ILP 622 LAN
DLL-API-Dokumentation

Rev1.0

1. **int jmLaserEnumerateDevices ()**
   Enumerate all output devices.

   This function creates or refreshes the list of all output devices that are connected locally or via network.
   The device list length is returned.

   Returns:
   The number of devices found, which equal the device list length. A value less then 0 is returned on error.

2. **int jmLaserGetDeviceListEntry (unsigned int index, char deviceName, unsigned int length)**
   Get an entry from the devices list.

   Use this function to retrieve an entry from the devices list. A devices list entry is a null terminated ASCII/UTF-8 encoded string, that uniquely indentifies an output device. This might be "NetPort 0815#0" for a NetPort Board, for instance.
   The entry at the specified index is retrieved and placed into the deviceName buffer allocated by the application.
   Not more then length bytes will be written to the buffer. If the entry is longer then length bytes then it will be truncated.
   You can use jmLaserGetDeviceListEntryLength() to query in advance what size the buffer needs to be to hold the device list entry at a specified index.
   The jmLaserEnumerateDevices() must have been called at least once prior to calling this function.

   Parameters:

index	The index of the device list. Valid range is 0 to the length returned by jmLaserGetDeviceListEntryLength() - 1
deviceName	Buffer into which the device list entry will be placed. Has to be allocated by the application.
length	The length of the deviceName buffer including terminating zero.

   Returns:
   Returns 0 on success.
   A negative value indicates an error. Possible error codes include:
   • JMLASER_ERROR_NOT_ENUMERATED
   jmLaserEnumerateDevices() has not been called.
   • JMLASER_ERROR_OUT_OF_RANGE
   index is out of range

3. **int jmLaserGetDeviceListEntryLength (unsigned int index)**
Get length of a devices list entry.

This function may be used to determine what size the buffer for jmLaserGetDeviceListEntry() needs to be to retrieve the device list entry at the specified index.
The jmLaserEnumerateDevices() must have been called at least once prior to calling this function.

Parameters:
index The index of the device list. Valid range is 0 to the length returned by jmLaserGetDeviceListLength() - 1

Returns:
Returns the length of the device list entry at index. A negative value indicates an error. Possible error codes include:
• JMLASER_ERROR_NOT_ENUMERATED
jmLaserEnumerateDevices() has not been called.
• JMLASER_ERROR_OUT_OF_RANGE
index is out of range

4. **int jmLaserGetDeviceName (int handle, char _ deviceName, unsigned int length)**
Get device name.

This function can be used to get the unique device name for an already open output device handle. This is the same name as has been passed to jmLaserOpenDevice() when opening device and as returned by jmLaserGetDeviceListEntry(). As this name is unique and does not change for when the application is run again, it can be stored in the application configuration to open the same device again next time.
This function reads the device name into the buffer pointed to by deviceName.
At most length bytes are written. If the device name is longer then length bytes it will be truncated. You can get the length of the device name with jmLaserGetDeviceNameLength() before calling this function.
The output device must have been opened with jmLaserOpenDevice() prior to calling this function. Additionally jmLaserEnumerateDevices() must have been called at least once.

Parameters:
handle A handle for an open output device as returned by jmLaserOpenDevice().
deviceName Buffer that the device's friendly name string will be placed into.
length Length of the deviceFriendlyName Buffer.

Returns:
Returns 0 on success. A negative value indicates an error. Possible error codes include:
• JMLASER_ERROR_NOT_ENUMERATED
jmLaserEnumerateDevices() has not been called.
• JMLASER_ERROR_INVALID_HANDLE
The handle parameter is invalid.
• JMLASER_ERROR_DEVICE_NOT_OPEN
The device has not been opened with jmLaserOpenDevice() or has been removed.

5. **int jmLaserGetDeviceNameLength (int handle)**
Get device name length.

