display cell flows, that is to say, image based flows. There is one such file per cell flow. However, unlike text flows, there is no body file.

A cell flow is composed of several cells, each cell having one or more scene(s). Figure A.3 illustrates the idea of cell flows, cells and scenes.

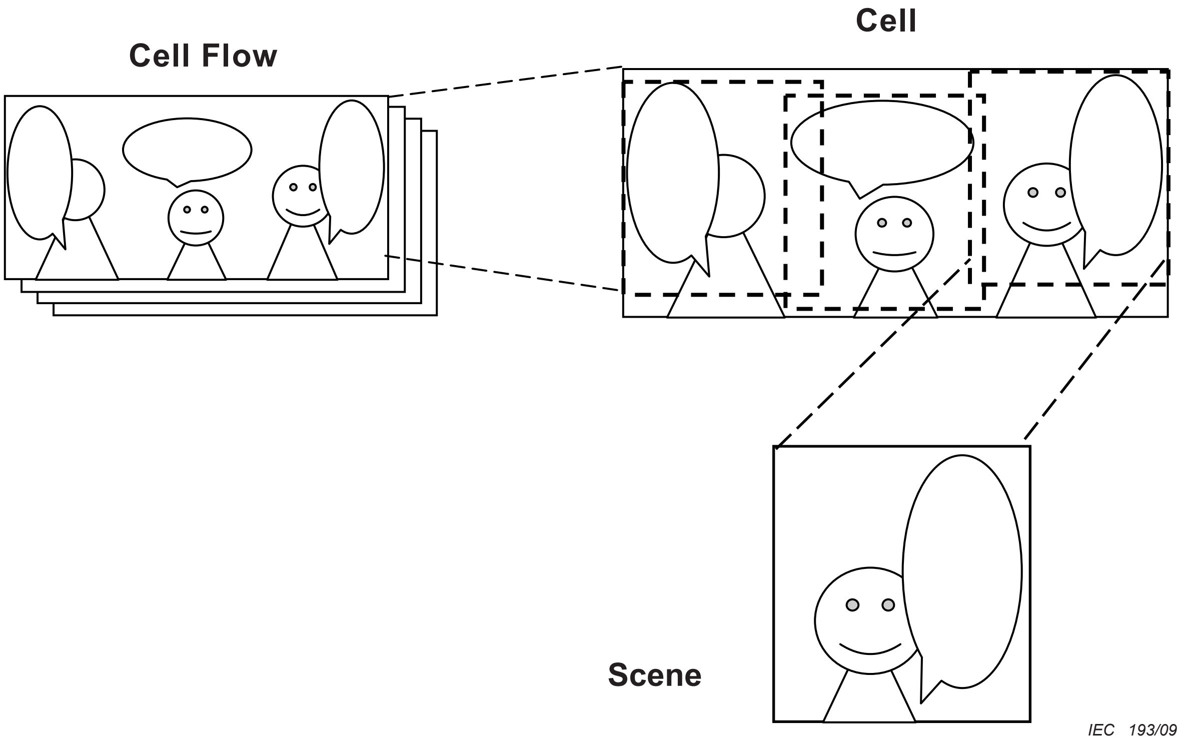


Figure A.3 – Relationship between cell flow, cell and scene

The viewer software should display a cell in the following steps.

a) Display the first scene of the first cell.

b) Wait for the user to press the next button. If scrolling is activated for this scene, scroll smoothly to the next scene, otherwise, jump directly to c).

c) Display the second scene of the first cell.
d) Wait for user input to move to the next scene.
e) Display the third scene of the first cell.

When all the scenes of a cell have been displayed, and when the user presses the next button, replace the current cell with the next one, using any transition effect specified, and display its scenes, one by one, in the same way as the previous cell.

Table A.11 specifies the details of the data structure of the cell flow control file.

Table A.11 – Data structure of cell flow control file

Type / number of bytes	Content	Details
2	Cell flow control file ID	Identifies the file as a cell flow control file. It is defined as the two characters: KC. On any other value, the viewer software should abort reading the file.
Cxmdf_char	Flow attributes 1	bit7(MSB) - 2: These bits are not used, and are always set to 0. bit1 - 0: Determines whether the flow has a default background color or not. The following values can be used. 00: Not set (use the device's default) 01: Set, in monochrome 10: Set, in color
Cxmdf_char/0	Background color info 1	Stores the R component of the flow's default background color if bits 1-0 of flow attributes 1 are set to 10, or the grayscale level if they are set to 01.
Cxmdf_char/0	Background color info 2	Stores the G component of the flow's default background color if bits 1-0 of flow attributes 1 are set to 10. Omitted for other values.
Cxmdf_char/0	Background color info 3	Stores the B component of the flow's default background color if bits 1-0 of flow attributes 1 are set to 10. Omitted for other values.
Cxmdf_short	Number of cells	Records the number of cells used in this flow. The following data structure is repeated accordingly. The cells are stored in order of appearance.
Cxmdf_short	Cell picture 1 Control information position	Records the offset in bytes from the start of this cell flow control file where the control information regarding the first cell picture is stored.
Cxmdf_short	Cell picture 2 Control information position	Records the offset in bytes from the start of this cell flow control file where the control information regarding the second cell picture is stored.
…	…	Repeated for each cell
1	Unused region	Set to 0x00 as of the present specification
n	Cell picture 1 Cell control information	Stores the control information for the first cell picture. See A.4.4.2 for details on the data structure.
n	Cell picture 2 Cell control information	Stores the control information for the second cell picture.
…	…	Repeated for each cell

A.4.4.2 Cell control information

Table A.12 details the cell control information data structure, found in the cell flow control file for each cell.

