SDK API Manual

Version 1.3

NetModule AG, Switzerland

October 24, 2017

Contents

1 Introduction

This manual describes the SDK API functions which provide a range of general-purpose extensions for the Arena scripting language.

Version 1.2 ships with the following features:

```
- Send and retrieve SMS
- Send E-mail
- Read or write from or to a serial device
- Control digital input/output ports
- Run TCP/UDP servers
- Run IP/TCP/UDP clients
- Access files of mounted media (e.g. an USB stick)
- Retrieve status information from the system
- Get or set configuration values
- Write to syslog
- Transfer files over HTTP/FTP
- Perform config/software updates
- Control LEDs
- Get system events or reboot system
- Scan available networks
- Web Pages
- SNMP functions
- Various network-related functions
- Other system-related functions
```

2 API functions

2.1 Serial

2.1.1 nb_serial_getattr

```
struct nb_serial_getattr (string dev)
```

The nb_serial_getattr function retrieves the current attributes associated to a serial device.

```
dev         serial device (e.g. serial0 for first device)
```

Returns a struct containing values for baudrate, databit, stopbit, parity, flowctl or void on error.

2.1.2 nb_serial_setattr

```
int nb_serial_setattr (string dev, int b, int d, int s, int p, int
    f)
```

The nb_serial_setattr function can be used to set the attributes of a serial device.

```
dev         serial device (e.g. serial0 for first device)
b           baudrate e.g. 9600, 19200, 38400, 57600, 115200
d           number of data bits (5, 6, 7, 8)
s           number of stop bits (1, 2)
p           parity (0=no parity, 1=odd parity, 2=even parity)
f           flow control (0=none, 1=xon/xoff, 2=hardware)
```

Returns -1 on error otherwise zero.

2.1.3 nb_serial_write

```
int nb_serial_write (string dev, string msg)
```

The nb_serial_write function can be used to directly write a message to a serial device.

```
dev        serial device (e.g. serial0 for first device)
msg        message to be written
```

Returns number of bytes write or -1 on error.

2.1.4 nb_serial_read

```
string nb_serial_read (string dev)
```

The nb_serial_read function can be used to read a message from a serial device.

```
dev        serial device (e.g. serial0 for first device)
```

Returns the string received from the serial port or an empty string in case of errors.

2.2 Media

2.2.1 nb_media_mount

```
int nb_media_mount (string dev)
```

The nb_media_mount function mounts the specified media device.

```
dev        device name (eg. usb0)
```

Returns 0 on success and -1 on error. The media will be mounted in e.g. /mnt/media/usb0, you can use any IO functions to operate on it.

2.2.2 nb_media_umount

```
int nb_media_umount (string dev)
```

The nb_media_umount function umounts the specified media device.

```
dev        device name (e.g usb0)
```

Returns -1 on error.

2.2.3 nb_media_getmount

```
string nb_media_getmount (void)
```

The nb_media_getmount function returns all media currently mounted including the corresponding mountpoint. Returns list of mounted media and its mountpoints line-by-line (such as "xx on xx/xx"). If nothing is mounted or in case of an error an empty string will be returned.

2.3 SMS

Please note that the SMS daemon must be properly configured prior to using the functions.

2.3.1 nb_sms_send

```
string nb_sms_send (string phonenum, string msg)
```

The nb_sms_send function can be used to send a SMS to the specified telephone number.

```
phonenum    recipient's phone number (international format)
msg         message to be sent
```

Returns message id on success or NULL on error.

2.3.2 nb_sms_list

```
array nb_sms_list (void)
```

The nb_sms_list function can be used to retrieve the messages in the inbox. Returns an array of message files.

2.3.3 nb_sms_retrieve

```
string nb_sms_retrieve (string file)
```

The nb_sms_retrieve function returns the whole message specified by file.

2.3.4 nb_sms_header

```
string nb_sms_header (string file, string tag)
```

The nb_sms_header function returns the headers of a given message.

```
file    the message file
tag     a specific header tag (such as "From")
```

Returns header value of specified tag, all headers if omitted or NULL on error.

2.3.5 nb_sms_body

```
string nb_sms_body (string file)
```

The nb_sms_body function returns the body of a given message.

```
file    the message file
```

Returns the message's body text or NULL on error.

2.3.6 nb_sms_delete

```
int nb_sms_delete (string file)
```

The nb_sms_delete function can be used to delete a message from the inbox.

```
file    the message file
```

Returns 0 on success or -1 on error.

2.4 E-mail

Please note that the E-Mail client must be properly configured prior to using the functions.