This function may be called before jmLaserGetDeviceName() to get the length of the device's name.
The output device must have been opened with jmLaserOpenDevice() prior to calling this function. Additionally jmLaserEnumerateDevices() must have been called at least once.

Parameters:
handle A handle for an open output device as returned by jmLaserOpenDevice().

Returns:
Returns the length of the device name string, including terminating zero. A negative value indicates an error. Possible error codes include:
• JMLASER_ERROR_NOT_ENUMERATED
jmLaserEnumerateDevices() has not been called.
• JMLASER_ERROR_INVALID_HANDLE
The handle parameter is invalid.
• JMLASER_ERROR_DEVICE_NOT_OPEN
The device has not been opened with jmLaserOpenDevice() or has been removed.

6. **int jmLaserGetDeviceFamilyName (char _ deviceName, char deviceFamilyName, _ unsigned int length)**
Get device family name.

This function reads the device family name into the buffer pointed to by deviceFamily. If the device family name is longer then length bytes it will be truncated. You can get the length of the family name with jmLaserGetFamilyNameLength() before calling this function. This function takes the device name as returned by jmLaserGetDeviceName() or jmLaserGetDeviceListEntry() as argument. The device does not need to be open.
An Example of a family name is "Netport".
The jmLaserEnumerateDevices() must have been called at least once prior to calling this function.

Parameters:
deviceName A device name string as returned by jmLaserGetDeviceName() or jmLaserGetDeviceListEntry().
deviceFamilyName Buffer that the device's family name string will be placed into.
length Length of the deviceFamilyName Buffer.

Returns:
Returns 0 on success. A negative value indicates an error. Possible error codes include
• JMLASER_ERROR_NOT_ENUMERATED
jmLaserEnumerateDevices() has not been called.
• JMLASER_ERROR_DEVICE_NOT_FOUND
A device with the name deviceName was not found.

7. **int jmLaserSetFriendlyName (int handle, char _ deviceFriendlyName)**
Set user-friendly device name.

This function will set the device's friendly name to the ASCII/UTF-8 encoded string the buffer device-FriendlyName points to.
The friendly name adds a description to the hard to remember device ID.
It is possible, for example, to set a friendly name "Projector left of table" for the device "NetPort". The friendly name should be displayed to the user next to the device id.
The friendly name can be read using jmLaserGetFriendlyName() and jmLaserGetFriendlyNameLength().

Parameters:

handle	A handle for an open output device as returned by jmLaserOpenDevice().
deviceFriendlyName	Buffer that contains the device's friendly name string. This string is ASCII/UTF-8 encoded and zero-terminated.

Returns:
Returns 0 on success. A negative value indicates an error. Possible error codes include
- JMLASER_ERROR_NOT_ENUMERATED
  jmLaserEnumerateDevices() has not been called.
- JMLASER_ERROR_INVALID_HANDLE
  The handle parameter is invalid.
- JMLASER_ERROR_DEVICE_NOT_OPEN
  The device has not been opened with jmLaserOpenDevice() or has been removed.

8. **int jmLaserGetFriendlyNameLength (char _ deviceName)**
Get user-friendly device name length.

This function may be called before jmLaserGetFriendlyName() to get the length of the device's friendly name. This function takes the device name as returned by jmLaserGetDeviceName() or jmLaserGetDeviceListEntry() as argument. The device does not need to be open.
The jmLaserEnumerateDevices() must have been called at least once prior to calling this function.

Parameters:

deviceName	A device name string as returned by jmLaserGetDeviceName() or jmLaserGetDeviceListEntry().

Returns:
Returns the length of the device friendly name string, including terminating zero.
This value may be 0. A negative value indicates an error. Possible error codes include:
- JMLASER_ERROR_NOT_ENUMERATED
  jmLaserEnumerateDevices() has not been called.
- JMLASER_ERROR_DEVICE_NOT_FOUND
  A device with the name deviceName was not found.

9. **int jmLaserGetFriendlyName (char _ deviceName, char _ deviceFriendlyName, unsigned int length)**
Get user-friendly device name.

