MNG-VLC (Multiple-image Network Graphics—Very Low Complexity) Format Version 1.0

For list of authors, see Credits (Chapter 19).

Status of this Memo

This document is a specification by the PNG development group. It has been approved by a vote of the group. Future technical changes will require formal approval by a vote of the group. It is the intent of the group to maintain backward compatibility if possible.

Comments on this document can be sent to the MNG specification maintainers at one of the following addresses:

- mng-list@ccrc.wustl.edu
- png-group@w3.org
- png-info@uunet.uu.net

Distribution of this memo is unlimited.

At present, the latest version of this document is available on the World Wide Web from

    ftp://swrinde.nde.swri.edu/pub/mng/documents/.

In the case of any discrepancy between this extract and the full MNG specification, the full MNG specification shall take precedence.

Abstract

This document presents the MNG-VLC (Multiple-image Network Graphics, Very Low Complexity) format, which is a proper subset of the MNG (Multiple-image Network Graphics) format.

MNG is a multiple-image member of the PNG (Portable Network Graphics) format family. It can contain animations, slide shows, or complex still frames, comprised of multiple PNG single-image datastreams.

The MNG-VLC format uses the same chunk structure that is defined in the PNG specification and shares other features of the PNG format. Any MNG-VLC decoder must be able to decode valid PNG datastreams.

A MNG-VLC frame normally contains a two-dimensional image or a two-dimensional layout of smaller images. It could also contain three-dimensional "voxel" data arranged as a series of two-dimensional planes (or tomographic slices), each plane being represented by a PNG datastream.

This document includes examples that demonstrate various capabilities of MNG-VLC.

2

Contents

1   Introduction

This document presents a very-low-complexity version (MNG-VLC, which is a proper subset) of the MNG (Multiple-image Network Graphics) format.

Note: This specification depends on the PNG (Portable Network Graphics) [PNG] and, for MNG-VLC applications that are enhanced with JNG support, the JNG (JPEG Network Graphics) specifications. The PNG specification is available at the PNG web site,

        http://www.libpng.org/pub/png/

and the JNG (JPEG Network Graphics) specification and the full MNG specification are available at the MNG web site,

        http://www.libpng.org/pub/mng/

MNG is a multiple-image member of the PNG format family that can contain

- animations,
- slide shows, or
- complex still frames,

comprised of multiple PNG single-image datastreams.

Like PNG, a MNG datastream consists of an 8-byte signature, followed by a series of chunks. It begins with the MHDR chunk and ends with the MEND chunk. Each chunk consists of a 4-byte data length field, a 4-byte chunk type code (e.g., "MHDR"), data (unless the length is zero), and a CRC (cyclical redundancy check value).

A MNG-VLC datastream describes a sequence of zero or more single frames, each of which is composed of a single embedded image, or one single frame that can be composed of zero or more embedded images.

The embedded images can be PNG or JNG datastreams. MNG-VLC datastreams do not contain JNG (JPEG Network Graphics) datastreams, which are allowed in full MNG datastreams, but MNG-VLC applications can be enhanced to recognize and process JNG datastreams as well.

A typical MNG-VLC datastream consists of:

- The 8-byte MNG signature.
- The MHDR chunk.
- Layer definitions.
  - An embedded potentially visible image, described by PNG or JNG datastreams (a foreground layer).
  - The background (a background layer).
- The MEND chunk.

MNG is fundamentally declarative; it describes the elements that go into an individual frame. It is up to the decoder to work out an efficient way of making the screen match the desired composition whenever a nonzero interframe delay occurs. Simple decoders can handle it as if it were procedural, compositing the images into the frame buffer in the order that they appear, but efficient decoders might do something different, as long as the final appearance of the frame is the same.

MNG is pronounced "Ming."

When a MNG datastream is stored in a file, it is recommended that ".mng" be used as the file suffix. In network applications, the Media Type "video/x-mng" can be used. Registration of the media type "video/ mng" might be pursued at some future date.

The MNG datastream begins with an 8-byte signature containing

```
   138  77  78  71  13  10  26  10   (decimal)
     8a  4d  4e  47  0d  0a  1a  0a   (hexadecimal)
  \212   M   N   G  \r  \n \032 \n   (ASCII C notation)
```

which is similar to the PNG signature with "\212 M N G" instead of "\211 P N G" in bytes 0–3.

Chunk structure (length, name, data, CRC) and the chunk-naming system are identical to those defined in the PNG specification. As in PNG, all integers that require more than one byte must be in network byte order.

The chunk copying rules for MNG employ the same mechanism as PNG, but with rules that are explained more fully in the full MNG specification.

Note that decoders are not required to follow any decoding models described in this specification nor to follow the instructions in this specification, as long as they produce results identical to those that could be produced by a decoder that did use this model and did follow the instructions.

Each chunk of the MNG datastream or of any embedded object is an independent entity, i.e., no chunk is ever enclosed in the data segment of another chunk.

MNG-VLC decoders are required to recognize and decode independent PNG datastreams, and any MNG-VLC decoder that has been enhanced to include JNG support is required to recognize and decode independent JNG datastreams.

Because the embedded objects making up a MNG are normally in PNG format, MNG shares the good features of PNG:

- It is unencumbered by patents.
- It is streamable.
- It has excellent, lossless compression.
- It stores up to four channels (red, green, blue, alpha), with up to 16 bits per channel.
- It provides both binary and alpha-channel transparency.
- It provides platform-independent rendition of colors by inclusion of gamma and chromaticity information.

- It provides early detection of common file transmission errors and robust detection of file corruption.
- Single-image GIF files can be losslessly converted to PNG.
- It is complementary to JPEG and does not attempt to replace JPEG for lossy storage of images (however, JNG-enhanced MNG-VLC can accommodate JPEG-encoded images that are encoded in the PNG-like JNG format.

In addition:

- It provides animation with a constant framing rate.
- It allows composition of frames containing multiple images.
- Using JPEG compression together with a magnification factor, it can achieve 1000:1 and higher lossy compression of Megapixel truecolor images. While some detail is lost, such highly-compressed images are useful as full-scale previews and for layout work.
- Multiple-image GIF files (except for those using a variable framing rate, the "restore-to-background" or the "restore-to-previous" disposal method) can be losslessly converted to MNG-VLC.

