Kvaser REST API Specification

Printed Thursday 21st May, 2015

(This page is intentionally left blank.)

Contents

1 About this Manual

This document specifies the JSON REST API that is available in selected Kvaser CAN interfaces. The reader is assumed to be familiar with Kvaser CANlib.

THIS SPECIFICATION IS SUBJECT TO CHANGE!

2 Introduction

This document specifies the JSON REST API which is available in selected Kvaser CAN interfaces. The API is based upon the Kvaser CANlib, so most function and parameter names are the same. Assuming that the device is connected, has IP address 192.168.1.10, and is listening on port 8080, you can access the API in the form:

`http://192.168.1.10:8080/deviceStatus`

The following rules applies to this API:

- All constants must be specified with its numerical value, e.g. `canBITRATE_1M` should be given as `-1`
- All numbers are decimal
- All calls can take an optional integer parameter, `ident=` which is included in the response

The status constants that are currently used by the JSON REST API are listed in Table 1.

Constant	Value
canOK	0
canERR_PARAM	-1
canERR_NOMSG	-2
canERR_NOCHANNELS	-5
canERR_TIMEOUT	-7
canERR_INVHANDLE	-10
canERR_NOT_IMPLEMENTED	-32
canERR_INVALID_PASSWORD[1]	-128
canERR_NO_SUCH_FUNCTION[1]	-129
canERR_NOT_AUTHORIZED[1]	-130
canERR_INVALID_SESSION[1]	-131

Table 1: Status constants used by the JSON REST API. Note that the last constants are extensions to current Kvaser CANlib.

[1] Extensions to current Kvaser CANlib.

3 Connection Functions

The following functions are used to query and connect to a device. This is done differently than in Kvaser CANlib, e.g. the session concept is new.

3.1 deviceStatus

You may at any time ask a device for its status with the function `deviceStatus`. The device can then respond whether it is free or already connected to some host.

- `uri:/deviceStatus`
- parameters:

  [mode=jsonp] If set, the response will be coded in JSONP, i.e. wrapped with the fixed string 'canlib_callback(...)'.

- returns:

```
"usage" : %u,           # Flags: 0=Free, 1=In use via service,
                        # 2=In use via JSON API,
                        # 4=In use via JSONP API

# The following response codes are only present if the device
# is in use:
"timeout" : %u,         # time left before current session
                        # will end (seconds)
"ip" : "%u.%u.%u.%u"    # IP address of the host the device
                        # is currently connected to.
```

- example:

```
http://192.168.1.10:8080/deviceStatus?mode=jsonp

canlib_callback({"usage":1})

http://192.168.1.10:8080/deviceStatus?mode=jsonp&ident=0001

canlib_callback({"usage":0, "ident":1})

http://192.168.1.10:8080/deviceStatus

{"usage":1}

http://192.168.1.10:8080/deviceStatus

{"usage":3, "timeout":1066, "ip":"192.168.1.12"}
```

3.2 canInitializeLibrary

The function `canInitializeLibrary` sets up a connection and creates a session. Therefore, this routine must be called before any other function that needs a session. When a session is active, the device will deny any further calls to this function by returning status code -131 (Invalid session).

The returned session must be used when calling other functions, denoted with <session> below. A session is terminated either by an explicit call to `canUnloadLibrary` or after an inactivity of more than 'timeout' seconds.

- `uri:/canInitializeLibrary`
- parameters:

  [password=%s] Access password, the string can be URI encoded if needed.

  [mode=jsonp] When set, the response will be coded as JSONP.

  [timeout=%u] The session timeout in seconds.

  [dummy_session=%u] If set to '1', the session returned will be '00000000000000000000000000000000'.

- returns:

```
{"stat":<canOK | canERR_xxx>, "session":"%32x"}
```

- example:

```
http://192.168.1.10:8080/canInitializeLibrary?timeout=1200

{"stat":0, "session":"1b6ed79f755f0ab94ff9ad62470ad0a0"}

http://192.168.1.10:8080/canInitializeLibrary?timeout=1200

{"stat":-131}
```

3.3 canUnloadLibrary

The function `canUnloadLibrary` terminates an active session, making the device free.

