LEARNING
# png

#png

Table of Contents

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: png

It is an unofficial and free png ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official png.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with png

Remarks

Portable Network Graphics is a raster graphics file format that supports lossless data compression. PNG was created as an improved, non-patented replacement for Graphics Interchange Format (GIF) and is the most widely used lossless image compression format on the Internet.

PNG supports palette-based images (24-bit RGB or 32-bit RGBA colors), grayscale images (with or without alpha channel), and full-color, non-paletted RGB[A] images with an optional alpha channel. PNG was designed for transferring images on the Internet, not for professional-quality print graphics, and does not support non-RGB color spaces such as CMYK.

PNG files nearly always use file extension PNG or png and are assigned MIME media type image/png. PNG was approved for use by the Internet Engineering Steering Group on 14 October 1996, and was published as an ISO/IEC standard in 2004.

For image editing, either professional or otherwise, PNG provides a useful format for the storage of intermediate stages of editing. PNG compression is fully lossless and supports up to 48-bit truecolor or 16-bit grayscal. Therefore saving, restoring, and re-saving an image will not degrade its quality unlike standard JPEG (even at highest quality JPEG settings). Unlike TIFF, the PNG specification leaves no room for implementors to pick and choose what features to support; a PNG image saved in one app is readable in any other PNG-supporting application.

Note that for transmission of finished truecolor images, especially photographic ones, JPEG is almost always a better choice. JPEG's lossy compression can introduce visible artifacts, but these can be minimized and the savings in file size even at high quality levels are much better than generally possible with a lossless format like PNG. For black-and-white images, particularly of text or drawings, TIFF's Group 4 fax compression or the JBIG format are often far better than 1-bit grayscale PNG.

PNG's compression is among the best that can be had without losing image information and without paying patent fees. But not all implementations take full advantage of the available power. Even those that do can be thwarted by unwise choices on the part of the user.

Versions

png

Version	Release Date
1.0	1996-10-01

Version	Release Date
RFC-2083	1997-03-31
1.1	1999-02-15
1.2	1999-07-14
ISO/IEC 15948:2003	2003-11-10
ISO/IEC 15948:2004	2004-03-03

Examples

libpng version 1.6.21

Libpng was written as a companion to PNG specification as a way of reducing the amount of time and effort it takes to support the PNG file format in application programs.

Libpng was designed to handle multiple sessions at one time, to be easily modifiable, to be portable to the vast majority of machines (ANSI, K&R, 16-, 32-, and 64-bit) available, and to be easy to use. The ultimate goal is to promote acceptance of PNG format in whatever way possible. While there is still work to be done (see the TODO file), libpng should cover the majority of user needs.

User limits:

PNG specification allows the width and height of an image to be as large as 2^31-1 (0x7fffffff), or about 2.147 billion rows and columns. For safety libpng imposes a default limit of 1 million rows and columns. Larger images will be rejected immediately with a `png_error()` call. Libpng may reject very wide images because of potential buffer overflow conditions, but you can set your own limits with:

```
png_set_user_limits(png_ptr, width_max, height_max);
```

Put this statement after creating the PNG structure and before calling `png_read_info()`, `png_read_png()`, or `png_process_data()`.

When writing a PNG datastream, put this statement before calling `png_write_info()` or `png_write_png()`.

To retrieve the limits that are being applied, use

```
width_max = png_get_user_width_max(png_ptr);
height_max = png_get_user_height_max(png_ptr);
```

The PNG specification sets no limit on the number of ancillary chunks allowed in a PNG datastream. By default libpng imposes a limit of a total of 1000 sPLT, tEXt, iTXt, zTXt, and unknown chunks to be stored. IF both `info_ptr` and `end_info_ptr` are set, the limit applies separately to each. Change the limit on the total number of such chunks to be stored with:

```
png_set_chunk_cache_max(png_ptr, user_chunk_cache_max);
```

where `0x7fffffffL` means unlimited. You can retrieve this limit with:

```
chunk_cache_max = png_get_chunk_cache_max(png_ptr);
```

Libpng imposes a limit of 8 Megabytes (8,000,000 bytes) on the amount of memory that a compressed chunk other than IDAT can occupy, when decompressed. You can change this limit with:

```
png_set_chunk_malloc_max(png_ptr, user_chunk_malloc_max);
```

and you can retrieve the limit with:

```
chunk_malloc_max = png_get_chunk_malloc_max(png_ptr);
```

Any chunks that would cause either of these limits to be exceeded will be ignored.

Detecting libpng version:

The `png_get_io_ptr()` function has been present since libpng-0.88, has never changed, and is unaffected by conditional compilation macros. It is the best choice for use in configure scripts for detecting the presence of any libpng version since 0.88. In an autoconf "configure.in" you could use

```
AC_CHECK_LIB(png, png_get_io_ptr, ...
```

Read Getting started with png online: https://riptutorial.com/png/topic/5313/getting-started-with-png

Chapter 2: Building and installing your own libpng from source

Examples

Simple "cmake" build and install

If you want to use "cmake" (see www.cmake.org), type

```
cd [your libpng source directory]
cmake . -DCMAKE_INSTALL_PREFIX=/path
make
make install
```

where "/path" points to the installation directory where you want to put the libpng "lib", "include", and "bin" subdirectories.

Building and installing libpng with "git" and "configure"

This will download libpng from the official "git" repository and build it in your "libpng" directory.

```
git clone https://github.com/glennrp/libpng.git libpng
cd libpng
./autogen.sh
./configure [--prefix=/path]
make install
```

where "/path" points to the installation directory where you want to put the libpng "lib", "include", and "bin" subdirectories.

Read Building and installing your own libpng from source online: https://riptutorial.com/png/topic/6225/building-and-installing-your-own-libpng-from-source

Chapter 3: Downloading and installing libpng from a distro

Examples

Ubuntu, Debian, Mint (apt-based systems)

```
sudo apt-get install libpng-dev
```

on an x86_64 Ubuntu system, this installs libpng version 1.2.x in

```
libpng12.so.0 => /lib/x86_64-linux-gnu/libpng12.so.0
```

Fedora, RHEL, CentOS (yum-based systems)

```
sudo yum install libpng-devel
```

Read Downloading and installing libpng from a distro online: https://riptutorial.com/png/topic/5982/downloading-and-installing-libpng-from-a-distro

Credits

S. No	Chapters	Contributors
1	Getting started with png	4444, Community, Dakta Moriamé, Glenn Randers-Pehrson
2	Building and installing your own libpng from source	Glenn Randers-Pehrson
3	Downloading and installing libpng from a distro	Glenn Randers-Pehrson