2 Terminology

See also the glossary in the PNG specification and the "terminology" section of the full MNG specification.

requirement levels

> The words "MUST", "MUST NOT", "REQUIRED", "SHOULD", "SHOULD NOT", "RECOMMENDED", and "OPTIONAL" in this document, which are to be interpreted as described in RFC-2119. The word "CAN" is equivalent to the word "MAY" as described therein. "NOT ALLOWED" and "NOT PERMITTED" describe conditions that "MUST NOT" occur. "ALLOWED" and "PERMITTED" describe conditions that "CAN" occur.

embedded image

> An image that appears in-line in a MNG datastream.

frame

> A composition of zero or more layers that have zero interframe delay time followed by a layer with a specified nonzero delay time or by the MEND chunk. A frame is to be displayed as a still picture or as part of a sequence of still images or an animation. An animation would ideally appear to a perfect observer (with an inhumanly fast visual system) as a sequence of still pictures.

> In MNG-VLC datastreams, each frame (except for the first, which also includes the background layer) contains a single layer, unless the framing rate (from the MHDR `ticks_per_second` field) is zero. When the framing rate is zero, the entire datastream describes a single frame.

> When the layers of a frame do not cover the entire area defined by the width and height fields from the MHDR chunk, the layers are composited over the previous frame to obtain the new frame.

When the frame includes the background layer, and the background layer is transparent, the transparent background is composited against the outside world and the subsequent layers are composited against the result to obtain the new frame.

frame origin

The upper left corner of the output device (frame buffer, screen, window, page, etc.) where the pixels are to be displayed. In MNG-VLC, all images must be placed with the image's upper left corner at the frame origin.

layer

One of

- A visible embedded image.
- The background that is displayed before the first image in the entire datastream is displayed. Its contents can be defined by the application or by the BACK chunk.

An embedded visible PNG or JNG datastream generates a single layer, even though it might be interlaced or progressive.

MNG-VLC

A very-low-complexity subset of MNG that does not use stored objects, variable framing rates, location of images at positions other than (0,0), or other complex features. The "simplicity profile" in the MHDR chunk must meet certain requirements (see the MHDR chunk specification below, Paragraph 4.1.1).

nullify

To nullify a chunk is to undo its effect, restoring the datastream to the condition it would have had if the chunk being nullified had never appeared.

visible image

Actually drawn on a display. If an object is visible, a person looking at the display can see it.

3 Objects

Omitted.

4 MNG Chunks

This chapter describes chunks that can appear at the top level of a MNG datastream.

Chunk structure (length, name, data, CRC) and the chunk-naming system are identical to those defined in the PNG specification [PNG]. As in PNG, all integers that require more than one byte must be in network byte order.

Unlike PNG, fields can be omitted from some MNG chunks with a default value if omitted. This is permitted only when explicitly stated in the specification for the particular chunk. If a field is omitted, all the subsequent fields in the chunk must also be omitted and the chunk length must be shortened accordingly.

4.1   Critical MNG control chunks

This section describes critical MNG control chunks that MNG-VLC-compliant decoders must recognize and process. "Processing" a chunk sometimes can consist of simply recognizing it and ignoring it. Some chunks have been declared to be critical only to prevent them from being relocated by MNG editors.

4.1.1   MHDR MNG datastream header

The MHDR chunk is always first in all MNG datastreams except for those that consist of a single PNG or JNG datastream with a PNG or JNG signature.

The MHDR chunk contains 28 bytes, none of which can be omitted:

```
Frame_width:          4 bytes (unsigned integer).
Frame_height:         4 bytes (unsigned integer).
Ticks_per_second:     4 bytes (unsigned integer).
Nominal_layer_count:  4 bytes (unsigned integer).
Nominal_frame_count:  4 bytes (unsigned integer).
Nominal_play_time:    4 bytes (unsigned integer).
Simplicity_profile:   4 bytes:(unsigned integer).
                        bit 0: Profile Validity
                          1: Absence of certain features is specified by
                             the remaining bits of the simplicity profile.
                          (must be 1 in MNG-VLC datastreams)
                        bit 1: Simple MNG features
                          0: Simple MNG features are absent.
                          (must be 0 in MNG-VLC datastreams)
                        bit 2: Complex MNG features
                          0: Complex MNG features are absent.
                          (must be 0 in MNG-VLC datastreams)
                        bit 3: Internal transparency
                          0: Transparency is absent or can be ignored.
                             All images in the datastream are opaque or
                             can be rendered as opaque without affecting
                             the final appearance of any frame.
                          1: Transparency may be present.
                        bit 4: JNG
                          0: JNG and JDAA are absent.
```

```
                            1: JNG or JDAA may be present.
                            (must be 0 in MNG-VLC datastreams)
                         bit 5: Delta-PNG
                            0: Delta-PNG is absent.
                            (must be 0 in MNG-VLC datastreams)
                         bit 6: Validity flag for bits 7, 8, and 9
                            0: The absence of background transparency,
                               semitransparency, and stored object buffers
                               is unspecified; bits 7, 8, and 9 have no
                               meaning and must be 0.
                            1: The absence or possible presence of
                               background transparency is expressed by
                               bit 7, of semitransparency by bit 8, and of
                               stored object buffers by bit 9.
                         bit 7: Background transparency
                            0: Background transparency is absent (i.e., the
                               first layer fills the entire MNG frame with
                               opaque pixels).
                            1: Background transparency may be present.
                         bit 8: Semi-transparency
                            0: Semitransparency (i.e., an image with an
                               alpha channel that has values that are
                               neither 0 nor the maximum value) is absent.
                            1: Semitransparency may be present.
                            If bit 3 is zero this field has no meaning.
                         bit 9: Stored object buffers
                            0: Object buffers need not be stored.
                            (must be 0 in MNG-LC and MNG-VLC
                             datastreams)
                            If bit 2 is zero, this field has no meaning.
                         bits 10-15: Reserved bits
                            Reserved for public expansion.  Must be zero in
                            this version.
                         bits 16-30: Private bits
                            Available for private or experimental expansion.
                            Undefined in this version and can be ignored.
                         bit 31: Reserved bit. Must be zero.
```

Decoders can ignore the "informative" `nominal_frame_count`, `nominal_layer_count`, `nominal_play_time`, and `simplicity_profile` fields.

The `frame_width` and `frame_height` fields give the intended display size (measured in pixels) and provide clipping boundaries. (see Recommendations for encoders, below).

The `ticks_per_second` field gives the framing rate. It must be nonzero if the datastream contains a sequence of images. When the datastream contains exactly one frame, this field should be set to zero.

When `ticks_per_second` is nonzero, viewers should display the sequence of frames at the rate of one frame per tick.