Table A.12 – Data structure of cell control information

Type / number of bytes	Content	Details
2	Cell control information ID	An identification number marking the cell control information data structure. It is defined as the two characters: KI. On any other value, the viewer software should abort reading the file.
Cxmdf_short	Cell's image number	Records the image number of the image to use as a base for this cell.
Cxmdf_char	Feature attribute flag	Records attributes for each feature of the image bit7(MSB): Determines whether transition effects should be used or not. 0: Not used 1: Used bit6-3: These bits are not used, and are always set to 0. bit2: Determines whether the image is allowed be shrunk to fit the screen size. 0: Not allowed 1: Allowed Even if set to 1, the image is shrunk only if all the following conditions are met: • There is only one scene in the cell. • The image is larger than the size of the display area used by the viewer. • The viewer supports this fitting feature. bit1- 0: Determines whether the cell has a background color or not. The following values can be used. 00: Not set (use the flow's default) 01: Set, in monochrome 10: Set, in color
Cxmdf_char/0	Cell transition effect	Only stored when the bit 7 of feature attribute flag is set to 1. 0x00: fade-in The screen is blanked to the background color, and the new cell's image is displayed progressively fading in. 0x01: dissolve The current cell's image is displayed progressively over the previous cell's image. This is similar to the fade-in effect, except that the screen is not blanked before displaying the new cell. 0x02: slide-right The screen is blanked to the background color, and the new cell's image slides in rightwards, starting from the left side. 0x03: slide-left The screen is blanked to the background color, and the new cell's image slides in leftwards, starting from the right side. 0x04: slide-up The screen is blanked to the background color, and the new cell's image slides in upwards, starting from the bottom. 0x05: slide-down The screen is blanked to the background color, and the new cell's image slides in downwards, starting from top. 0x06: overwrite-right The new cell's image slides in rightwards, starting from the left side, covering the previous cell's image. 0x07: overwrite-left The new cell's image slides in leftwards, starting from the right side, covering the previous cell's image. 0x08: overwrite-up The new cell's image slides in upwards, starting from the bottom, covering the previous cell's image. 0x09: overwrite-down The new cell's image slides in downwards, starting from top, covering the previous cell's image 0x0A: box-center The screen is blanked to the background color, and a progressively enlarging square portion of the new cell's image is displayed, starting in the center, and growing outwards. 0x0B: box-rightdown

Type / number of bytes	Content	Details
		The screen is blanked to the background color, and a progressively enlarging square portion of the new cell’s image is displayed, starting at the upper left corner, and growing towards the lower right one. 0x0C: box-leftdown The screen is blanked to the background color, and a progressively enlarging square portion of the new cell’s image is displayed, starting at the upper right corner, and growing towards the lower left one. 0x0D: box-rightup The screen is blanked to the background color, and a progressively enlarging square portion of the new cell’s image is displayed, starting at the lower left corner, and growing towards the upper right one. 0x0E: box-leftup The screen is blanked to the background color, and a progressively enlarging square portion of the new cell’s image is displayed, starting at the lower right corner, and growing towards the upper left one. 0x0F: box-free The screen is blanked to the background color, and a progressively enlarging square portion of the new cell’s image is displayed, starting at the position specified by the effect_pos attribute, and growing outwards. 0x10: pushout-right The new cell’s image slides in rightwards, starting from the left side, while the previous cell’s image slides out of the screen in the same direction. 0x11: pushout-left The new cell’s image slides in leftwards, starting from the right side, while the previous cell’s image slides out of the screen in the same direction. 0x12: pushout-up The new cell’s image slides in upwards, starting from the bottom, while the previous cell’s image slides out of the screen in the same direction. 0x13: pushout-down The new cell’s image slides in downwards, starting from top, while the previous cell’s image slides out of the screen in the same direction. 0x14: wipe-right The previous cell’s image is progressively replaced by the new one, starting from the left, rightwards. 0x15: wipe-left The previous cell’s image is progressively replaced by the new one, starting from the right, leftwards. 0x16: wipe-up The previous cell’s image is progressively replaced by the new one, starting from bottom, upwards. 0x17: wipe-down The previous cell’s image is progressively replaced by the new one, starting from top, downwards. 0x18: wipe-vertical-in The previous cell’s image is progressively replaced by the new one, starting from top and bottom, towards the center. 0x19: wipe-vertical-out The previous cell’s image is progressively replaced by the new one, starting from the center, towards the top and bottom sides. 0x1A: wipe-horizontal-in The previous cell’s image is progressively replaced by the new one, starting from left and right, towards the center. 0x1B: wipe-horizontal-out The previous cell’s image is progressively replaced by the new one, starting from the center, towards the left and right sides. 0x1C: wipe-center-in The previous cell’s image is progressively replaced by the new one, starting from the four corners, towards the center. 0x1D: wipe-center-out The previous cell’s image is progressively replaced by the new one, starting from the center, towards the four corners.

Type / number of bytes	Content	Details
		0x1E: randomblock The previous cell's image is replaced by displaying blocks of the new cell's image in random order.
Cxmdf_Sshort/0	Transition effect Box center x	Only stored when the transition effect is 0x0F (free box). Records the x coordinate of point from which the box grows.
Cxmdf_Sshort/0	Transition effect Box center y	Only stored when the transition effect is 0x0F (free box). Records the y coordinate of point from which the box grows.
Cxmdf_char/0	Transition effect Display speed	Specifies the desired speed of the transition effect. Only stored when the bit 7 of feature attribute flag is set to 1. Possible values range from 1 (fast) to 10 (slow).
Cxmdf_char/0	Background color info 1	Stores the R component of the cell's background color if bits 1-0 of feature attribute flag are set to 10, or the grayscale level if they are set to 01.
Cxmdf_char/0	Background color info 2	Stores the G component of the cell's background color if bits 1-0 of feature attribute flag are set to 10. Omitted for other values.
Cxmdf_char/0	Background color info 3	Stores the B component of the cell's background color if bits 1-0 of feature attribute flag are set to 10. Omitted for other values.
Cxmdf_char	Number of scenes	Records the number of scenes in the cell. Must be at least 1.
Cxmdf_char	Scene 1 positioning method flag	Determines the method used to position the scene on screen. The following values can be used 0x00: center coordinates 0x01: position flag
Cxmdf_Sshort /0	Scene 1 center x coordinate	The x coordinate of the point in the picture that should be displayed in the middle of the screen. Only stored if positioning method flag is set to 0x00.
Cxmdf_Sshort /0	Scene 1 center y coordinate	The y coordinate of the point in the picture that should be displayed in the middle of the screen. Only stored if positioning method flag is set to 0x00.
Cxmdf_char /0	Scene 1 position flag	Determines how the image should be aligned. Only stored if positioning method flag is set to 0x01. The following values can be used. 0x00: Centered 0x01: Top aligned 0x02: Bottom aligned 0x03: Right aligned 0x04: Left aligned For instance, in the 0x01 case, the image is centered horizontally, and this upper edge is placed at the top of the display area. 0x02 to 0x04 work similarly, for other directions.
Cxmdf_char	Scene 1 feature flag	Determines option for the various features of a scene. bit7(MSB): Determines whether to scroll when moving to the next scene or not. 0: Don't scroll 1: Scroll bit6: Determines whether to make the terminal vibrate when displaying the scene or not. 0: Do not vibrate 1: Vibrate bit5: Determines whether a sound effect should be played or not 0: Do not play 1: Play bit4: Determines whether there is an URL attached to this scene or not. 0: Does not have an URL 1: Has an URL bit3-2: Determines whether the image should shake or not. 00: Does not shake 01: Shakes vertically 10: Shakes horizontally bit1-0: These bits are not used, and are always set to 0.
Cxmdf_char/0	Scene 1 scroll speed	Records the animation time, or scrolling speed, to be used when scrolling from this scene to the next one. Only stored when the bit 7 of scene feature flag is set to 1. Possible values range from 1 (fastest) to 10 (slowest).
Cxmdf_char/0	Scene 1 vibration time	Determines how long the image or the terminal should vibrate. Only stored when the bit 7 of scene feature flag is set to 1, or

