NAME

OPENSSL_VERSION_MAJOR, OPENSSL_VERSION_MINOR, OPENSSL_VERSION_PATCH, OPENSSL_VERSION_PRE_RELEASE, OPENSSL_VERSION_BUILD_METADATA, OPENSSL_VERSION_TEXT, OPENSSL_VERSION_PREREQ, OPENSSL_version_major, OPENSSL_version_minor, OPENSSL_version_patch, OPENSSL_version_pre_release, OPENSSL_version_build_metadata, OpenSSL_version, OPENSSL_VERSION_NUMBER, OpenSSL_version_num, OPENSSL_info - get OpenSSL version number and other information

SYNOPSIS

```
#include <openssl/opensslv.h>

#define OPENSSL_VERSION_MAJOR  x
#define OPENSSL_VERSION_MINOR  y
#define OPENSSL_VERSION_PATCH  z

/* The definitions here are typical release values */
#define OPENSSL_VERSION_PRE_RELEASE ""
#define OPENSSL_VERSION_BUILD_METADATA ""

#define OPENSSL_VERSION_TEXT "OpenSSL x.y.z xx XXX xxxx"

#define OPENSSL_VERSION_PREREQ(maj,min)

#include <openssl/crypto.h>

unsigned int OPENSSL_version_major(void);
unsigned int OPENSSL_version_minor(void);
unsigned int OPENSSL_version_patch(void);
const char *OPENSSL_version_pre_release(void);
const char *OPENSSL_version_build_metadata(void);

const char *OpenSSL_version(int t);

const char *OPENSSL_info(int t);

/* from openssl/opensslv.h */
#define OPENSSL_VERSION_NUMBER 0xnnnnnnnnL

/* from openssl/crypto.h */
unsigned long OpenSSL_version_num();
```

DESCRIPTION

Macros

The three macros **OPENSSL_VERSION_MAJOR**, **OPENSSL_VERSION_MINOR** and **OPENSSL_VERSION_PATCH** represent the three parts of a version identifier, ***MAJOR.MINOR.PATCH***.

The macro **OPENSSL_VERSION_PRE_RELEASE** is an added bit of text that indicates that this is a pre-release version, such as "-dev" for an ongoing development snapshot or "-alpha3" for an alpha release. The value must be a string.

The macro **OPENSSL_VERSION_BUILD_METADATA** is extra information, reserved for other parties, such as "+fips", or "+vendor.1"). The OpenSSL project will not touch this macro (will leave it an empty string). The value must be a string.

OPENSSL_VERSION_STR is a convenience macro to get the short version identifier string, "*MAJOR.MINOR.PATCH*".

OPENSSL_FULL_VERSION_STR is a convenience macro to get the longer version identifier string, which combines **OPENSSL_VERSION_STR**, **OPENSSL_VERSION_PRE_RELEASE** and **OPENSSL_VERSION_BUILD_METADATA**.

OPENSSL_VERSION_TEXT is a convenience macro to get a full descriptive version text, which includes **OPENSSL_FULL_VERSION_STR** and the release date.

OPENSSL_VERSION_PREREQ is a useful macro for checking whether the OpenSSL version for the headers in use is at least at the given pre-requisite major (**maj**) and minor (**min**) number or not. It will evaluate to true if the header version number (**OPENSSL_VERSION_MAJOR**.**OPENSSL_VERSION_MINOR**) is greater than or equal to **maj**.**min**.

OPENSSL_VERSION_NUMBER is a combination of the major, minor and patch version into a single integer 0xMNN00PP0L, where:

M is the number from **OPENSSL_VERSION_MAJOR**, in hexadecimal notation

NN is the number from **OPENSSL_VERSION_MINOR**, in hexadecimal notation

PP is the number from **OPENSSL_VERSION_PATCH**, in hexadecimal notation

Functions

OPENSSL_version_major(), **OPENSSL_version_minor()**, **OPENSSL_version_patch()**,

OPENSSL_version_pre_release(), and **OPENSSL_version_build_metadata()** return the values of the macros above for the build of the library, respectively.