If the frame count field contains a zero, the frame count is unspecified. If it is nonzero, it contains the number of frames that would be displayed, ignoring the TERM chunk. If the frame count is greater than $2^{31} - 1$, encoders should write $2^{31} - 1$, representing an infinite frame count. In MNG-VLC datastreams, the frame count is the same as the number of embedded images in the datastream (or one, the background layer, if there are no embedded images).

If the `nominal_layer_count` field contains a zero, the layer count is unspecified. If it is nonzero, it contains the number of layers (including the background layer) in the datastream, ignoring any effects of the TERM chunk. If the layer count is greater than $2^{31} - 1$, encoders should write $2^{31} - 1$, representing an infinite layer count. In MNG-VLC datastreams, the layer count is the number of embedded images, plus one (for the background layer).

If the `nominal_play_time` field contains a zero, the nominal play time is unspecified. Otherwise, it gives the play time, in ticks, when the file is displayed ignoring the TERM chunk. Authors who write this field should choose a value of `ticks_per_second` that will allow the nominal play time to be expressed in a four-bit integer. If the nominal play time is greater than $2^{31} - 1$ ticks, encoders should write $2^{31} - 1$, representing an infinite nominal play time. In MNG-VLC datastreams, the nominal play time is the same as the frame count, except when the `ticks_per_second` field is zero, in which case the nominal play time is also zero.

When bit 0 of the `simplicity_profile` field is zero, the simplicity (or complexity) of the MNG datastream is unspecified, and *all* bits of the simplicity profile must be zero. The simplicity profile must be nonzero in MNG-VLC datastreams.

If the simplicity profile is nonzero, it can be regarded as a 32-bit profile, with bit 0 (the least significant bit) being a "profile-validity" flag, bit 1 being a "simple MNG" flag, bit 2 being a "complex MNG" flag, bits 3, 7, and 8 being "transparency" flags, bit 4 being a "JNG" flag, bit 5 being a "Delta-PNG" flag, and bit 9 being a "stored object buffers" flag. Bit 6 is a "validity" flag for bits 7, 8, and 9, which were added at version 0.98 of this specification. These three flags mean nothing if bit 6 is zero.

If a bit is zero, the corresponding feature is guaranteed to be absent or if it is present there is no effect on the appearance of any frame if the feature is ignored. If a bit is one, the corresponding feature may be present in the MNG datastream.

Bits 10 through 15 of the simplicity profile are reserved for future MNG versions, and must be zero in this version.

Bits 16 through 30 are available for private test or experimental versions. The most significant bit (bit 31) must be zero.

"Transparency is absent or can be ignored" means that either the PNG tRNS chunk is not present and no PNG or JNG image has an alpha channel, or if they are present they have no effect on the final appearance of any frame and can be ignored (e.g., if the only transparency in a MNG datastream appears in a thumbnail that is never displayed in a frame, or is in some pixels that are overlaid by opaque pixels before being displayed, the transparency bit should be set to zero).

"Semitransparency is absent" means that if the PNG tRNS chunk is present or if any PNG or JNG image has an alpha channel, they only contain the values 0 and the maximum (opaque) value. It also means that

the JDAA chunk is not present. The "semitransparency" flag means nothing and must be 0 if bit 3 is 0 or bit 6 is 0.

"Background transparency is absent" means that the first layer of every segment fills the entire frame with opaque pixels, and that nothing following the first layer causes any frame to become transparent. Whatever is behind the first layer does not show through.

When "Background transparency" is present, the application is responsible for supplying a background color or image against which the MNG background layer is composited, and if the MNG is being displayed against a changing scene, the application should refresh the entire MNG frame against a new copy of the background layer whenever the application's background scene changes. The "background transparency" flag means nothing and must be 0 if bit 6 is 0. Note that bit 3 does not make any promises about background transparency.

The "stored object buffers" flag must be zero in MNG-VLC datastreams.

A MNG-VLC (i.e., a "very low-complexity MNG") datastream must have a simplicity profile with bit 0 equal to 1 and all other bits except possibly for bits 3, 6, 7, and 8 (transparency) equal to zero. If bit 4 (JNG) is 1, the datastream is a "MNG-VLC with JNG" datastream. It might contain a JNG datastream carrying an image or an alpha channel. MNG-VLC decoders are allowed to reject such datastreams unless they have been enhanced with JNG capability.

Encoders that write a nonzero simplicity profile should endeavor to be accurate, so that decoders that process it will not unnecessarily reject datastreams or avoid possible optimizations. For example, the simplicity profile 351 (0x15f) indicates that JNG, critical transparency, semitransparency, and at least one "complex" MNG feature are all present, but Delta-PNG, stored object buffers, and background transparency are not. This example would not qualify as a MNG-VLC datastream because a "complex" MNG feature might be present. If the simplicity profile promises that certain features are absent, but they are actually present in the MNG datastream, the datastream is invalid.

4.1.2 MEND End of MNG datastream

The MEND chunk's data length is zero. It signifies the end of a MNG datastream.

4.1.3 LOOP, ENDL Define a loop

The LOOP chunk can be ignored by MNG-VLC decoders, along with the ENDL chunk.

4.2 Critical MNG image defining chunks

The chunks described in this section create images and may cause them to be immediately displayed.

4.2.1 IHDR, PNG chunks, IEND

A PNG (Portable Network Graphics) datastream.

See the PNG specification [PNG] and the Extensions to the PNG Specification document [PNG-EXT] for the format of the PNG chunks.

The IHDR and IEND chunks and any chunks between them are written and decoded according to the PNG specification, except as extended in this section. These extensions do not apply to standalone PNG datastreams that have the PNG signature, but only to PNG datastreams that are embedded in a MNG datastream that begins with a MNG signature. Nor are they allowed in MNG-VLC datastreams.

- The PNG oFFs and pHYs chunks and any chunks in a future version of this specification that attempt to set the pixel dimensions or the drawing location must be ignored by MNG viewers and simply copied (according to the copying rules) by MNG editors.

- The PNG gIFg, gIFt, and gIFx chunks must be ignored by viewers and must be copied according to the copying rules by MNG editors.

4.2.2 JHDR, JNG chunks, IEND

A JNG (JPEG Network Graphics) datastream.

See the JNG specification for the format of the JNG datastream.

The JHDR and IEND chunks and any chunks between them are written and decoded according to the JNG specification.

The remaining discussion in the previous paragraph about PNG datastreams also applies to JNG datastreams.

MNG-VLC applications are not expected to process JNG datastreams unless they have been enhanced with JNG capability.