Type / number of bytes	Content	Details
		when bits 3-2 are set to 01 or 10. Possible values range from 1 (short time) to 5 (long time)
Cxmdf_short/0	Scene 1 sound effect sound number	Records the sound number of the sound file to be played when displaying this scene. Only stored if bit 5 of scene feature flag is set to 1.
Cxmdf_string/0	Scene 1 web site URL	Records the URL of the web site linked to this page. Only URL beginning with http:// or https:// can be used. Only stored if bit 4 of "scene feature flag" is set to 1.
Cxmdf_char Cxmdf_Sshort /0 Cxmdf_Sshort /0 Cxmdf_char /0	Scene 2 positioning method flag Scene 2 center x coordinate Scene 2 center y coordinate Scene 2 position flag	
Cxmdf_char	Scene 2 feature flag	
Cxmdf_char/0	Scene 2 scroll speed	
Cxmdf_char/0	Scene 2 vibration time	
Cxmdf_short/0	Scene 2 sound effect sound number	
Cxmdf_string/0	Scene 2 web site URL	
…	…	Repeated for each scene
1	Unused region	Set to 0x00 as of publication of this standard.

A.4.5 Index body file, index control file

These two files are used to define the index. They are identical to the text flow body file and text flow control file used by text flow. See A.4.2 and A.4.3 for more details. These two files are only stored if the index availability flag of the root file is set to 1. To allow the user to use the index properly, the viewer software should provide a way to display the index, from the menus or some similar mechanism.

A.5 Control tags

A.5.1 Outline

In this clause, the tags used to control and format text flows will be explained in detail.

A.5.2 Paragraph tag（tag number: 0）

This tag is used to create a paragraph. Generally, a line break is inserted before the beginning of the paragraph. However, if the paragraph would start on a new line anyway, there is no need to add the line break.

Moreover, if the tag occurs in text that is flowing around an image, the new paragraph is to begin after the end of the image. Figure A.4 illustrates the situation. If the tag occurs at point A, the new paragraph will start at point B.

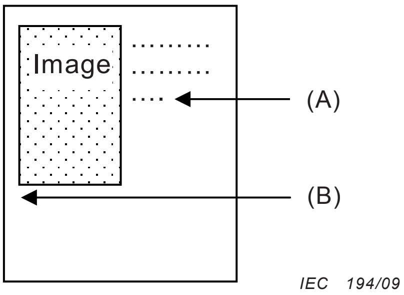


Figure A.4 – Paragraph tag and an image

The settings are applied until the beginning of the next paragraph tag, or the end of the flow.

This tag has the following parameters.

align — Decides between centering, left aligning, or right aligning for the text of the paragraph. When omitted, it defaults to the same setting as the previous paragraph. The following values are possible.

- center — Text and pictures are placed after the insertion point of the tag and are displayed centered.
- right — Text and pictures are placed after the insertion point of the tag and are displayed aligned to the end of the line. In horizontal writing, this means right aligned, and in vertical writing, bottom aligned.
- left — Text and pictures are place after the insertion point of the tag and are displayed aligned to the beginning of the line. In horizontal writing, this means left aligned, and in vertical writing, top aligned.

top_line_indent — Defines the width of the indentation to use on the first line of the paragraph, and on the first line that follows a line break. It is a space inserted between the margin and the first character of the line. Its size is specified in em, which is the width of a character of the current font. Negative values can be used. It defaults to 0 when omitted.

top — Defines the left margin (or upper margin in vertical writing), expressed in em. This is the size of the space inserted before all lines of the paragraph. It defaults to 0 when omitted. Negative values cannot be used.

bottom — Defines the right margin (or lower margin in vertical writing), expressed in em. It defaults to 0 when omitted. Negative values cannot be used.

The parameters of this tag are stored in the block control information as shown in Table A.13.

Table A.13 – Parameters of a paragraph tag stored in block control information

Type / number of bytes	Bit (MSB)	Content	Notes
Cxmdf_char	7-6	Align parameter	Records the value of the align parameter. The following values can be used. ..00: omitted. ..01: right ..10: left ..11: center
	5	top_line_indent parameter presence flag	Records whether the top_line_indent parameter is stored or not. 0: Not stored 1: Stored
	4	Sign of top_line_indent	Records whether the value of top_line_indent should be interpreted as positive or negative. 0: positive or null 1: negative
	3	Top parameter presence flag	Records whether the top parameter is stored or not. 0: Not stored 1: Stored
	2	Bottom parameter presence flag	Records whether the bottom parameter is stored or not. 0: Not stored 1: Stored
	1-0	Unused	Set to 0
Cxmdf_char/0		top_line_indent	Records the value of the top_line_indent parameter. Only stored if the top_line_indent parameter presence fla" is set to 1. Must be interpreted as a negative value if sign of top_line_indent is set to 1.
Cxmdf_char/0		Top	Records the value of the top parameter. Only stored if top parameter presence flag is set to 1.
Cxmdf_char/0		Bottom	Records the value of the bottom parameter. Only stored if bottom parameter presence flag is set to 1.

A.5.3 Line break tag（tag number: 1）

This tag is used to cause a line break. If the text is flowing around a picture (see A.5.9), the flowing can be interrupted. Figure A.5 illustrates the difference between a line break interrupting the flowing text, or leaving it as is. Point A is where the line break tag appears, and point B is where the text resumes after the line break.