OpenSSL_version() returns different strings depending on *t*:

OPENSSL_VERSION
: The value of **OPENSSL_VERSION_TEXT**

OPENSSL_VERSION_STRING
: The value of **OPENSSL_VERSION_STR**

OPENSSL_FULL_VERSION_STRING
: The value of **OPENSSL_FULL_VERSION_STR**

OPENSSL_CFLAGS
: The compiler flags set for the compilation process in the form "compiler: ..." if available, or "compiler: information not available" otherwise.

OPENSSL_BUILT_ON
: The date of the build process in the form "built on: ..." if available or "built on: date not available" otherwise. The date would not be available in a reproducible build, for example.

OPENSSL_PLATFORM
: The "Configure" target of the library build in the form "platform: ..." if available, or "platform: information not available" otherwise.

OPENSSL_DIR
: The **OPENSSLDIR** setting of the library build in the form "OPENSSLDIR: "..."" if available, or "OPENSSLDIR: N/A" otherwise.

OPENSSL_ENGINES_DIR
: The **ENGINESDIR** setting of the library build in the form "ENGINESDIR: "..."" if available, or "ENGINESDIR: N/A" otherwise. This option is deprecated in OpenSSL 3.0.

OPENSSL_MODULES_DIR
: The **MODULESDIR** setting of the library build in the form "MODULESDIR: "..."" if available, or "MODULESDIR: N/A" otherwise.

OPENSSL_CPU_INFO
: The current OpenSSL cpu settings. This is the current setting of the cpu capability flags. It is

usually automatically configured but may be set via an environment variable. The value has the same syntax as the environment variable. For x86 the string looks like "CPUINFO: OPENSSL_ia32cap=0x123:0x456" or "CPUINFO: N/A" if not available.

For an unknown *t*, the text "not available" is returned.

OPENSSL_info() also returns different strings depending on *t*:

OPENSSL_INFO_CONFIG_DIR
: The configured "OPENSSLDIR", which is the default location for OpenSSL configuration files.

OPENSSL_INFO_ENGINES_DIR
: The configured "ENGINESDIR", which is the default location for OpenSSL engines.

OPENSSL_INFO_MODULES_DIR
: The configured "MODULESDIR", which is the default location for dynamically loadable OpenSSL modules other than engines.

OPENSSL_INFO_DSO_EXTENSION
: The configured dynamically loadable module extension.

OPENSSL_INFO_DIR_FILENAME_SEPARATOR
: The separator between a directory specification and a filename. Note that on some operating systems, this is not the same as the separator between directory elements.

OPENSSL_INFO_LIST_SEPARATOR
: The OpenSSL list separator. This is typically used in strings that are lists of items, such as the value of the environment variable $PATH on Unix (where the separator is ":") or "%PATH%" on Windows (where the separator is ";").

OPENSSL_INFO_CPU_SETTINGS
: The current OpenSSL cpu settings. This is the current setting of the cpu capability flags. It is usually automatically configured but may be set via an environment variable. The value has the same syntax as the environment variable. For x86 the string looks like "OPENSSL_ia32cap=0x123:0x456".

For an unknown *t*, NULL is returned.

OpenSSL_version_num() returns the value of **OPENSSL_VERSION_NUMBER**.

RETURN VALUES

OPENSSL_version_major(), **OPENSSL_version_minor()** and **OPENSSL_version_patch()** return the version number parts as integers.

OPENSSL_version_pre_release() and **OPENSSL_version_build_metadata()** return the values of **OPENSSL_VERSION_PRE_RELEASE** and **OPENSSL_VERSION_BUILD_METADATA** respectively as constant strings. For any of them that is undefined, the empty string is returned.

OpenSSL_version() returns constant strings.

SEE ALSO

crypto(7)

HISTORY

The macros and functions described here were added in OpenSSL 3.0, except for OPENSSL_VERSION_NUMBER and **OpenSSL_version_num()**.

COPYRIGHT