- `uri:/<session>/canUnloadLibrary`
- parameters: None.
- returns:

  ```
  {"stat":<canOK | canERR_xxx>}
  ```

- example:

  ```
  http://192.168.1.10:8080/1b6ed79f755f0ab94ff9ad62470ad0a0/
      canUnloadLibrary

  {"stat":0}
  ```

4 CANlib Equivalent Functions

The following functions mirror functions in Kvaser CANlib. Please refer to the CANlib documentation for further information about these functions and their parameters.

Note that all constants must be specified with their decimal numerical values, e.g. `canBITRATE_1M` should be given as `-1`. All return values will also be returned as decimal integers, e.g. `canOK` will be returned as `0`.

4.1 canOpenChannel

The function `canOpenChannel` returns a handle to the opened channel. This handle should be passed in other functions as the `hnd` parameter as needed.

- `uri:<session>/canOpenChannel`
- parameters:

  channel=%u Channel number on the device.

  flags=%u Flags according to canOPEN_xxx.
- returns:

```
{"stat":<canOK | canERR_xxx>, "hnd":%d}
# hnd is only valid if stat returns canOK
```

- example:

```
http://192.168.1.10:8080/1b6ed79f755f0ab94ff9ad62470ad0a0/
    canOpenChannel?channel=0&flags=8

{"stat":0, "hnd":0}
```

4.2 canClose

- `uri:<session>/canClose`
- parameters:

  hnd=%u
- returns:

```
{"stat":<canOK | canERR_xxx>}
```

- example:

```
http://192.168.1.10:8080/1b6ed79f755f0ab94ff9ad62470ad0a0/
    canClose?hnd=0

{"stat":0}
```

4.3 canSetBusParams

- `uri:<session>/canSetBusParams`
- parameters:

  hnd=%u

  freq=%d Bit rate, or one of canBITRATE_xxx. If freq is not any of canBITRATE_xxx, the following parameters must also be set:

  tseg1=%u

  tseg2=%u

  sjw=%u

  noSamp=%u

- returns:

```
{"stat":<canOK | canERR_xxx>}
```

- example:

```
http://192.168.1.10:8080/1b6ed79f755f0ab94ff9ad62470ad0a0/
    canSetBusParams?hnd=0&freq=-1&ident=1234

{"stat":0, "ident":1234}
```

4.4 canBusOn

- `uri:<session>/canBusOn`
- parameters:

  hnd=%u

- returns:

```
{"stat":<canOK | canERR_xxx>}
```

- example:

```
http://192.168.1.10:8080/1b6ed79f755f0ab94ff9ad62470ad0a0/
    canBusOn?hnd=0

{"stat":0}
```

4.5 canBusOff

- `uri:<session>/canBusOff`
- parameters:

  hnd=%u
- returns:

  `{"stat":<canOK | canERR_xxx>}`

- example:

  `http://192.168.1.10:8080/1b6ed79f755f0ab94ff9ad62470ad0a0/canBusOff?hnd=0`

  `{"stat":0}`

4.6 canSetBusOutputControl

- `uri:<session>/canSetBusOutputControl`
- parameters:

  hnd=%u

  drivertype=%u Driver type according to canDRIVER_xxx.
- returns:

  `{"stat":<canOK | canERR_xxx>}`

- example:

  `http://192.168.1.10:8080/1b6ed79f755f0ab94ff9ad62470ad0a0/canSetBusOutputControl?hnd=0&drivertype=4`

  `{"stat":0}`

4.7 canRead

- `uri:<session>/canRead`
- parameters:

  hnd=%u

  max=%u Max number of messages to be received in the answer, default is 1.

- returns:

```
{"stat":<canOK | canERR_xxx>,
 "msgs:" : [
   {
     "id" : %u,
     "dlc"  : %u,
     "time" : %u,
     "flag" : %u,
     "msg"  : [%u, ...]
   },
   ...
 ]
}
```

- example:

```
192.168.1.10:8080/1b6ed79f755f0ab94ff9ad62470ad0a0/canRead?
   hnd=0&max=5

{"stat":0, "msgs":[
  {"id":10, "dlc":4, "msg":[67,12,8,0], "time":3520517614,"
   flag":2},
  {"id":12, "dlc":3, "msg":[32,12,16], "time":3520517724,"
   flag":2},
  {"id":14, "dlc":4, "msg":[0,0,24,0], "time":3520517835,"
   flag":2}]}
```

4.8 canWrite

- `uri:<session>/canWrite`
- parameters:

  hnd=%u

  id=%u

  flag=%u

  dlc=%u

  msg=%u[,%u[...]]

