

# **Yubico YubiKey NEO Manager C Library**

---

**COLLABORATORS**

	<i>TITLE :</i> Yubico YubiKey NEO Manager C Library		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		November 12, 2013	

**REVISION HISTORY**

NUMBER	DATE	DESCRIPTION	NAME

# Contents

<b>1</b>	<b>Yubico YubiKey NEO Manager C Library</b>	<b>1</b>
1.1	ykneomgr . . . . .	1
1.2	ykneomgr-types . . . . .	6
1.3	ykneomgr-version . . . . .	7
<b>2</b>	<b>Index</b>	<b>9</b>

## Chapter 1

# Yubico YubiKey NEO Manager C Library

This is a C library to interact with the YubiKey NEO. There is a command line tool "ykneomgr" for interactive use. It supports querying the YubiKey NEO for firmware version, operation mode (OTP/CCID) and serial number. You may also mode switch the device and manage applets (list, delete and install).

For more information about Yubico and the YubiKey, see: <https://www.yubico.com/>

### 1.1 ykneomgr

ykneomgr —

#### Synopsis

ykneomgr_rc	ykneomgr_applet_delete	(ykneomgr_dev *dev, const uint8_t *aid, size_t aidlen);
ykneomgr_rc	ykneomgr_applet_install	(ykneomgr_dev *dev, const char *capfile);
ykneomgr_rc	ykneomgr_applet_list	(ykneomgr_dev *dev, char *appletstr, size_t *len);
ykneomgr_rc	ykneomgr_authenticate	(ykneomgr_dev *dev, const uint8_t *key);
ykneomgr_rc	ykneomgr_connect	(ykneomgr_dev *dev, const char *name);
ykneomgr_rc	ykneomgr_discover	(ykneomgr_dev *dev);
void	ykneomgr_done	(ykneomgr_dev *dev);
uint8_t	ykneomgr_get_mode	(ykneomgr_dev *dev);
uint32_t	ykneomgr_get_serialno	(ykneomgr_dev *dev);
uint8_t	ykneomgr_get_version_build	(ykneomgr_dev *dev);
uint8_t	ykneomgr_get_version_major	(ykneomgr_dev *dev);
uint8_t	ykneomgr_get_version_minor	(ykneomgr_dev *dev);
void	ykneomgr_global_done	(void);
ykneomgr_rc	ykneomgr_global_init	(ykneomgr_initflags flags);
ykneomgr_rc	ykneomgr_init	(ykneomgr_dev **dev);
ykneomgr_rc	ykneomgr_list_devices	(ykneomgr_dev *dev, char *devicestr, size_t *len);
ykneomgr_rc	ykneomgr_modeswitch	(ykneomgr_dev *dev,

```
uint8_t mode);
const char *      ykneomgr_strerror          (int err);
const char *      ykneomgr_strerror_name     (int err);
```

## Description

### Details

#### ykneomgr\_applet\_delete ()

ykneomgr_rc	ykneomgr_applet_delete	(ykneomgr_dev *dev, const uint8_t *aid, size_t aidlen);
-------------	------------------------	---------------------------------------------------------------

Delete specified applet.

**dev** : a **ykneomgr\_dev** device handle.

**aid** : aid to delete.

**aidlen** : length of *aid* buffer.

**Returns** : On success, **YKNEOMGR\_OK** (integer 0) is returned, or another **ykneomgr\_rc** error code.

#### ykneomgr\_applet\_install ()

ykneomgr_rc	ykneomgr_applet_install	(ykneomgr_dev *dev, const char *capfile);
-------------	-------------------------	----------------------------------------------

Install specified applet.

**dev** : a **ykneomgr\_dev** device handle.

**capfile** : string with path filename to CAP file

**Returns** : On success, **YKNEOMGR\_OK** (integer 0) is returned, or another **ykneomgr\_rc** error code.

#### ykneomgr\_applet\_list ()

ykneomgr_rc	ykneomgr_applet_list	(ykneomgr_dev *dev, char *appletstr, size_t *len);
-------------	----------------------	----------------------------------------------------------

List installed applets.