4.2.3 TERM Termination action

The TERM chunk suggests how the end of the MNG datastream should be handled, when a MEND chunk is found. It contains either a single byte or ten bytes:

```
Termination_action:        1 byte (unsigned integer)
                              0: Show the last frame indefinitely.
                              1: Cease displaying anything.
                              2: Show the first frame after the TERM chunk.
                              3: Repeat the sequence starting immediately
                                 after the TERM chunk and ending with the
                                 MEND chunk.
```

```
Action_after_iterations: 1 byte
                            0: Show the last frame indefinitely after
                               iteration_max iterations have been done.
                            1: Cease displaying anything.
                            2: Show the first frame after the TERM chunk.

                            This and the subsequent fields must be present
                            if termination_action is 3, and must be omitted
                            otherwise.

Delay:                   4 bytes (unsigned integer).  Delay, in ticks,
                         before repeating the sequence.

Iteration_max:           4 bytes (unsigned integer).  Maximum number of
                         times to execute the sequence.  Infinity is
                         represented by 0x7fffffff.
```

The TERM chunk, if present, must appear either immediately after the MHDR chunk or immediately prior to a SEEK chunk. Only one TERM chunk is permitted in a MNG datastream.

Simple viewers and single-frame viewers can ignore the TERM chunk. It has been made critical only so MNG editors will not inadvertently relocate it.

4.3   Critical MNG image displaying chunks

The chunk in this section provides the background against which images are displayed.

4.3.1   BACK Background

The BACK chunk suggests or mandates a background color against which transparent or less-than-full-frame images can be displayed. This information will be used whenever the application subsequently needs to insert a background layer, unless another BACK chunk provides new background information before that happens.

The BACK chunk contains 6, 7, 9, or 10 bytes. If any field is omitted, all subsequent fields must also be omitted.

```
Red_background:    2 bytes (unsigned integer).

Green_background:  2 bytes (unsigned integer).

Blue_background:   2 bytes (unsigned integer).
```

```
Mandatory_background:
                 1 byte (unsigned integer).
                    0: Background color is advisory.  Applications
                       can use it if they choose to.
                    1: Background color is mandatory.
                       Applications must use it.
                    This byte can be omitted.  If so, the
                    background color is advisory.
```

The first layer displayed by a viewer is always a background layer that fills the entire frame. The BACK chunk provides a background that the viewer can use for this purpose (or must use, if it is mandatory). If it is not "mandatory" the viewer can choose another background if it wishes. If the BACK chunk is not present, the viewer must provide its own background layer for the first frame. Each layer after the first must be composited over the layers that precede it.

The three BACK components are always written as though for an RGBA PNG with 16-bit sample depth. For example, a mid-level gray background could be specified with the RGB color samples {1.09, 1.09, 1.09}. The background color is interpreted in the current color space as defined by any top-level gAMA, cHRM, iCCP, sRGB chunks that have appeared prior to the BACK chunk in the MNG datastream. If no such chunks appear, the color space is unknown.

The data from the BACK chunk takes effect the next time the decoder needs to insert a background layer, and remains in effect until another BACK chunk appears.

For the purpose of counting layers, when the background consists of both a background color and a background image, these are considered to generate a single layer and there is no delay between displaying the background color and the background image.

Multiple instances of the BACK chunk are permitted in a MNG datastream.

The BACK chunk can be omitted. If a background is needed and the BACK chunk is omitted, then the viewer must supply its own background. For the purpose of counting layers, such a viewer-supplied background layer is counted the same as a background supplied by the BACK chunk.

In practice, most applications that use MNG as part of a larger composition should ignore the BACK data if `mandatory_background=0` and the application already has its own background definition. This will frequently be the case in World Wide Web pages, to achieve nonrectangular transparent animations displayed against the background of the page.

4.4 SAVE and SEEK chunks

Simple decoders that only read MNG datastreams sequentially can safely ignore the SAVE and SEEK chunks.

4.5 Ancillary MNG chunks

This section describes ancillary MNG chunks. MNG-compliant decoders are not required to recognize and process them.

4.5.1 eXPI Export image

The eXPI chunk takes a snapshot of an image, associates the name with that snapshot, and makes the name available to the "outside world" (like a scripting language).

The chunk contains an object identifier (snapshot id) and a name:

```
Snapshot_id:    2 bytes (unsigned integer).  Must be zero in
                    MNG-VLC datastreams.
Snapshot_name: 1-79 bytes (Latin-1 text).
```

When the snapshot_id is zero, the snapshot is the first instance of an embedded image following the eXPI chunk.

Note that the `snapshot_name` is associated with the snapshot, not with the `snapshot_id` nor its subsequent contents; changing the image identified by `snapshot_id` will not affect the snapshot. The `snapshot_name` means nothing inside the scope of the MNG-VLC specification. If two eXPI chunks use the same name, it is the outside world's problem (and the outside world's prerogative to regard it as an error). It is recommended, however, that the `snapshot_name` not be the same as that appearing in any other eXPI chunk. A decoder that knows of no "outside world" can simply ignore the eXPI chunk. This chunk could be used in MNG datastreams that define libraries of related images, rather than animations, to allow applications to extract images by their `snapshot_id`.

Names beginning with the word "thumbnail" are reserved for snapshot images that are intended to make good icons for the MNG. Thumbnail images are regular PNG images, but they would normally have smaller dimensions and fewer colors than the MNG frames.

The `snapshot_name` string must follow the format of a tEXt keyword: It must consist only of printable Latin-1 characters and must not have leading or trailing blanks, but can have single embedded blanks. There must be at least one and no more than 79 characters in the keyword. Keywords are case-sensitive. There is no null byte terminator within the `snapshot_name` string, nor is there a separate null byte terminator. Snapshot names should not begin with the case-insensitive strings "CLOCK(", "FRAME(", or "FRAMES(" which are reserved for use in URI queries and fragments (see Uniform Resource Identifier below).

Multiple instances of the eXPI chunk are permitted in a MNG datastream, and they need not have different values of `snapshot_id`.

4.5.2 pHYg Physical pixel size (global)

The MNG pHYg chunk is identical in syntax to the PNG pHYs chunk. It applies to complete MNG layers and not to the individual images within them. MNG datastreams can include both the PNG pHYs chunk

(either at the MNG top level or within the PNG and JNG datastreams) and the MNG pHYg chunk (only at the MNG top level), to ensure that the images are properly displayed either when displayed by a MNG viewer or when extracted into a series of individual PNG or JNG datastreams and then displayed by a PNG or JNG application. The pHYs and pHYg chunks would normally contain the same values, but this is not necessary.

The MNG top-level pHYg chunk can be nullified by a subsequent empty pHYg chunk appearing in the MNG top level.