2.4.1 nb_email_send

```
int nb_email_send (string rcpt, string subj, string msg)
```

The nb_email_send function can be used to send an E-Mail to a particular address. Please note that the E-Mail client must have been set up correctly prior to using this function.

```
rcpt    recipient's email address (e.g. abc@abc.com)
subj    email subject
msg     email content
```

Returns 0 on success or any error code.

2.5 Digital I/O

2.5.1 nb_dio_get

```
int nb_dio_get (string port)
```

The nb_dio_get function retrieves the status of a digital I/O port.

```
port    DIO port to be queried (in1, in2, out1, out2)
```

Returns the DIO status (0 = off, 1 = on) or -1 on error.

2.5.2 nb_dio_set

```
int nb_dio_set (string port, int state)
```

The nb_dio_set function can be used to turn on/off the status of a digital output port.

```
port    digital output port to be configured (out1, out2)
state   new output status (0 = off, 1 = on)
```

Returns -1 on error.

2.5.3 nb_dio_summary

```
string nb_dio_summary (void)
```

The nb_dio_summary function retrieves the status of all digital I/O ports.
Returns a string holding the status of all ports or an empty string on error.

2.6 Configuration

2.6.1 nb_config_get

```
string nb_config_get (string key)
```

The nb_config_get function returns the currently configured value of a particular config parameter.

```
    key         config key (such as sdk.status)
```

Returns the config value or NULL on error.

2.6.2 nb_config_set

```
    int nb_config_set (string config)
```

The nb_config_set function can be used to set system configuration parameters. Note, that this function will immediately apply the values to the system.

```
    config      config to be set in the form key=value (e.g.
       sdk.status=0)
```

Returns -1 on error.

2.6.3 nb_config_summary

```
    string nb_config_summary (void)
```

The nb_config_summary function will return the current system configuration, that is, the difference between the factory and the running configuration.

2.7 Status Information

2.7.1 nb_status

```
    struct nb_status (string section)
```

The nb_status function will return various status values, the following sections can be specified:

```
    section     the status section which shall be queried which can be:
                system          System information
                license         License information
                wwan            WWAN module status
                wlan            WLAN module status
                gnss            GNSS (GPS) module status
```

```
                lan             LAN interface status
                wan             WAN interface status
                openvpn         OpenVPN connection status
                ipsec           IPsec connection status
                dio             Digital IO status
```

Returns a struct holding the relevant status values (see 'status.are' example).

2.7.2 nb_status_summary

```
    string nb_status_summary (void)
```

The nb_status_summary function will return a short summary about the current system status or NULL on error.

2.8 Network Scanning

2.8.1 nb_scan_networks

```
    struct nb_scan_networks (string ifc)
```

The nb_scan_networks function can be used to scan for available networks.

```
    ifc      the interface to scan (e.g. WLAN1 or Mobile1)
```

Returns a struct holding the relevant networks (see examples).

Please note, that scanning a mobile interface will tear down any running WWAN connections. Same applies to WLAN interfaces operating in access-point mode. Therefore the scan interval is limited to 30 seconds.

2.9 File Transfers

The file transfer functions can be used to transfer files from or to a remote server denoted by an FTP or HTTP/HTTPS URL.

2.9.1 nb_transfer_get

```
    int nb_transfer_get (string usr, string pwd, string url, string
       path)
```

The nb_transfer_get function can be used to get a file from a remote server. The usr/pwd arguments can be applied in order to perform authentication.

```
usr the username used for authentication (can be empty)
pwd the password used for authentication (can be empty)
url the URL where to get the file from
path  the local path where the file should be stored
```

Returns -1 on error.

2.9.2 nb_transfer_put

```
int nb_transfer_put (string usr, string pwd, string url, string
   path)
```

The nb_transfer_put function can be used to transfer a file to a remote server. The usr/pwd arguments can be applied in order to perform authentication.

```
usr the username used for authentication (can be empty)
pwd the password used for authentication (can be empty)
url the URL where to put the file
path  the path to the local file which sould be sent
```

Returns -1 on error.

2.9.3 nb_transfer_post

```
int nb_transfer_post (string usr, string pwd, string url, string
   path, string postfields)
```

The nb_transfer_post function can be used to transfer a file to a remote HTTP server. By using the POST method, additional parameters may be passed. The usr/pwd arguments can be applied in order to perform authentication.

```
usr         the username used for authentication (can be empty)
pwd         the password used for authentication (can be empty)
url         the URL where to put the file
path          the path to the local file which sould be sent
postfields      addition POST parameters in the form:
   <key1>=<val1>&<key2>=<val2>&<keyN>=<valN>
```

Returns -1 on error.