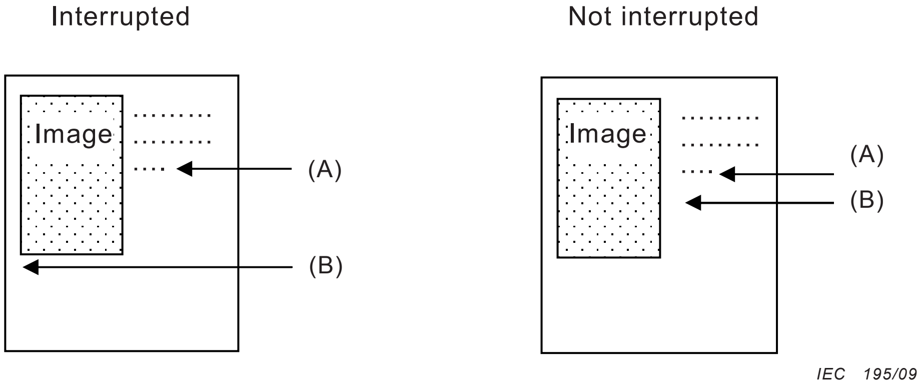


Figure A.5 – Flowing text interrupted by line break tag

This tag can use the following parameter. When no parameters are declared, that is when the relevant parameter data offset field of the block control information (see A.4.3.4) is set to 0xFFFF, text flowing around pictures is not interrupted.

clear When the text is flowing around a picture, this flag determines whether to interrupt it or not. As of publication of this standard, only the following value can be used.

all Interrupts text flowing around a picture, regardless of whether the picture is at the beginning or the end of the line.

This tag's parameter is stored in the block control information as described in the Table 14.

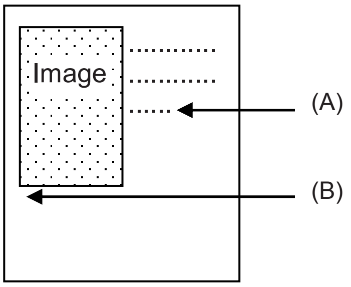
Table A.14 – Parameter of a line break tag stored in the block control information

Type / number of bytes	Bit (MSB)	Content	Notes
Cxmdf_char		clear	The value of the clear parameter. Only the following value can be used. 0x01: all

A.5.4 Horizontal line tag (tag number: 2)

This tag draws a line along the text's base line (horizontal with horizontal text, vertical with vertical text). The line must be drawn on an empty text line, fill the whole line's length, and be placed at mid height. The line length is not necessarily equal to the rendering area's length, as the paragraph tag (see A.5.2) may reduce the effective length of a line. The line's thickness is implementation dependant.

Note that if the tag occurs within text flowing around an image (see A.5.9), the line should be drawn after the image, not beside it. In Figure A.6, if the tag is located at point A, the line should be drawn on the text line marked B.



IEC 196/09

Figure A.6 – Horizontal line tag and an image

This tag has no parameter.

A.5.5 Font settings tag (tag number: 3)

This tag determines various font settings, applied as far as the text font tag or the end of the flow.

This tag has the following parameters.

bold — Determines whether the text should be displayed in bold face or not. The following values can be used. If omitted, the current status, as set by the previous font tag remains unchanged.

- yes — Display in bold
- no — Do not display in bold
- default — Use the viewer software’s default setting

underline — Determines whether the text should be underlined or not. If omitted, the current status, as set by the previous font tag, remains unchanged.

- yes — Underline the text
- no — Do not underline the text
- default — Use the viewer software’s default setting

color — Determines the color to be used to display the text. The following values can be used. If omitted, the current status, as set by the previous font tag, remains unchanged.

- Monochrome — A value from 0 to 255 represents the gray level.
- Color — R,G,B values, each between 0 and 255
- default — User the flow’s default setting.

The parameters of this tag are stored in the block control information as described in Table A.15.

Table A.15 – Parameters of a font settings tag stored in the block control information

Type / number of bytes	Bit (MSB)	Content	Notes
Cxmdf_char	7-6	bold	Records the value of the bold parameter. The following values can be used. 00: omitted 01: yes 10: no 11: default
	5-4	underline	Records the value of the underline parameter. The following values can be used. 00: omitted 01: yes 10: no 11: default
	3-2	Color mode flag	Determines if and how the font color is stored. The following values can be used. 00: omitted 01: Set, monochrome 10: Set, color 11: default
	1-0	Unused	Always set to 0.
Cxmdf_char/0		Font color info 1	Stores the R component of the font color if the color mode flag is set to 1-0, or the grayscale level if they are set to 0-1.
Cxmdf_char/0		Font color info 2	Stores the G component of the font color if the color mode flag is set to 1-0. Omitted for other values.
Cxmdf_char/0		Font color info 3	Stores the G component of the font color if the color mode flag is set to 1-0. Omitted for other values.

A.5.6 Ruby tag (tag number: 4)

This tag inserts a ruby character string. Ruby is an often used feature in Asian and particularly Japanese typography. It consists in a short run of text display above the base text, mostly used as a pronunciation guide for ideographic characters. As far as the available fonts permit it, ruby text is usually displayed in smaller characters than the base text it annotates. How line spacing is handled to open enough space to display ruby depends on the viewer. It should also be noted that, as ruby is fundamentally a reading aid and some people may not want or need it, some viewers may have an option to disable ruby displaying altogether. The number of base characters covered is indicated by the scope parameter. See below for an example of ruby.

Example:

なつめそうせき

夏目漱石

This tag has the following parameters.

scope — The length, in bytes, from the tags point of insertion in the body file, of the string that will be used as the base of the ruby. Cannot be omitted.

ruby — Ruby itself, stored as a standard character string. Cannot be omitted.

The parameters of this tag are stored in the block control information as described in Table A.16.

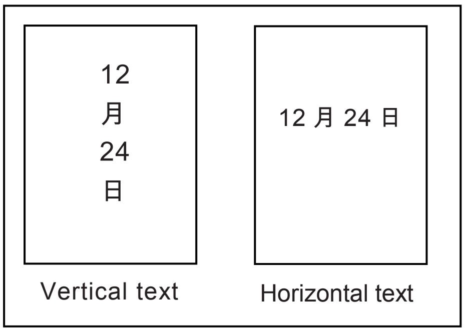
Table A.16 – Parameters of a ruby tag stored in the block control information

Type / number of bytes	Bit (MSB)	Content	Notes
Cxmdf_char		Scope	Stores the value of the scope parameter.
Cxmdf_string		Ruby	Stores the value of the ruby parameter.

A.5.7 Horizontal tag (tag number: 5)

This tag marks a portion of text that must be displayed horizontally, even if the general direction of the text is vertical. The number of base characters covered is indicated by the scope parameter. Note, however, that it may only cover a maximum of 2 characters, and that these must be half width characters (within the 0x20 to 0x7E range of Unicode). When the general direction of the text is horizontal, this tag should be ignored.

Example:



IEC 197/09

Figure A.7 – Effect of horizontal tag

This tag has the following parameter.

scope Determines the length in bytes, of the string to be cover by the horizontal tag. Cannot be omitted.