4.6 Ancillary PNG chunks

The namespace for MNG chunk names is separate from that of PNG. Only those PNG chunks named in this paragraph are also defined at the MNG top level. They have exactly the same syntax and semantics as when they appear in a PNG datastream:

- iTXt, tEXt, zTXt

- tIME Same format as in PNG. Can appear at most once in the prologue segment (before the first SEEK chunk), and at most once per segment (between two consecutive SEEK chunks). In the prologue it indicates the last time any part of the MNG was modified. In a regular segment (between SEEK chunks or between the final SEEK chunk and the MEND chunk), it indicates the last time that segment was modified.

  A MNG editor that writes PNG datastreams should not include the top-level iTXt, tEXt, tIME, and zTXt chunks in the generated PNG datastreams.

- cHRM, gAMA, iCCP, sRGB, bKGD, sBIT, pHYs

  These PNG chunks are also defined at the MNG top level. They provide default values to be used in case they are not provided in subsequent PNG datastreams. Any of these chunks can be nullified by the appearance of a subsequent empty chunk with the same chunk name. Such empty chunks are not legal PNG or JNG chunks and must only appear in the MNG top level.

  In the MNG top level, all of these chunks are written as though for 16-bit RGBA PNG datastreams. Decoders are responsible for reformatting the chunk data to suit the actual bit depth and color type of the datastream that inherits them.

  A MNG editor that writes PNG or JNG datastreams is expected to include the top-level cHRM, gAMA, iCCP, and sRGB chunks in the generated PNG or JNG datastreams, if the embedded image does not contain its own chunks that define the color space. When it writes the sRGB chunk, it should write the gAMA chunk (and perhaps the cHRM chunk), in accordance with the PNG specification, even though no gAMA or cHRM chunk is present in the MNG datastream. It is also expected to write the pHYs chunk and the reformatted top-level bKGD chunk in the generated PNG or JNG datastreams, and the reformatted sBIT chunk only in generated PNG datastreams, when the datastream does not have its own bKGD, pHYs, or sBIT chunks.

The top-level sRGB chunk nullifies the preceding top-level gAMA and cHRM chunks, if any, and either the top-level gAMA or the top-level cHRM chunk nullifies the preceding top-level sRGB chunk, if any.

5 The JPEG Network Graphics (JNG) Format

JNG (JPEG Network Graphics) is the lossy sub-format for MNG objects. It is described in the full MNG specification and is also available as a separate extract from the full MNG specification. Both documents are available at the MNG home page,

```
http://www.libpng.org/pub/mng/
```

MNG-VLC applications can choose to support JNG or not. Those that do not can check bit 4 (JNG is present/absent) of the MHDR simplicity profile to decide whether they can process the datastream.

6 The Delta-PNG Format

Omitted.

7 Extension and Registration

New public chunk types, and additional options in existing public chunks, can be proposed for inclusion in this specification by contacting the PNG/MNG specification maintainers at png-info@uunet.uu.net, png-group@w3.org, or at mng-list@ccrc.wustl.edu.

New public chunks and options will be registered only if they are of use to others and do not violate the design philosophy of PNG and MNG. Chunk registration is not automatic, although it is the intent of the authors that it be straightforward when a new chunk of potentially wide application is needed. Note that the creation of new critical chunk types is discouraged unless absolutely necessary.

Applications can also use private chunk types to carry data that is not of interest to other applications.

Decoders must be prepared to encounter unrecognized public or private chunk type codes. If the unrecognized chunk is critical, then decoders should abandon the segment, and if it is ancillary they should simply ignore the chunk. Editors must handle them as described in the following section, Chunk Copying Rules.

8 Chunk Copying Rules

The chunk copying rules for MNG are similar to those in PNG. Authors of MNG editing applications should consult the full MNG specification for details.

9 Minimum Requirements for MNG-VLC-Compliant Viewers

This section specifies the minimum level of support that is expected of MNG-VLC-compliant decoders, and provides recomendations for viewers that will support slightly more than the minimum requirements. All critical chunks must be recognized, but some of them can be ignored after they have been read and recognized. Ancillary chunks can be ignored, and do not even have to be recognized.

Applications that provide less than minimal MNG support should check the MHDR "simplicity profile" for the presence of features that they are unable to support or do not wish to support. A specific subset, in which "complex MNG features", "simple MNG features" and JNG are absent, is called *"MNG-VLC"*. In MNG-VLC datastreams, bit 0 of the simplicity profile must be 1 and bits 1, 2, and 4 must be 0.

Subsets are useable when the set of MNG datastreams to be processed is known to be (or is very likely to be) limited to the feature set in MNG-VLC. Limiting the feature set in a widely-deployed WWW browser to anything less than MNG with 8-bit JNG support would be highly inappropriate.

Some subsets of MNG support are listed in the following table, more or less in increasing order of complexity.

```
MHDR Profile bits          Profile  Level of support
31-10 9 8 7 6 5 4 3 2 1 0  value

  0   0 0 0 1 0 0 0 0 0 1   65    MNG-VLC without transparency
  0   0 1 1 1 0 0 1 0 0 1   457   MNG-VLC
  0   0 1 1 1 0 1 1 0 0 1   473   MNG-VLC with JNG
      | | | | | | | | | | |
      | | | | | | | | | +- Validity
      | | | | | | | | +--- Must be zero in MNG-VLC
      | | | | | | | +----- Must be zero in MNG-VLC
      | | | | | | +------- Transparency
      | | | | | +--------- JNG
      | | | | +----------- Must be zero in MNG-VLC
      | | | +------------- Validity of bits 7, 8, and 9
      | | +--------------- Semitransparency
      | +----------------- Background transparency
      +------------------- Must be zero in MNG-VLC
```

One reasonable path for an application developer to follow might be to develop and test the application at each of the following levels of support in turn:

1. MNG-VLC,
2. MNG-LC (according to the MNG or MNG-LC specification),
3. MNG-LC with JNG,
4. MNG (according to the full MNG specification).

An equally reasonable development path might be

1. MNG-VLC with JNG,

2. MNG-LC with JNG (according to the MNG or MNG-LC specification),

3. MNG (according to the full MNG specification).

On the other hand, a developer working on an application for storing multi-page fax documents might have no need for more than "MNG-VLC without transparency".

9.1 Required MNG chunk support

MHDR

The `ticks_per_second` must be supported by animation viewers. The simplicity profile, frame count, layer count, and nominal play time can be ignored. Decoders that provide less than minimal support can use the simplicity profile to identify datastreams that they are incapable of processing.

MEND

The MEND chunk must be recognized but does not require any processing other than completing the last frame.