- returns:

```
{"stat":<canOK | canERR_xxx>}
```

- example:

```
http://192.168.1.10:8080/1b6ed79f755f0ab94ff9ad62470ad0a0/
    canWrite?hnd=0&id=55&flag=0&msg
    =99,100,101,102,103,104,105&dlc=7

{"stat":0}
```

4.9 canIoCtl

- `uri:<session>/canIoCtl`
- parameters:

  hnd=%u

  func=%u Function according to canIOCTL_xxx.

  buf=%s Parameter depending on the actual function used.

- returns:

```
{"stat":<canOK | canERR_xxx>}
```

- example:

```
http://192.168.1.10:8080/1b6ed79f755f0ab94ff9ad62470ad0a0/
    canIoCtl?hnd=0&func=10

{"stat":0}
```

5 Connection flow examples

This capter contains a number of flow examples.

- Successful log in, Figure 1.
- Unsuccessful log in, Figure 2.
- Log in, send a CAN message and clean up, Figure 3 on Page 15.
- Invalid channel, Figure 4 on Page 16.
- Read CAN messages (messages available), Figure 5 on Page 16.
- Read CAN messages (no messages available), Figure 6 on Page 17.
- Error examples, Figure 7 on Page 17.

Figure 1: Successful log in

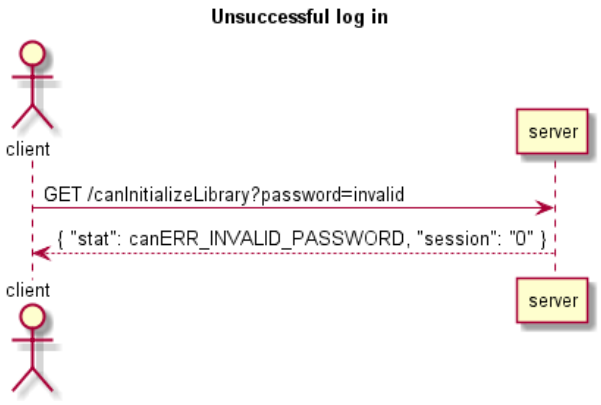

Figure 2: Unsuccessful log in

Figure 3: Log in, send a CAN message and clean up

Figure 4: Invalid channel

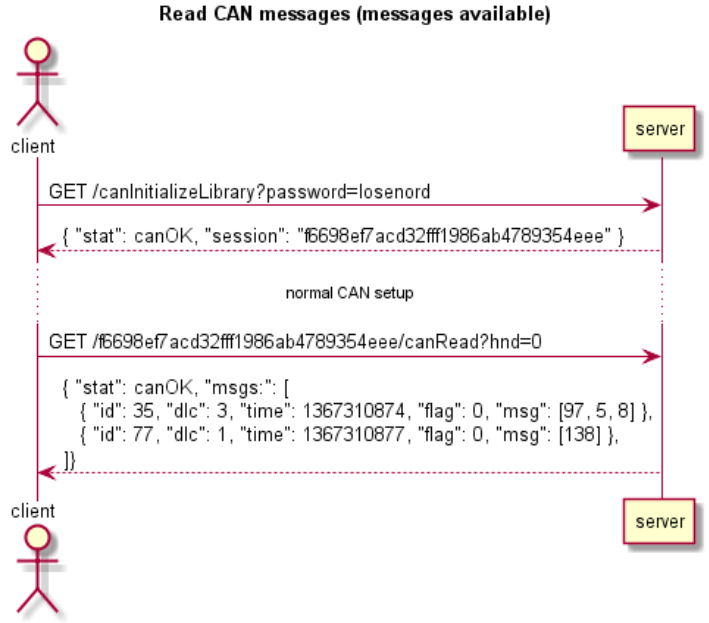

Figure 5: Read CAN messages (messages are available)