The parameter of this tag is stored in the block control information as described in Table A.17.

Table A.17 – Parameter stored in the block control information

Type / number of bytes	Bit (MSB)	Content	Notes
Cxmdf_char		Scope	Stores the value of the scope parameter.

A.5.8 External character tag (tag number: 6)

This tag is used to display an external character. It is used to replace a character or a sequence of characters by a picture, usually to represent characters not available in the font. The length of the string to be replaced is determined by the scope parameter. However, if the images specified by the alt_img or alt_vimg are not available (not downloaded, for example), the viewer can display the original string.

If the images specified in alt_img and alt_vimg do not match the font size, they should be resized accordingly, while preserving the aspect ratio.

This tag has the following parameters.

alt_img Records the image number of the picture to use as an external character. Based on this number, the viewer software will retrieve the image file (see A.3.3). When only this parameter (and not alt_vimg) is set, the image is used for both vertical and horizontal text; otherwise, it is used only for horizontal text. As of publication of this standard, only the PBM image format may be used. These images can only be black and white. The black pixels must be rendered with the same color as the font color, while the white pixels use the background color.

alt_vimg Records the image number of the picture to use as an external character when the text is displayed vertically. Based on this number, the viewer software will retrieve the image file (see A.3.3). When alt_img is set too, use the picture specified by alt_img when displaying the text horizontally. If alt_img is not set, do not replace the text when displaying horizontally. As of publication of this standard, only the PBM image format can be used, and is to be interpreted as described in the alt_img explanation.

scope Records the length, in bytes, of the character string to replace with an external character image. Cannot be omitted.

As said above, the PBM format is used to store external character images. The specification of the PBM format can be found at http: //netpbm.sourceforge.net/doc/pbm.html. Note, however, that it can only be used with the following restrictions.

- The magic number identifying the file type can only be P4.
- The height an width of the image must be one or two characters (from 1 to 99).
- The comment starting with a # character is not allowed.
- Only line feed (0x0A) can be used as a white space.

```
Example:
  P4 + LF(0x0A) + 12 + LF(0x0A) + 12 + LF(0x0A)
  ↑       ↑        ↑
magic number    width     Height
```

The parameters of this tag are stored in the block control information as described in Table A.18.

Table A.18 – Parameters stored in the block control information

Type / number of bytes	Bit (MSB)	Content	Notes
Cxmdf_char	7	alt_img parameter presence flag	Determines whether the alt_img parameter is stored or not. 0: Not stored 1: Stored
	6	alt_vimg parameter presence flag	Determines whether the alt_vimg parameter is stored or not. 0: Not stored 1: Stored
	5-0	Unused	Always set to 0.
Cxmdf_char		scope	Stores the value of the scope parameter.
Cxmdf_short/ 0		alt_img	Stores the value of the alt_img parameter. Only stored if the alt_img parameter presence flag is set to 1.
Cxmdf_short/ 0		alt_vimg	Stores the value of the alt_vimg parameter. Only stored if the alt_vimg parameter presence flag is set to 1.

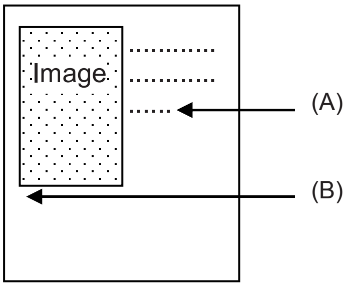
A.5.9 Image tag (tag number: 7)

This tag inserts an image right before the character pointed at by the tag's starting offset. Line breaks are inserted immediately after and before the image, except in the following cases.

- There is already a line break right before the image.

- The character immediately preceding the image is at the end of the line, and therefore the image would begin on a new line even without an additional line break.
- Even without adding a line break right after the image, the next element would be on the beginning of a line.

Note also, that when an image is inserted inside text that is flowing around another image, the flowing should be interrupted, and the new image should be inserted on the line following the one which was flowed around. In other words, if an image tag is at point A, the image should be displayed at point B, as indicated in Figure A.8..



IEC 198/09

Figure A.8 – Text and image tag

This tag has the following parameters.

src The image number of the picture to use. Based on this number, the viewer software will retrieve the image file (see A.3.3). Cannot be omitted. Only Jpeg can be used.

align Determines whether the text should flow around the image, and how. The following values can be used.

no The text does not flow around the picture.

left The image is displayed and the head of the line (left for horizontal writing, top for vertical writing), and text is written beside it, on the line's end side (right for horizontal writing, or bottom for vertical writing). The picture should be shifted if it falls in the scope of a paragraph tag (see A.5.2) having a non null top parameter.

right The image is displayed and the end of the line (right for horizontal writing, bottom for vertical writing), and text is written beside it, on the line's head side (left for horizontal writing, or top for vertical writing). The picture should be shifted if it falls in the scope of a paragraph tag (see A.5.2) having a non null bottom parameter.

When the image's align attribute is either set to left or right, and the image is in the scope of a paragraph tag (see A.5.2), the align parameter of the paragraph is ignored, and the image is placed according to its own align parameter. On the other hand the text flowing around the picture must obey the align parameter of the paragraph tag.

Following the description above, depending on the value of the align parameter, the text immediately following an image tag (which is inserted at point A) can be displayed in three different ways.

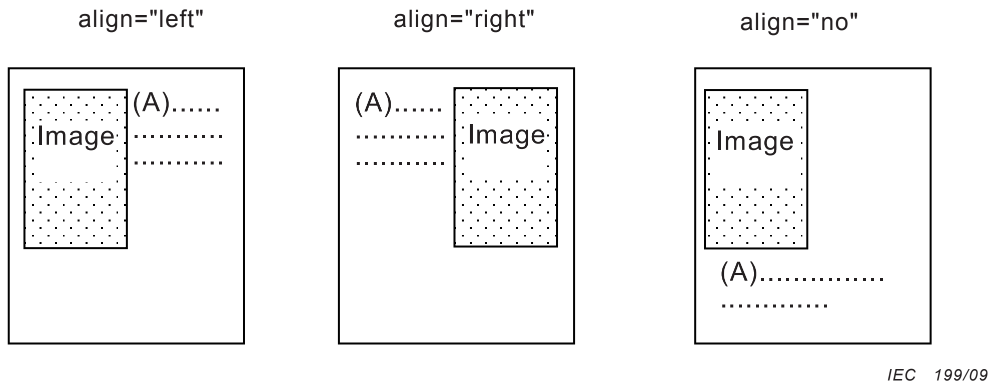


Figure A.9 – Effect of align parameter

The parameters of this tag are stored in the block control information as described in Table A.19.