**dev** : a **ykneomgr\_dev** device handle.

**appletstr** : output buffer to hold string, or **NULL**.

**len** : on input length of *appletstr* buffer, on output holds output length

**Returns** : On success, **YKNEOMGR\_OK** (integer 0) is returned, or another **ykneomgr\_rc** error code.

---

**ykneomgr\_authenticate ()**

```
ykneomgr_rc          ykneomgr_authenticate          (ykneomgr_dev *dev,  
                                                    const uint8_t *key);
```

Authenticate to the device, to prepare for privileged function access.

**dev** : a **ykneomgr\_dev** device handle.

**key** : Double-DES key in binary, 16 bytes

**Returns** : On success, **YKNEOMGR\_OK** (integer 0) is returned, or another **ykneomgr\_rc** error code.

**ykneomgr\_connect ()**

```
ykneomgr_rc          ykneomgr_connect                (ykneomgr_dev *dev,  
                                                    const char *name);
```

Establish connection to a named PCSC device and verify that it has the YubiKey NEO applet. The *name* string should be a PCSC device name; you can use the command line tool "pcsc\_scan" to list connected devices.

**dev** : a **ykneomgr\_dev** device handle.

**name** : input string with device name to connect to.

**Returns** : On success, **YKNEOMGR\_OK** (integer 0) is returned, when no device could be found **YKNEOMGR\_NO\_DEVICE** is returned, or another **ykneomgr\_rc** error code.

**ykneomgr\_discover ()**

```
ykneomgr_rc          ykneomgr_discover                (ykneomgr_dev *dev);
```

Discover and establish connection to a YubiKey NEO. The function will return an error if more than one device is present, or if no device is present.

**dev** : a **ykneomgr\_dev** device handle.

**Returns** : On success, **YKNEOMGR\_OK** (integer 0) is returned, when no device could be found **YKNEOMGR\_NO\_DEVICE** is returned, when too many devices are present **YKNEOMGR\_TOO\_MANY\_DEVICES** is returned, or another **ykneomgr\_rc** error code.

**ykneomgr\_done ()**

```
void                  ykneomgr_done                    (ykneomgr_dev *dev);
```

Release all resources allocated to a YubiKey NEO device handle.

**dev** : device handle to deallocate, created by **ykneomgr\_init()**.

**ykneomgr\_get\_mode ()**

```
uint8_t              ykneomgr_get_mode                (ykneomgr_dev *dev);
```

Get mode of a YubiKey NEO.

**dev** : a **ykneomgr\_dev** device handle.

**Returns** : the YubiKey NEO device mode.

---

**ykneomgr\_get\_serialno ()**

```
uint32_t          ykneomgr_get_serialno          (ykneomgr_dev *dev);
```

Get serial number of a YubiKey NEO, if visible.

**dev** : a **ykneomgr\_dev** device handle.

**Returns** : the YubiKey NEO device mode, or 0 if not visible.

**ykneomgr\_get\_version\_build ()**

```
uint8_t          ykneomgr_get_version_build      (ykneomgr_dev *dev);
```

Get build version of a YubiKey NEO. Versions are in the form of BUILD.MINOR.BUILD, for example 3.0.4, in which case this function would return 4.

**dev** : a **ykneomgr\_dev** device handle.

**Returns** : the YubiKey NEO build version number.

**ykneomgr\_get\_version\_major ()**

```
uint8_t          ykneomgr_get_version_major      (ykneomgr_dev *dev);
```

Get major version of a YubiKey NEO. Versions are in the form of MAJOR.MINOR.BUILD, for example 3.0.4, in which case this function would return 3.

**dev** : a **ykneomgr\_dev** device handle.

**Returns** : the YubiKey NEO major version number.

**ykneomgr\_get\_version\_minor ()**

```
uint8_t          ykneomgr_get_version_minor      (ykneomgr_dev *dev);
```

Get minor version of a YubiKey NEO. Versions are in the form of MINOR.MINOR.BUILD, for example 3.0.4, in which case this function would return 0.