BACK

Must be fully supported.

DEFI, LOOP, ENDL, SAVE, SEEK, TERM

Must be recognized but can be ignored.

9.2 Required PNG chunk support

IHDR, PLTE, IDAT, IEND

All PNG critical chunks must be fully supported. All values of `color_type`, `bit_depth`, `compression_method`, `filter_method` and `interlace_method` must be supported. Interlacing, as in PNG, need not necessarily be displayed on-the-fly; the image can be displayed after it is fully decoded. The alpha-channel must be supported, at least to the degree that fully opaque pixels are opaque and fully transparent ones are transparent. It is recommended that alpha be fully supported. Alpha is not present, or can be ignored because it has no effect on the appearance of any frame, if bit 3 of the simplicity profile is 0. Bit 1 of the simplicity profile can be used to promise that only filter methods defined in the PNG specification are present.

tRNS

The PNG tRNS chunk, although it is an ancillary chunk, must be supported in MNG-compliant viewers, at least to the degree that fully opaque pixels are opaque and fully transparent ones are transparent. It is recommended that alpha data from the tRNS chunk be fully supported in the same manner as alpha data from an RGBA image or a JNG with an alpha channel contained in IDAT chunks. The tRNS chunk is not present (or can be ignored because it has no effect on the appearance of any frame) if bit 3 of the simplicity profile is 0.

Other PNG ancillary chunks

Ancillary chunks other than PNG tRNS can be ignored, and do not even have to be recognized.

Color management

It is highly recommended that decoders support at least the gAMA chunk to allow platform-independent color rendering. If they support the gAMA chunk, they must also support the sRGB chunk, at least to the extent of interpreting it as if it were a gAMA chunk with gamma value 0.45455.

9.3 Optional JNG chunk support

Bit 4 of the simplicity profile can be used to promise that JNG chunks are not present. Viewers that choose not to support JNG can check this bit before deciding to proceed. MNG-VLC decoders are not required to support JNG.

JHDR, JDAT, IDAT, JDAA, JSEP, IEND

All JNG critical chunks must be fully supported. All values of `color_type`, `bit_depth`, `compression_method`, `filter_method` and `interlace_method` must be supported. Interlacing, as in PNG, need not necessarily be displayed on-the-fly; the image can be displayed after it is fully decoded. The alpha-channel must be supported, at least to the degree that fully opaque pixels are opaque and fully transparent ones are transparent. It is recommended that alpha be fully supported.

JNG ancillary chunks

All JNG ancillary chunks can be ignored, and do not even have to be recognized.

JNG image sample depth

Only `image_sample_depth=8` must be supported. The JSEP chunk must be recognized and must be used by minimal decoders to select the eight-bit version of the image, when both eight-bit and twelve-bit versions are present, as indicated by `image_sample_depth=20` in the JHDR chunk. When `image_sample_depth=12`, minimal decoders are not obligated to display anything. Such decoders can choose to display nothing or an empty rectangle of the width and height specified in the JHDR chunk.

10 Recommendations for Encoders

The following recommendations do not form a part of the specification.

10.1 Use a common color space

It is a good idea to use a single color space for all of the layers in an animation, where speed and fluidity are more important than exact color rendition. This is best accomplished by defining a single color space at the top level of MNG, using either an sRGB chunk or the gAMA and cHRM chunks and perhaps the

iCCP chunk, and removing any color space chunks from the individual images after converting them to the common color space.

When the encoder converts all images to a single color space before putting them in the MNG datastream, decoders can improve the speed and consistency of the display.

For single-frame MNG datastreams, however, decoding speed is less important and exact color rendition might be more important. Therefore, it is best to leave the images in their original color space, as recommended in the PNG specification, retaining the individual color space chunks if the images have different color spaces. This will avoid any loss of data due to conversion.

11 Recommendations for Decoders

11.1 Using the simplicity profile

The simplicity profile in the MHDR chunk can be ignored or it can be used for

- Deciding whether to abandon a datatream immediately if it is beyond the decoder's capabilities. Decoders are of course free to plunge ahead, rendering whatever is possible and abandoning any segments that contain critical chunks that they do not recognize or cannot handle. Unmanageable features might not be present even when the simplicity profile indicates that the features "might be present". The profile never guarantees that a certain feature is present; it only guarantees that certain features are not present or have no effect on the appearance of any frame.

- Deciding whether to perform certain optimizations. For example, the transparency flags can be used to determine whether full alpha composition is going to be necessary, and to choose appropriate code paths and internal representations of abstract objects accordingly.

11.2 Decoder handling of fatal errors

When a fatal error is encountered, such as a bad CRC or an unknown critical MNG chunk, minimal viewers should simply abandon the MNG datastream.

11.3 Decoder handling of interlaced images

Decoders are required to be able to interpret datastreams that contain interlaced PNG images, but are only required to display the completed frames; they are not required to display the images as they evolve. Viewers that are decoding datastreams coming in over a slow communication link might want to do that, but MNG authors should not assume that the frames will be displayed in other than their final form.

11.4 Decoder handling of palettes

When a PLTE chunk is received, it only affects the display of the PNG datastream that includes it. Decoders must take care that it does not retroactively affect anything that has already been decoded.

If a frame contains two or more images, the PLTE chunk in one image does not affect the display of the other.

A composite frame consisting only of indexed-color images should not be assumed to contain 256 or fewer colors, since the individual palettes do not necessarily contain the same set of colors.

11.5 Behavior of single-frame viewers

Viewers that can only display a single frame must display the first frame that they encounter.

11.6 Clipping

MNG-VLC provides the following types of clipping, in addition to any clipping that might be required due to the physical limitations of the display device.

Frame width and frame height

The `frame_width` and `frame_height` are defined in the MHDR chunk and cannot be changed by any other MNG chunk.

This is the only type of clipping available in MNG-VLC datastreams.

Decoders can use these parameters to establish the size of a window in which to display the MNG frames. When the `frame_width` or `frame_height` exceeds the physical dimensions of the display hardware, the contents of the area outside those dimensions is undefined. If a viewer chooses, it can create "scroll bars" or the like, to enable persons to pan and scroll to the offscreen portion of the frame. If this is done, then the viewer is responsible for maintaining and updating the offscreen portion of the frame.

In the case of a MNG datastream that consists of a PNG or JNG datastream, with the PNG or JNG signature, the `frame_width` and `frame_height` are defined by the `width` and `height` fields of the IHDR (or JHDR) chunk.

The clipping boundaries are expressed in pixels, measured rightward and downward from the frame origin.