Table A.19 – Parameters of an image tag stored in the block control information

Type / number of bytes	Bit (MSB)	Content	Notes
Cxmdf short		src	Records the value of the src parameter
Cxmdf char		align	Records the value of the align parameter. The following values can be used, 0x00: no 0x01: left 0x02: right

A.5.10 Mask tag (tag number: 8)

This tag puts a part of the text inside a mask. A mask is an area that can be covered to hide its content, or uncovered to show it. The length of the string to be masked is determined by the scope parameter. The string should be displayed differently from the rest of the text, so that it can be identified visually. When the string is clicked, it must switch from non-covered to covered state, and back if clicked again. The viewer does not have to remember the state of the mask when it is not in the display area. When the mask does not fit in the display area, and scrolling is used to see the continuation of the string, the viewer must remember the state of the mask. However, when switching between horizontal and vertical mode, the viewer does not have to remember the state of the mask, and may use the initial state to display it.

This tag has the following parameters.

initial flag — Determines the initial status of the mask. Cannot be omitted. The following values can be used.
  on — The mask initially covers its content.
  off — The mask initially lets its content visible.
scope — Records, in bytes, the length of the string to be covered by the mask. Cannot be omitted.

The parameters of this tag are stored in the block control information as described in Table A.20.

Table A.20 – Parameters of mask tag stored in the block control information

Type / number of bytes	Bit (MSB)	Content	Notes
Cxmdf_char	7	initial_flag	Records, the value of the initial_flag parameter. The following values can be used. 0: on 1: off
	6-0	unused	Always set to 0.
Cxmdf_char		scope	Stores the value of the scope parameter.

A.5.11 Link jump tag (tag number: 9)

This tag is used to define a hyperlink within the document or to another related C-XMDF document. The text of the link is the string whose length is specified by the scope parameter. The string should be displayed differently from the rest of the text, so that it can be identified visually. When the string is clicked, the viewer software should jump to the point indicated by the tag's parameters.

This tag has the following parameters.

link — Records the number of the flow to jump to. Based on this number, the viewer software can retrieve the appropriate flow body and control file (see A.3.3). The jump destination can only be in the same document as the source. When this parameter is set, the part_id parameter is ignored.

part_id — Defines the target of the jump as being in another part of the document, as some documents can have multiple parts and be made of several separate e-books. It cannot refer to the document currently being viewed. The document pointed by this parameter should be displayed from the beginning of its first flow.

scope — Records, in bytes, the length of the string to be used for the link's text. Cannot be omitted.

position — Defines a position in the target flow the viewer should jump at. In the case of text flows, it is an offset in bytes, 0 being the beginning of the flow, whereas in the case of cell flows, it is the number of the cell to display, 0 being the first cell. Can only be stored if the link parameter is stored too. Defaults to 0 when omitted.

The parameters of this tag are stored in the block control information as described in Table A.21.

Table A.21 – Parameters of link jump tag stored in the block control information

Type / number of bytes	Bit (MSB)	Content	Notes
Cxmdf_char	7	link parameter presence flag	Determines whether the link parameter is stored or not. 0: Not stored 1: Stored
	6	part_id parameter presence flag	Determines whether the part_id parameter is stored or not. 0: Not stored 1: Stored
	5	position parameter presence flag	Determines whether the position parameter is stored or not. 0: Not stored 1: Stored
	3-0	unused	Always set to 0.
Cxmdf_char		scope	Records the value of the scope parameter.
Cxmdf_short/ 0		link	Records the value of the link parameter. Only stored when Link parameter presence flag is set to 1.
Cxmdf_short/ 0		part_id	Records the value of the part_id parameter. Only stored when part_id parameter presence flag is set to 1.
Cxmdf_short/ 0		position	Records the value of the position parameter. Only stored when position parameter presence flag is set to 1.

A.5.12 URL jump tag (tag number: 10)

This tag is used to define a hyperlink to an URL. The text of the link is the string whose length is specified by the scope parameter. The string should be displayed differently from the rest of the text, so that it can be identified visually. When the string is clicked, the viewer software should jump to the designated URL.

This tag has the following parameters.

url	Records the URL of the web site targeted by the link. Only http:// and https:// URL can be used. Cannot be omitted.
scope	Records, in bytes, the length of the string to be used for the link's text. Cannot be omitted.

The parameters of this tag are stored in the block control information as described in Table A22.

Table A.22 – Parameters of a URL jump tag stored in the block control information

Type / number of bytes	Bit (MSB)	Content	Notes
Cxmdf_char		scope	Records the value of the scope parameter.
Cxmdf_string		url	Records the value of the url parameter.

A.5.13 Mailer launch tag (tag number: 11)

This tag is used to create a link launching the mail client. The text of the link is the string whose length is specified by the scope parameter. The string should be displayed differently from the rest of the text, so that it can be identified visually. When the string is clicked, the mail client software should be launched, with creating a new mail, with its TO field, and optionally its body, initialized to the values specified in the tag's parameters.

This tag has the following parameters.

to	Specifies the address to use to initialize the TO field of the mail created when launching the mail client. Only one address can be used. Cannot be omitted.
body	Specifies the text to use to initialize the body of the mail created when launching the mail client. Stored as a standard character string. Line breaks (0x0d, 0x0a, 0x0d0a), spaces (0x20) and tabulations (0x09) can be used. Can be omitted.
scope	Records, in bytes, the length of the string to be used for the link's text. Cannot be omitted.

The parameters of this tag are stored in the block control information as described in Table A.23.

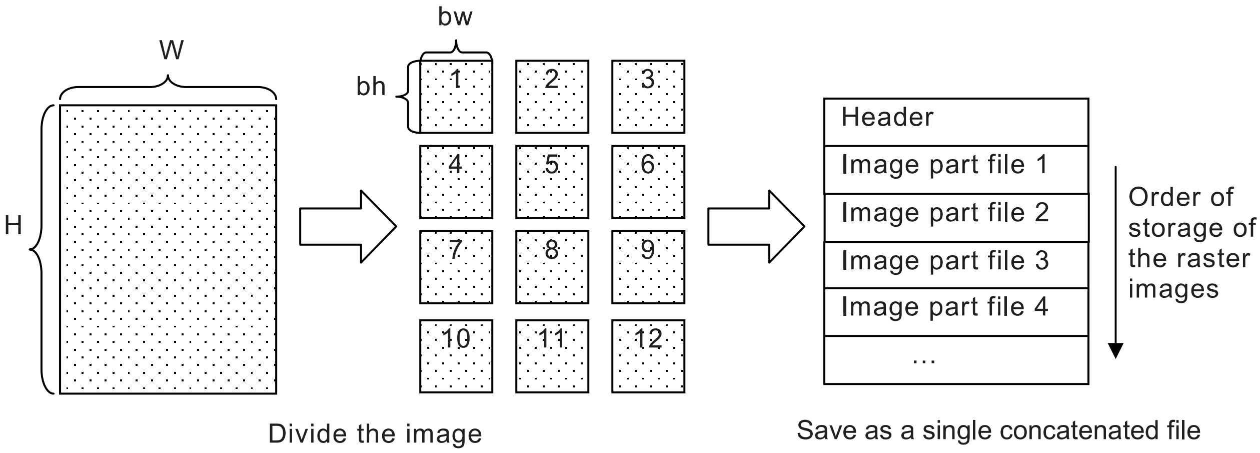
Table A.23 – Parameters of a mailer launch tag stored in the block control information