**dev** : a **ykneomgr\_dev** device handle.

**Returns** : the YubiKey NEO minor version number.

**ykneomgr\_global\_done ()**

```
void             ykneomgr_global_done            (void);
```

Release all resources from the library. Call this function when no further use of the library is needed.

**ykneomgr\_global\_init ()**

```
ykneomgr_rc      ykneomgr_global_init            (ykneomgr_initflags flags);
```

Initialize the library. This function is not guaranteed to be thread safe and must be invoked on application startup.

**flags** : initialization flags, ORed **ykneomgr\_initflags**.

**Returns** : On success **YKNEOMGR\_OK** (integer 0) is returned, and on errors an **ykneomgr\_rc** error code.

---

**ykneomgr\_init ()**

```
ykneomgr_rc          ykneomgr_init          (ykneomgr_dev **dev);
```

Create a YubiKey NEO device handle. The handle must be deallocated using **ykneomgr\_done()** when you no longer need it.

**dev** : pointer to newly allocated device handle.

**Returns** : On success, **YKNEOMGR\_OK** (integer 0) is returned, or another **ykneomgr\_rc** error code.

**ykneomgr\_list\_devices ()**

```
ykneomgr_rc          ykneomgr_list_devices  (ykneomgr_dev *dev,  
                                              char *devicestr,  
                                              size_t *len);
```

List devices.

**dev** : a **ykneomgr\_dev** device handle.

**devicestr** : output buffer to hold string, or **NULL**.

**len** : on input length of **devicestr** buffer, on output holds output length

**Returns** : On success, **YKNEOMGR\_OK** (integer 0) is returned, or another **ykneomgr\_rc** error code.

**ykneomgr\_modeswitch ()**

```
ykneomgr_rc          ykneomgr_modeswitch   (ykneomgr_dev *dev,  
                                              uint8_t mode);
```

Mode switch a YubiKey NEO.

**dev** : a **ykneomgr\_dev** device handle.

**mode** : new mode to switch the device into

**Returns** : On success, **YKNEOMGR\_OK** (integer 0) is returned, or another **ykneomgr\_rc** error code.

**ykneomgr\_strerror ()**

```
const char *          ykneomgr_strerror    (int err);
```

Convert return code to human readable string explanation of the reason for the particular error code.

This string can be used to output a diagnostic message to the user.

This function is one of few in the library that can be used without a successful call to **ykneomgr\_global\_init()**.

**err** : error code

**Returns** : Returns a pointer to a statically allocated string containing an explanation of the error code **err**.

---

## ykneomgr\_strerror\_name ()

```
const char *      ykneomgr_strerror_name      (int err);
```

Convert return code to human readable string representing the error code symbol itself. For example, `ykneomgr_strerror_name(YKNEOMGR_OK)` returns the string "YKNEOMGR\_OK".

This string can be used to output a diagnostic message to the user.

This function is one of few in the library that can be used without a successful call to `ykneomgr_global_init()`.

**err** : error code

**Returns** : Returns a pointer to a statically allocated string containing a string version of the error code *err*, or NULL if the error code is not known.

## 1.2 ykneomgr-types

ykneomgr-types —

### Synopsis

```
typedef            ykneomgr_dev;
enum               ykneomgr_initflags;
enum               ykneomgr_rc;
```

### Description

#### Details

##### ykneomgr\_dev

```
typedef struct ykneomgr_dev ykneomgr_dev;
```

##### enum ykneomgr\_initflags

```
typedef enum {
    YKNEOMGR_DEBUG = 1
} ykneomgr_initflags;
```

Flags passed to `ykneomgr_global_init()`.

**YKNEOMGR\_DEBUG** Print debug messages.

##### enum ykneomgr\_rc

```
typedef enum {
    YKNEOMGR_OK = 0,
    YKNEOMGR_MEMORY_ERROR = -1,
    YKNEOMGR_NO_DEVICE = -2,
    YKNEOMGR_TOO_MANY_DEVICES = -3,
    YKNEOMGR_BACKEND_ERROR = -4,
} ykneomgr_rc;
```

Error codes.