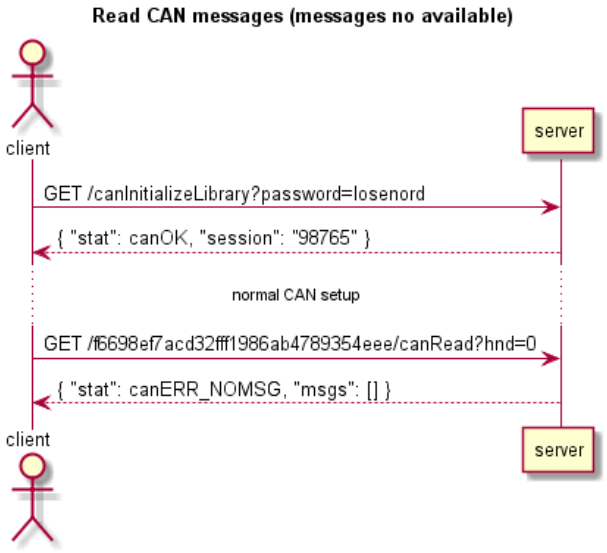

Figure 6: Read CAN messages (no messages are available)

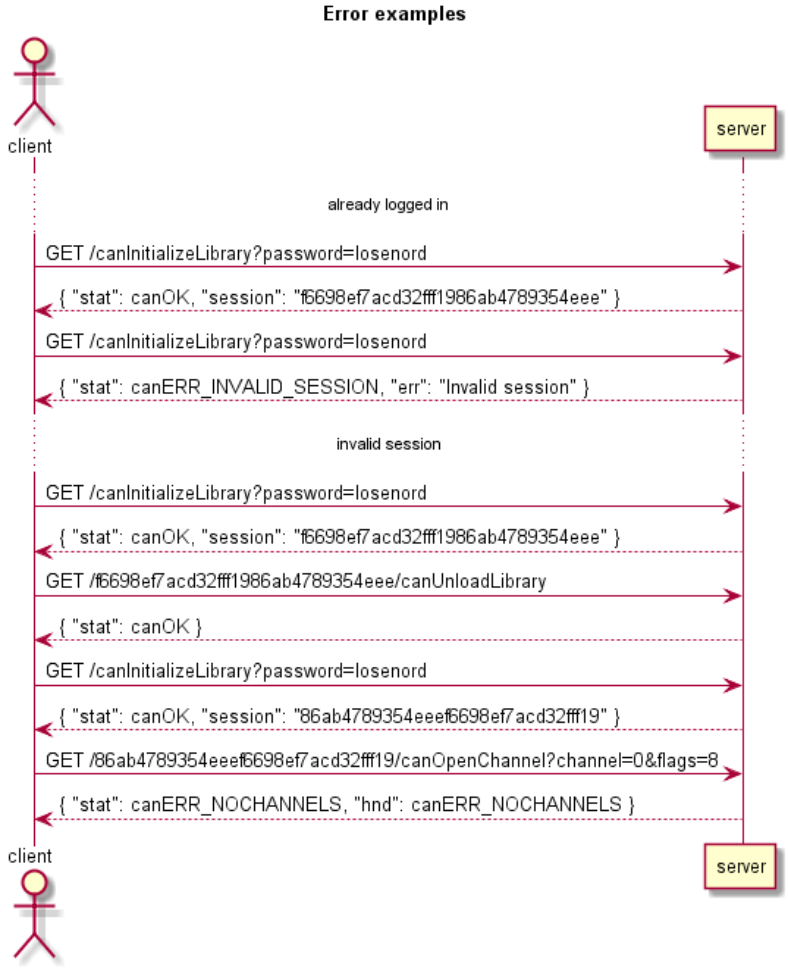

Figure 7: Error examples

6 Document Revision History

Version history for document IN_98151_kvaser_rest_api_specification:

Revision	Date	Changes
-	2013-09-23	Initial version
	2013-11-28	Changed layout of references, figures.
1	2014-08-19	Minor updates. First public version.
1.1	2014-10-02	Added description of dummy_session.