Type / number of bytes	Bit (MSB)	Content	Notes
Cxmdf_char	7	body parameter presence flag	Determines whether the body parameter is stored or not. 0: Not stored 1: Stored
	6-0	unused	Always set to 0
Cxmdf_char		scope	Records the value of the scope parameter.
Cxmdf_string		to	Records the value of the to parameter.
Cxmdf_string/0		body	Records the value of the body parameter. Only stored if the body parameter presence flag is set to 1.

A.6 Multi-image group

A.6.1 MIG (Multi Image Group) split image file format

As the images composing a cell flow are generally quite large, some low end terminals may face problems processing them, such as lack of memory, or exceedingly long processing time. To counter these problems, the MIG image format was designed. The original image is split in parts of defined equal size, which are then stored in a MIG image container. An image of W×H dimensions is split in bw×bh sized smaller pieces, which are then store together as a single file, as illustrated by Figure A.10.



IEC 200/09

Figure A.10 – Image splitting

Depending on the size of the original image, and the size of the parts, the parts situated on the right and lower side (3, 6, 9 10, 11 and 12) may not be the same size as other parts. For example if W = 1 024, H = 1 600, bw = 100 and bh = 160, the right border parts will be 24×160. Nevertheless, these smaller parts are to be saved as is. Note also, that the MIG container format stores the image body and the image header of each part independently, and that different parts with identical headers do not need to store the header several times, and can share one.

Table A.24 describes the data structure of the MIG format.

Table A.24 – Data structure of the MIG format

Type / number of bytes	Content	Explanation
3	MIG ID number	Identifies the file as a MIG image. It is defined as the three characters: MIG.
4	MIG version	Records the MIG version number. As of publication of this standard, it is 1.10, stored as an ASCII string.
Cxmdf_char	Image encoding type	Records the file format of the embedded image parts. As of publication of this standard, the following values can be used. 0x00 : JPEG 0x01 : PNG 0x02 : GIF
Cxmdf_short	Original picture width	Records the width of the original (unsplit) picture, in pixels.
Cxmdf_short	Original picture height	Records the height of the original (unsplit) picture, in pixels.
Cxmdf_short	Part image width	Records the width of image parts
Cxmdf_short	Part image height	Records the height of image parts
Cxmdf_char	Number of shared header	Records the number of image parts headers stored in the MIG.
Cxmdf_int Cxmdf_int	Shared header 1 start offset Shared header 1 data size	Stores the offset in this file of the beginning of the first shared header Stores the size, in bytes, of the first shared header.
Cxmdf_int Cxmdf_int	Shared header 2 start offset Shared header 2 data size	Stores the offset in this file of the beginning of the second shared header Stores the size, in bytes, of the second shared header.
…	…	Repeated for all shared headers.
Cxmdf_short	Number of image parts	Records the total number of image parts.
Cxmdf_char Cxmdf_int Cxmdf_int	Part image 1 shared header number Part image 1 start offset Part image 1 data size	Records the number of the shared header to use for the first image. Header numbers start from 0. Stores the offset in this file of the beginning of the first image part's data. Stores the size, in bytes, of the first image part's data.
Cxmdf_char Cxmdf_int Cxmdf_int	Part image 1 shared header number Part image 1 start offset Part image 1 data size	
…	…	Repeated for all part images.
1	Unused	Set to 0x00 as of publication of this standard.
n	Shared header 1 data	Stores the first shared header's data.
n	Shared header 2 data	Stores the second shared header's data.
	…	
n	Sub-image 1 data	Stores the main data, the body of the first image part. This is stored in standard jpeg of gif format, stripped of the header which has been stored independently, as a shared header, whose number is recorded in the part image 1 shared header number field.
n	Sub-image 2 data	Stores the main data, the body of the second image part.
	…	Repeated for all part images.

A.6.2 Supported GIF image files

2 of GIF format variations are supported by this standard as shown in Table A.25. The same limitations apply to the GIF images embedded in MIG images.

Table A.25 – Gif image support

GIF format	Support
GIF 87	No
GIF 87a	Yes
GIF 89a	Yes
Interlaced GIF	No
Transparent GIF	No
Animated GIF	No

A.6.3 Tag nesting specification

Table A.26 specifies which elements of a text flow of a compact XMDF document can be nested.

Table A.26 – Possible tag nesting

Parent tag		Child element												
Tag name (tag number)	**Position of child elements**	**Paragraph**	**Line break**	**Horizontal line**	**Font**	**Ruby**	**Horizontal**	**External character**	**Image**	**Mask**	**Link jump**	**URL**	**Mailer launch**	
Paragraph (0)	Until the next paragraph		○	○	○	○	○	○	○	○	○	○	○	Standard character string (text body)
Line break(1)	N/A													
Horizontal line(2)	N/A													
Font settings(3)	Until the next font setting tag	○	○	○		○	○	○	○	○	○	○	○	Standard character string (text body)
Ruby(4)	Within the area define by the scope parameter.	×	×	×	×	×	×	○	×	×	×	×	×	Standard character string (base of the ruby)
Horizontal (5)	Within the area define by the scope parameter.	×	×	×	×	×	×	×	×	×	×	×	×	Up to two half width characters (text body)
External character (6)	Within the area define by the scope parameter.	×	×	×	×	×	×	×	×	×	×	×	×	Standard character string (text to be replaced)
Image(7)	N/A													
Mask(8)	Within the area define by the scope parameter.	×	×	×	×	○	○	○	○	×	×	×	×	Standard character string (text body)
Link jump(9)	Within the area define by the scope parameter.	×	×	×	×	○	○	○	○	×	×	×	×	Standard character string (link text)
URL jump (10)	Within the area define by the scope parameter.	×	×	×	×	○	○	○	○	×	×	×	×	Standard character string (link text)
Mailer launch(11)	Within the area define by the scope parameter.	×	×	×	×	○	○	○	○	×	×	×	×	Standard character string (link text)

A.6.4 Conformance levels

Software viewer implementing the compact XMDF format may be classified into three levels of conformance, depending on the features they implement. The three levels are defined in Table A.27, Table A.28, and Table A.29.

