

**NAME**

curl\_version\_info - returns run-time libcurl version info

**SYNOPSIS**

```
#include <curl/curl.h>
```

```
curl_version_info_data *curl_version_info( CURLversion type);
```

**DESCRIPTION**

Returns a pointer to a filled in struct with information about various run-time features in libcurl. *type* should be set to the version of this functionality by the time you write your program. This way, libcurl will always return a proper struct that your program understands, while programs in the future might get an different struct. CURLVERSION\_NOW will be the most recent one for the library you have installed:

```
data = curl_version_info(CURLVERSION_NOW);
```

Applications should use this information to judge if things are possible to do or not, instead of using compile-time checks, as dynamic/DLL libraries can be changed independent of applications.

The curl\_version\_info\_data struct looks like this

```
typedef struct {
    CURLversion age;      /* see description below */

    /* when 'age' is 0 or higher, the members below also exist: */
    const char *version;  /* human readable string */
    unsigned int version_num; /* numeric representation */
    const char *host;     /* human readable string */
    int features;         /* bitmask, see below */
    char *ssl_version;    /* human readable string */
    long ssl_version_num; /* number */
    const char *libz_version; /* human readable string */
    const char **protocols; /* list of protocols */

    /* when 'age' is 1 or higher, the members below also exist: */
    const char *ares;     /* human readable string */
    int ares_num;         /* number */

    /* when 'age' is 2 or higher, the member below also exists: */
    const char *libidn;   /* human readable string */

} curl_version_info_data;
```

*age* describes what age of this struct this is. The number depends on how new libcurl you're using. You are however guaranteed to get a struct that you have a matching struct for in the header, as you tell libcurl your "age" with the input argument.

*version* is just an ascii string for the libcurl version.

*version\_num* is a 24 bit number created like this: <8 bits major number> | <8 bits minor number> | <8 bits patch number>. Version 7.9.8 is therefore returned as 0x070908.

*host* is an ascii string showing what host information that this libcurl was built for. As discovered by a configure script or set by the build environment.

*features* can have none, one or more bits set, and the currently defined bits are:

CURL\_VERSION\_IPV6

supports IPv6

CURL\_VERSION\_KERBEROS4

supports kerberos4 (when using FTP)

CURL\_VERSION\_SSL

supports SSL (HTTPS/FTPS)

CURL\_VERSION\_LIBZ

supports HTTP deflate using libz

CURL\_VERSION\_NTLM

supports HTTP NTLM (added in 7.10.6)

CURL\_VERSION\_GSSNEGOTIATE

supports HTTP GSS-Negotiate (added in 7.10.6)

CURL\_VERSION\_DEBUG

libcurl was built with extra debug capabilities built-in. This is mainly of interest for libcurl hackers. (added in 7.10.6)

CURL\_VERSION\_ASYNCHDNS

libcurl was built with support for asynchronous name lookups, which allows more exact timeouts (even on Windows) and less blocking when using the multi interface. (added in 7.10.7)

CURL\_VERSION\_SPNEGO

libcurl was built with support for SPNEGO authentication (Simple and Protected GSS-API Negotiation Mechanism, defined in RFC 2478.) (added in 7.10.8)

CURL\_VERSION\_LARGEFILE

libcurl was built with support for large files.

CURL\_VERSION\_IDN

libcurl was built with support for IDNA, domain names with international letters.

*ssl\_version* is an ascii string for the OpenSSL version used. If libcurl has no SSL support, this is NULL.

*ssl\_version\_num* is the numerical OpenSSL version value as defined by the OpenSSL project. If libcurl has no SSL support, this is 0.

*libz\_version* is an ascii string (there is no numerical version). If libcurl has no libz support, this is NULL.

*protocols* is a pointer to an array of char \* pointers, containing the names protocols that libcurl supports (using lowercase letters). The protocol names are the same as would be used in URLs. The array is terminated by a NULL entry.

## RETURN VALUE

A pointer to a curl\_version\_info\_data struct.

## SEE ALSO

*curl\_version(3)*