The left and top clipping boundaries are inclusive and the right and bottom clipping boundaries are exclusive, i.e., the pixel located at {x,y} is only displayed if the pixel falls within the physical limits of the display hardware and all of the following are true:

```
0 ≤ x < frame_width   (from the MHDR chunk)
0 ≤ y < frame_height
```

12 Recommendations for Editors

Omitted.

13 Miscellaneous Topics

13.1 File name extension

On systems where file names customarily include an extension signifying file type, the extension `.mng` is recommended for MNG (including MNG-VLC) files. Lowercase `.mng` is preferred if file names are case-sensitive. The extension `.jng` is recommended for JNG files.

14 Rationale

This (incomplete as of version 1.0) section does not form a part of the specification. It provides the rationale behind some of the design decisions in MNG.

Interframe delay

Explain why the interframe delay has to be provided *before* the subframes of layers are defined, instead of having a simpler DELA chunk that occurs in the stream where the delay is wanted.

DHDR delta types

Some delta types are not allowed when the parent object is a JNG image. Explain why types 4 and 6 (pixel replacement and color channel replacement) are not allowed under these circumstances.

Additional filter methods

Filter method 64 could have been implemented as a new critical chunk in embedded PNG datastreams.

```
FILT
   method (1 byte)
      64: intrapixel differencing
   data (variable, depends on method)
      method 64 requires no data
```

The FILT chunk would turn on this type of filtering.

The choice of using a new filter method instead of a new critical chunk was made based on simplicity of implementation and possible eventual inclusion of this method in PNG. Also, using the filter-method byte helps implementors avoid confusion about whether this is a color transform (which could affect the implementation of tRNS and other color-related chunks) or part of the filtering mechanism (which would not conceivably affect color-related chunks).

We considered using an ancillary chunk (e.g., fILt or fILT) to turn on the new filtering method. This would have the advantage that existing applications could manipulate the files, but viewers that ignore the chunk would display the image in unacceptably wrong colors, and editors could mistakenly discard the chunk.

MAGN chunk rationale

Q. Why not just use a BASI chunk to encode solid-color rectangles?

A. The MAGN chunk also allows encoding of gradient-filled rectangles.

Q. Why not just use PNG to encode gradient-filled rectangles?

A. While PNG can encode vertical and horizontal gradients fairly efficiently, it cannot do diagonal ones efficiently, and none are as efficient as a 30-byte MAGN chunk plus a 4-pixel PNG.

Q. Why not use full-scale low-quality JPEG/JNG?

A. Low-quality JPEG with reduced dimensions can be much smaller than even the lowest-quality full-sized JPEG. Such images can then be magnified to full scale with the MAGN chunk, for use as preview ("LOWSRC") images. this has been demonstrated to be about 40 to 50 times as efficient as using Adam7 interlacing of typical natural images,

It appears that in general, usable preview images of truecolor photographic images can be made at compression ratios from M*800:1 to M*2500:1, where M is the number of megapixels in the original image, by reducing the original image spatially to width and height in the range 64 to 200 pixels and then compressing the result to a medium-quality JNG.

Q. Why not use the pHYg chunk?

A. It is not mandatory for decoders to process the pHYg chunk and it does not apply to individual images; it is used to scale the entire MNG frame. The pHYs chunk cannot be used either because MNG decoders are required to ignore it.

Q. Why not 4-byte magnification factors instead of 2-byte ones?

A. Encoders can start with a larger object or, except for object 0, magnify it twice.

Q. Why not 1-byte magnification factors, then?

A. With typical screen widths currently 1280 or 1600 pixels and film and printer pages currently about 3000 pixels wide, magnifying a 1x1 image to a width of more than 255 pixels would not be uncommon.

Q. I want to magnify a "frozen" object.

A. You can make a full clone and magnify that.

Q. Why define Methods 4 and 5?

A. Method 4 is useful for magnifying an alpha-encoded image while maintaining binary transparency. Method 5 is useful for making an alpha-gradient while preserving sharp edges in the main image.

Global JPEG tables

It has been suggested that a new global MNG chunk, JTAB, be defined to hold global JPEG quantization and Huffman tables that could be inherited by JNG datastreams from which these have been omitted. This has not been tested, and we are reluctant to add new critical chunks to the MNG specification now.

15   Revision History

15.1   Version 1.0

Released 31 January 2001

- No changes.

15.2   Version 0.99

Released 10 December 2000

Revised and clarified the specification for the BACK chunk.

15.3   Version 0.98

Released 01 October 2000

- Added JPEG-encoded alpha channel in JNG and Delta-PNG datastreams, stored in a new JDAA chunk. This was approved by a formal vote.
- Separated the "transparency" profile bit into "transparency", "semitransparency", and "background transparency", and added discussion of "background transparency" to the BACK chunk specification. This proposal was approved by a formal vote.
- Added a "validity" flag to maintain backward compatibility of the simplicity profile. If it is zero, then the "background transparency", "semitransparency", and "stored object buffers" flags do *not* make any promises.

- Global sRGB nullifies global gAMA and cHRM, and *vice versa*.
- Added a note that top-level color-space chunks do not have any effect on already-decoded objects.
- Added terminology entries for "animation", "framing rate", "iteration", and "nullify".
- Various editorial changes.

15.4   Version 0.97

Released 28 February 2000.

- Minor editorial changes only.

15.5   Version 0.96

Released 18 July 1999.

- Started a Revision History section.
- Added the pHYg chunk and changed the meaning of the global pHYs chunk.
- Added the global bKGD and sBIT chunks.

15.6   Version 0.95

- Initial public release, approved by the PNG Development Group on 11 May 1999.

16   References

[LOCO]

   Weinberger, Marcelo J., Gadiel Seroussi, and Guillermo Sapiro, "The LOCO-I Lossless Image Compression Algorithm: Principles and Standardization into JPEG-LS" Hewlett Packard Report HPL-98-193R1, November 1998, revised October 1999, available at http://www.hpl.hp.com/loco/.

[PNG]

   Boutell, T., et. al., "PNG (Portable Network Graphics Format) Version 1.0", RFC 2083, ftp://ftp.isi.edu/in-notes/rfc2083.txt also available at ftp://swrinde.nde.swri.edu/pub/png/documents/. This specification has also been published as a W3C Recommendation, which is available at http://www.w3.org/TR/REC-png.html.

   See also the PNG-1.2 specification: Randers-Pehrson, G., et. al., "PNG (Portable Network Graphics Format) Version 1.2", which is available at ftp://swrinde.nde.swri.edu/pub/png/documents/.