Table A.27 – Conformance levels

Levels	Capabilities	Comments
Minimum	– displays text data with new lines	Example of intended applications: Simple digital signage
Medium	– all the capabilities of minimum conformance level – displays text data with interactive and multimedia capabilities	
Rich	– all the capabilities of medium conformance level – displays comic data	Example of intended applications: Comic viewer on mobile phone handsets NOTE This includes all the capabilities of C-XMDF format as described in Annex A.

Table A.28 – Tags eligible for each conformance level

Tags	Conformance level		
	Minimum	Medium	Rich
Paragraph		o	○
Line break(1)	o	o	o
Horizontal line(2)		o	o
Font settings(3)		o	o
Ruby(4)		o	o
Horizontal(5)		o	o
External character(6)		o	o
Image(7)		o	o
Mask(8)		o	o
Link jump(9)		o	o
URL jump (10)		o	o
Mailer launch(11)		o	o

Table A.29 – Files eligible for each conformance level

File Type	Conformance level		
	Minimum	Medium	Rich
Root file	o	o	o
Text flow body file	o	o	o
Text flow control file	o	o	o
Cell flow control file			o
Index body file		o	o
Index control file		o	o
Picture file		o	o
Sound file		o	o

See A.3 for the data structure of each file type.

A.7 Localization and internationalization

Due to the considerations for different languages and environments, this International Standard stipulates several items which are to be modified for localization.

A.7.1 Character sets

As shown in Table A.6 (standard character set flag), values 0x00-0x05 are already allocated for character sets. It is to be understood that new values should be specified for them starting from 0x06 when extending this standard to use with other character sets.

A.7.2 In-line note character

Some languages and applications may have different needs for in-line note characters. In such cases, the set of in-line note characters should be decided accordingly.

A.7.3 End-of-line rules

Some languages have generally accepted rules on ends of lines, for example characters that cannot appear on line ends and rules on how to proceed when it happens.

However, C-XMDF specification regards such rules as a viewer-dependent issue and does not state any explicit requirements.

A.7.4 Image files

As shown in Table A.3 (picture file), 4 image file formats (viz.JPEG/PBM/MIG/GIF) are allowed in this standard. It is to be understood that flag values and other treatments will be defined for image files when the need arises to use other image formats. Values should be defined in line with what is specified in Table A.4 (Picture file), Table A.5 (Picture file) and Table A.6 (Encoding format of image data *n*).

A.7.5 Audio files

As shown in Table A.3 (sound file), 3 audio file formats (viz.MFi/SMF/SP-MIDI) are allowed in this standard. It is to be understood flag values and other treatments will be defined for sound files when the need arises to use other audio formats. Values should be defined in line with what is specified in Table A.4 (sound file), Table A.5 (sound file) and Table A.6 (encoding format of sound data *n*).

Bibliography

BPM format, see http: //netpbm.sourceforge.net/doc/pbm.html

This page deliberately left blank

British Standards Institution (BSI)

BSI is the national body responsible for preparing British Standards and other standards-related publications, information and services.

BSI is incorporated by Royal Charter. British Standards and other standardization products are published by BSI Standards Limited.

About us

We bring together business, industry, government, consumers, innovators and others to shape their combined experience and expertise into standards -based solutions.

The knowledge embodied in our standards has been carefully assembled in a dependable format and refined through our open consultation process. Organizations of all sizes and across all sectors choose standards to help them achieve their goals.

Information on standards

We can provide you with the knowledge that your organization needs to succeed. Find out more about British Standards by visiting our website at bsigroup.com/standards or contacting our Customer Services team or Knowledge Centre.

Buying standards

You can buy and download PDF versions of BSI publications, including British and adopted European and international standards, through our website at bsigroup.com/shop, where hard copies can also be purchased.

If you need international and foreign standards from other Standards Development Organizations, hard copies can be ordered from our Customer Services team.

Subscriptions

Our range of subscription services are designed to make using standards easier for you. For further information on our subscription products go to bsigroup.com/subscriptions.

With **British Standards Online (BSOL)** you'll have instant access to over 55,000 British and adopted European and international standards from your desktop. It's available 24/7 and is refreshed daily so you'll always be up to date.

You can keep in touch with standards developments and receive substantial discounts on the purchase price of standards, both in single copy and subscription format, by becoming a **BSI Subscribing Member**.

PLUS is an updating service exclusive to BSI Subscribing Members. You will automatically receive the latest hard copy of your standards when they're revised or replaced.

To find out more about becoming a BSI Subscribing Member and the benefits of membership, please visit bsigroup.com/shop.

With a **Multi-User Network Licence (MUNL)** you are able to host standards publications on your intranet. Licences can cover as few or as many users as you wish. With updates supplied as soon as they're available, you can be sure your documentation is current. For further information, email bsmusales@bsigroup.com.

Revisions

Our British Standards and other publications are updated by amendment or revision.

We continually improve the quality of our products and services to benefit your business. If you find an inaccuracy or ambiguity within a British Standard or other BSI publication please inform the Knowledge Centre.

Copyright

All the data, software and documentation set out in all British Standards and other BSI publications are the property of and copyrighted by BSI, or some person or entity that owns copyright in the information used (such as the international standardization bodies) and has formally licensed such information to BSI for commercial publication and use. Except as permitted under the Copyright, Designs and Patents Act 1988 no extract may be reproduced, stored in a retrieval system or transmitted in any form or by any means – electronic, photocopying, recording or otherwise – without prior written permission from BSI. Details and advice can be obtained from the Copyright & Licensing Department.

Useful Contacts:

Customer Services
Tel: +44 845 086 9001
Email (orders): orders@bsigroup.com
Email (enquiries): cservices@bsigroup.com

Subscriptions
Tel: +44 845 086 9001
Email: subscriptions@bsigroup.com

Knowledge Centre
Tel: +44 20 8996 7004
Email: knowledgecentre@bsigroup.com

Copyright & Licensing
Tel: +44 20 8996 7070
Email: copyright@bsigroup.com

BSI Group Headquarters

389 Chiswick High Road London W4 4AL UK

bsi.

...making excellence a habit.™