2.9.4 nb_transfer_list

```
array nb_transfer_list (string usr, string pwd, string url)
```

The nb_transfer_list function can be used to retrieve the list of files from a remote server. The usr/pwd arguments can be applied in order to perform authentication.

```
usr          the username used for authentication (can be empty)
pwd          the password used for authentication (can be empty)
url          the URL, specifying the directory to be listed
```

Returns an array of struct describing the directory files. They are made up of:
name name of the file size file size in bytes mode file mode and permission (see chmod) user owner username group owner groupname time modification time tm time struct of modification time

2.9.5 nb_transfer_delete

```
int nb_transfer_delete (string usr, string pwd, string url)
```

The nb_transfer_delete function can be used to delete a file from a remote FTP server. The usr/pwd arguments can be applied in order to perform authentication.

```
usr the username used for authentication (can be empty)
pwd the password used for authentication (can be empty)
url the URL containing the path to the to be deleted file
```

Returns -1 on error.

2.10 LED

2.10.1 nb_led_acquire

```
int nb_led_acquire (int led)
```

The nb_led_acquire function will acquire the specified indication LED to be used for a script. Any associated system indication will stop being signalled until the LED is released again. The status LED cannot be acquired/set.

```
led the number of the LED to be acquired
  (starting from left/top) or LED_ALL for all LEDs
```

Returns -1 on error otherwise zero.

2.10.2 nb_led_release

```
int nb_led_release (int led)
```

The nb_led_release function will release an acquired LED again.

```
led the number of the LED to be released or LED_ALL for all LEDs
```

Returns -1 on error otherwise zero.

2.10.3 nb_led_set

```
int nb_led_set (int led, int mode)
```

The nb_led_set function will set the specified LED to a specific mode.

```
led the number of the LED to be released or LED_ALL for all LEDs
mode    the LED mode to be applied which can be specified by
   OR'ing the following colors and types:
  LED_OFF           turn off LED
  LED_COLOR_GREEN   color is green
  LED_COLOR_RED     color is red
  LED_COLOR_YELLOW  color is yellow
  LED_SOLID         type is solid
  LED_BLINK_FAST    type is fast blinking
  LED_BLINK_SLOW    type is slow blinking
```

Returns -1 on error otherwise zero.

2.11 Config/Software Update

The following functions can be used to trigger a configuration or software update of the system. The URL can be specified as follows:

```
http://<server>/<path>      (retrieve file via HTTP)
https://<server>/<path>     (retrieve file via HTTPS)
ftp://<server>/<path>       (retrieve file via FTP)
tftp://<server>/<path>      (retrieve file via TFTP)
file:///<path>              (use local file)
```

Please bear in mind that calling nb_update_software() will result in a system reboot. The nb_update_config() call will restart the SDK which will terminate your scripts. Thus, it is recommended to exit the script after calling this function and check the result later on via nb_update_status().

2.11.1 nb_update_status

```
string nb_update_status (void)
```

The nb_update_status function returns the status of the last or currently running update operation

2.11.2 nb_update_config

```
int nb_update_config (string url)
```

The nb_update_config function will perform a configuration update from the specified URL.

```
url     the URL of the config
```

Returns zero on success.

2.11.3 nb_update_software

```
int nb_update_software (string url)
```

The nb_update_software function will perform a software update from the specified URL.

```
url     the URL of the software
```

Returns zero on success.

2.11.4 nb_update_sshkeys

```
int nb_update_sshkeys (string url)
```

The nb_update_sshkeys function will perform an update of the SSH authorized keys.

```
url     the URL of the keys
```

Returns zero on success.

2.12 Web Pages

The following functions can be used to implement your own pages within the Web Manager. Such a page will appear under the SDK menu as soon as it has been registered.

2.12.1 nb_page_register

```
int nb_page_register (int id, string title)
int nb_page_register (int id, string title, string submenu)
```

The nb_page_register function registeres a new page with the specified identifier and title (as well as submenu).

```
id              identifier
title           page title
submenu         submenu for page
```

Returns -1 on error and otherwise a page structure which can be used for further page functions.

2.12.2 nb_page_unregister

```
int nb_page_unregister (struct page)
```

The nb_page_unregister function can be used to unregister a page again.

```
page    page structure
```

Returns -1 on error, otherwise zero.