[PNG-EXT]

   Randers-Pehrson, G., et al, "Extensions to the PNG 1.2 Specification",
   ftp://swrinde.nde.swri.edu/pub/png/documents/pngext-*.

[RFC-2119]

   Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119/BCP
   14, Harvard University, March 1997.

[RFC-2045]

   Freed, N., and N. Borenstein, "Multipurpose Internet Mail Extensions (MIME) Part One: For-
   mat of Internet Message Bodies", RFC 2045, Innosoft, First Virtual, November 1996.
   ftp://ftp.isi.edu/in-notes/rfc2045.txt

[RFC-2048]

   Freed, N., Klensin, J., and J. Postel, "Multipurpose Internet Mail Extensions (MIME) Part
   Four: Registration Procedures", RFC 2048, Innosoft, MCI, USC/Information Sciences Institute,
   November 1996.
   ftp://ftp.isi.edu/in-notes/rfc2048.txt

17   Security Considerations

Security considerations are addressed in the PNG specification.

Some people may experience epileptic seizures when they are exposed to certain kinds of flashing lights or
patterns that are common in everyday life. This can happen even if the person has never had any epileptic
seizures. All graphics software and file formats that support animation and/or color cycling make it possible
to encode effects that may induce an epileptic seizure in these individuals. It is the responsibility of authors
and software publishers to issue appropriate warnings to the public in general and to animation creators in
particular.

No known additional security concerns are raised by this format.

18   Appendix: Examples

We use the "#" character to denote commentary in these examples; such comments are not present in actual
MNG datastreams.

18.1   Example 1: A single image

The simplest MNG datastream is a single-image PNG datastream. The simplest way to create a MNG from a PNG is:

```
copy file.png file.mng
```

The resulting MNG file looks like:

```
\211 P N G \r \n ^z \n  # PNG signature.
IHDR 720 468 8 0 0 0 0  # Width and Height, etc.
sRGB 2
gAMA 45455
IDAT ...
IEND
```

If `file.png` contains an sRGB chunk and also gAMA and cHRM chunks that are recommended in the PNG specification for "fallback" purposes, you can remove those gAMA and cHRM chunks from `file.mng` because any MNG viewer that processes the gAMA chunk is also required to recognize and process the sRGB chunk, so those chunks will always be ignored. Any MNG editor that converts the MNG file back to a PNG file is supposed to insert the recommended gAMA and cHRM chunks.

18.2   Example 2: A very simple movie

This example demonstrates a very simple movie, such as might result from directly converting an animated GIF that contains a simple series of full-frame images:

```
\212 M N G \r \n ^z \n  # MNG signature.
MHDR 256 300   # Width and height.
     1         # 1 tick per second.
     5 4 4     # Layers, frames, play time
     65        # MNG-VLC simplicity
TERM 3 0 120 10   # When done, repeat animation 10 times.
IHDR ...   IDAT ...   IEND # Four PNG datastreams
IHDR ...   IDAT ...   IEND # are read and displayed.
IHDR ...   IDAT ...   IEND
IHDR ...   IDAT ...   IEND
MEND
```

18.3   Examples 3-16: Omitted from MNG-VLC.

These examples in the full MNG specification use features that are not available in MNG-VLC.

18.4   Example 17: Storing an icon library

Here is an example of storing a library of icons in a MNG-VLC datastream. All of the icons use the same "sRGBrdquo; colorspace, so a global sRGB chunk appears at the beginning. The eXPI chunks allow applications to extract the images by name.

```
MHDR 96 96 1 6 5 5 459 # Profile 457 is MNG-VLC
sRGB 2                  # Global sRGB
eXPI 0 "thumbnail"
IHDR 32 32 ... PLTE ... tRNS 0 IDAT ... IEND
eXPI 0 "left arrow"
IHDR 96 96 ... PLTE ... tRNS 0 IDAT ... IEND
eXPI 0 "right arrow"
IHDR 96 96 ... PLTE ... tRNS 0 IDAT ... IEND
eXPI 0 "up arrow"
IHDR 96 96 ... PLTE ... tRNS 0 IDAT ... IEND
eXPI 0 "down arrow"
IHDR 96 96 ... PLTE ... tRNS 0 IDAT ... IEND
MEND
```

19 Credits

Editor

- Glenn Randers-Pehrson, randeg @ alum.rpi.edu

Contributors

Contributors' names are presented in alphabetical order:

- Mark Adler, madler @ alumni.caltech.edu
- Matthias Benkmann, mbenkmann @ gmx.de
- Thomas Boutell, boutell @ boutell.com
- John Bowler, jbowler @ acm.org
- Christian Brunschen, christian @ brunschen.com
- Glen Chapman, glenc @ clark.net
- Adam M. Costello, amc @ cs.berkeley.edu
- Lee Daniel Crocker, lee @ piclab.com
- Peter da Silva, peter @ starbase.neosoft.com
- Andreas Dilger, adilger @ turbolinux.com
- Oliver Fromme, oliver @ fromme.com
- Jean-loup Gailly, jloup @ gzip.org
- Chris Herborth, chrish @ pobox.com
- Alex Jakulin, jakulin @ acm.org
- Gerard Juyn, gjuyn @ xs4all.nl
- Neal Kettler, neal @ westwood.com
- Tom Lane, tgl @ sss.pgh.pa.us
- Alexander Lehmann, lehmann @ usa.net
- Chris Lilley, chris @ w3.org
- Dave Martindale, davem @ cs.ubc.ca
- Carl Morris, msftrncs @ htcnet.com
- Owen Mortensen, ojm @ acm.org
- Josh M. Osborne, stripes @ va.pubnix.com
- Keith S. Pickens, ksp @ rice.edu
- Glenn Randers-Pehrson, randeg @ alum.rpi.edu
- Nancy M. Randers-Pehrson, randeg @ alum.rpi.edu
- Greg Roelofs, newt @ pobox.com

- Willem van Schaik, willem @ schaik.com
- Guy Schalnat, gschal @ infinet.com
- Paul Schmidt, pschmidt @ photodex.com
- Smarry Smarasderagd, smar @ reptiles.org
- Alaric B. Snell, alaric @ alaric-snell.com
- Thomas R. Tanner, ttehtann @ argonet.co.uk
- Cosmin Truta, cosmin @ cs.toronto.edu
- Guido Vollbeding, guivol @ esc.de
- Tim Wegner, twegner @ phoenix.net

Trademarks

- GIF is a service mark of CompuServe Incorporated.

Document source

This document was built from the file `mng-master-20010209` on 09 February 2001.

Copyright Notice

End of MNG-VLC Specification.