**YKNEOMGR\_OK** Success.

**YKNEOMGR\_MEMORY\_ERROR** Memory error.

**YKNEOMGR\_NO\_DEVICE** No device found.

**YKNEOMGR\_TOO\_MANY\_DEVICES** Too many devices found.

**YKNEOMGR\_BACKEND\_ERROR** Input/Output error.

## 1.3 ykneomgr-version

ykneomgr-version —

### Synopsis

```
#define YKNEOMGR_VERSION_MAJOR
#define YKNEOMGR_VERSION_MINOR
#define YKNEOMGR_VERSION_NUMBER
#define YKNEOMGR_VERSION_PATCH
#define YKNEOMGR_VERSION_STRING
const char * ykneomgr_check_version (const char *req_version);
```

### Description

#### Details

##### YKNEOMGR\_VERSION\_MAJOR

```
#define YKNEOMGR_VERSION_MAJOR 0
```

Pre-processor symbol with a decimal value that describe the major level of the header file version number. For example, when the header version is 1.2.3 this symbol will be 1.

##### YKNEOMGR\_VERSION\_MINOR

```
#define YKNEOMGR_VERSION_MINOR 0
```

Pre-processor symbol with a decimal value that describe the minor level of the header file version number. For example, when the header version is 1.2.3 this symbol will be 2.

##### YKNEOMGR\_VERSION\_NUMBER

```
#define YKNEOMGR_VERSION_NUMBER 0x000001
```

Pre-processor symbol with a hexadecimal value describing the header file version number. For example, when the header version is 1.2.3 this symbol will have the value 0x01020300. The last two digits are only used between public releases, and will otherwise be 00.

---

## YKNEOMGR\_VERSION\_PATCH

```
#define YKNEOMGR_VERSION_PATCH 1
```

Pre-processor symbol with a decimal value that describe the patch level of the header file version number. For example, when the header version is 1.2.3 this symbol will be 3.

## YKNEOMGR\_VERSION\_STRING

```
#define YKNEOMGR_VERSION_STRING "0.0.1"
```

Pre-processor symbol with a string that describe the header file version number. Used together with `ykneomgr_check_version()` to verify header file and run-time library consistency.

## ykneomgr\_check\_version ()

```
const char *      ykneomgr_check_version      (const char *req_version);
```

Check that the version of the library is at minimum the requested one and return the version string; return NULL if the condition is not satisfied. If a NULL is passed to this function, no check is done, but the version string is simply returned.

See **YKNEOMGR\_VERSION\_STRING** for a suitable *req\_version* string.

**req\_version** : Required version number, or NULL.

**Returns** : Version string of run-time library, or NULL if the run-time library does not meet the required version number.

---

## Chapter 2

# Index

### Y

ykneomgr\_applet\_delete, 2  
ykneomgr\_applet\_install, 2  
ykneomgr\_applet\_list, 2  
ykneomgr\_authenticate, 3  
ykneomgr\_check\_version, 8  
ykneomgr\_connect, 3  
ykneomgr\_dev, 6  
ykneomgr\_discover, 3  
ykneomgr\_done, 3  
ykneomgr\_get\_mode, 3  
ykneomgr\_get\_serialno, 4  
ykneomgr\_get\_version\_build, 4  
ykneomgr\_get\_version\_major, 4  
ykneomgr\_get\_version\_minor, 4  
ykneomgr\_global\_done, 4  
ykneomgr\_global\_init, 4  
ykneomgr\_init, 5  
ykneomgr\_initflags, 6  
ykneomgr\_list\_devices, 5  
ykneomgr\_modeswitch, 5  
ykneomgr\_rc, 6  
ykneomgr\_strerror, 5  
ykneomgr\_strerror\_name, 6  
YKNEOMGR\_VERSION\_MAJOR, 7  
YKNEOMGR\_VERSION\_MINOR, 7  
YKNEOMGR\_VERSION\_NUMBER, 7  
YKNEOMGR\_VERSION\_PATCH, 8  
YKNEOMGR\_VERSION\_STRING, 8

---