2.12.3 nb_page_request

```
struct nb_page_request (struct page)
```

The nb_page_request function listens for incoming requests.

```
page    page structure
```

Returns void on error, otherwise a request structure which holds possible GET and POST parameters.

2.12.4 nb_page_respond

```
int nb_page_respond (struct page, string fmt, ...)
```

The nb_page_responds function can be used to echo back a string to the request and can be called multiple times until page is to be finished. It supports a fmt string and additional arguments that are formatted according to the format string. Please refer to the printf function for more information about format options.

```
page    page structure
```

fmt format string

Returns -1 on error and zero on success.

2.12.5 nb_page_finish

```
int nb_page_finish (struct page)
```

The nb_page_finish function will be used to signal that responding to a request has been finished and data shall be passed to the client.

```
page    page structure
```

Returns -1 on error and zero on success.

2.13 SNMP functions

2.13.1 nb_snmp_traphost

```
int nb_snmp_traphost (string host, int port, int version, string
   community)
int nb_snmp_traphost (string host, int port, int version, string
   user, string password, string auth, string priv)
```

The nb_send_traphost function will set the host for sending SNMP traps.

For an SNMPv1/v2 host the parameters are:

```
host    hostname or address
port    trap port
version   SNMP version (1 or 2)
community community string
```

For an SNMPv3 host the parameters are:

```
host  hostname or address
port  trap port
version SNMP version (3)
user  username
pass  password
auth  authentication protocol (MD5 or SHA)
priv  privacy protocol (DES or AES)
```

Returns -1 on error.

2.13.2 nb_snmp_trap

```
string nb_snmp_trap (string oid, string type, string value)
```

The nb_send_trap function will send an SNMP trap with the specified OID to a remote traphost.

```
oid   SNMP object identifier of the trap
type  type of value to be sent ('e' for empty, 'i' for integer and
   's' for octet string)
value   value to be sent
```

Please note that a traphost has to be set prior to using this function.
Returns -1 on error.

2.14 Various network-related functions

2.14.1 nb_gethostbyname

```
array nb_gethostbyname (string host)
```

The nb_gethostbyname function performs a DNS lookup for the given hostname and returns an array of resolved IP addresses. Please note that a valid DNS is available prior to using this function.

```
host    the to be resolved host
```

Returns an empty array if host could not be resolved.

2.14.2 nb_ifc_address

```
string nb_ifc_address (string interface)
```

The nb_ifc_address function can be used to obtain the first address of an interface.

```
interface       internal interface name (e.g. lan0)
```

Returns the interface address or NULL on error.

2.14.3 nb_ping

```
int nb_ping (string host)
int nb_ping (string host, int timeout)
```

The nb_ping function will send ICMP ping packets to the specified host and returns whether the host correctly responded or not.

```
host            the host to ping
timeout         timeout waiting for a reply (in milliseconds)
```

Returns 1 in case the host is alive, 0 if down and -1 on error.

2.14.4 nb_arp_ping

```
int nb_arp_ping (string host)
```

The nb_arp_ping function will send an ARP request for the specified host and returns whether the host address has been successfully resolved.

```
host            the host address to ping
```

Returns 1 in case the specified host could be resolved, 0 if not and -1 on error.

2.14.5 nb_arp_gratuitous

```
int nb_arp_gratuitous (string interface)
int nb_arp_gratuitous (string interface, string host)
```

The nb_arp_gratuitous function will send an gratuitous ARP advert for the address of the specified interface (or the host address provided). This can be used to update the ARP tables of your neighbors.

```
interface       the interface on which the packet should be send
host            the host address to advert
```

Returns 1 in case the packet could be sent or -1 on error.

2.14.6 nb_etherwake

```
int nb_etherwake (string hwaddr, string interface)
```

The nb_etherwake function will send a WakeOnLan magic packet to wake up sleeping hosts.

```
hwaddr          the Ethernet MAC address of the host
interface       the interface on which the packet is sent
```

Returns 0 in case the packet has been successfully sent or -1 on error.

2.15 Other system functions

2.15.1 nb_syslog

```
int nb_syslog (string fmt, ...)
```

The nb_syslog function creates a message in the system log. Please refer to sprintf for more information about the format string fmt and additional arguments.

```
msg       message to be written to syslog
```

Returns -1 on error.

2.15.2 nb_event_get

```
string nb_event_get (int timeout)
```

The nb_event_get function will poll for system events.

```
timeout         max. number of seconds to wait for an event
```

Returns received event as string or NULL when the specified timeout has been reached.

2.15.3 nb_reboot

```
void nb_reboot (int delay)
```

The nb_reboot function will trigger a system reboot.