This function reads the device friendly name into the buffer pointed to by deviceName. If the device name is longer then length bytes it will be truncated. You can get the length of the friendly name with jmLaserGetFriendlyNameLength() before calling this function.
This function takes the device name as returned by jmLaserGetDeviceName() or jmLaserGetDeviceListEntry() as argument. The device does not need to be open.
The friendly name adds a description to the hard to remember device ID. It is possible, for example, to set a friendly name "Projector left of table" for the device "NetPort". The friendly name should be displayed to the user next to the device id.
The friendly name can be set using jmLaserSetFriendlyName().
The jmLaserEnumerateDevices() must have been called at least once prior to calling this function.

Parameters:

deviceName	A device name string as returned by jmLaserGetDeviceName() or jmLaserGetDeviceListEntry().
deviceFriendlyName	Buffer that the device's friendly name string will be placed into.
length	Length of the deviceFriendlyName Buffer.

Returns:
Returns 0 on success. A negative value indicates an error. Possible error codes include
- JMLASER_ERROR_NOT_ENUMERATED
  jmLaserEnumerateDevices() has not been called.
- JMLASER_ERROR_DEVICE_NOT_FOUND
  A device with the device name was not found.

10. int jmLaserOpenDevice (char _ deviceName)

Open output device.

This function is used to open an output device. The output device needs to be opened before any of the Laser Output or Input/Output functions can be used.
This function takes a unique device name as returned by jmLaserGetDeviceName() or jmLaserGetDeviceListEntry() as its argument. As this name is unique and does not change for when the application is run again, it can be stored in the application configuration to open the same device again next time. When the device is not present any more, you will receive an JMLASER_ERROR_DEVICE_NOT_FOUND Error, though.
Only open the devices that you are going to use in your application. If you are done with the device, close it using jmLaserCloseDevice().

Parameters:

deviceName A device name string as returned by jmLaserGetDeviceName() or jmLaserGetDeviceListEntry().

Returns:

Returns a device handle on success. A negative value indicates an error. Possible error codes include

- JMLASER_ERROR_NOT_ENUMERATED
  jmLaserEnumerateDevices() has not been called.
- JMLASER_ERROR_DEVICE_NOT_FOUND
  A device with the name deviceName was not found.
- JMLASER_ERROR_DEVICE_BUSY
  The device could not be opened because it is in use by another application.
- JMLASER_ERROR_IO
  IO Error

11. int jmLaserCloseDevice (int handle)

Close output device.

Closes the output device the handle points to. After this call the handle will become invalid. The output device will be closed and accessible again by other applications.

Parameters:

handle A handle for an open output device as returned by jmLaserOpenDevice().

Returns:

Returns 0 on success. A negative value indicates an error. Possible error codes include

- JMLASER_ERROR_NOT_ENUMERATED
  jmLaserEnumerateDevices() has not been called.
- JMLASER_ERROR_INVALID_HANDLE
  The handle parameter is invalid.
- JMLASER_ERROR_DEVICE_NOT_OPEN
  The device has not been opened with jmLaserOpenDevice() or has been removed.
- JMLASER_ERROR_IO
  IO Error

12. int jmLaserStartOutput (int handle)

Start the laser output.

Call this function to start the laser output. If the laser output is already running this function does nothing.
The output device must have been opened with jmLaserOpenDevice() prior to calling this function. Additionally jmLaserEnumerateDevices() must have been called at least once.

Parameters:

handle A handle for an open output device as returned by jmLaserOpenDevice().

Returns:

This function returns 0 on success. A negative value indicates an error. Possible error codes include

- JMLASER_ERROR_NOT_ENUMERATED
  jmLaserEnumerateDevices() has not been called.
- JMLASER_ERROR_INVALID_HANDLE
  The handle parameter is invalid.
- JMLASER_ERROR_DEVICE_NOT_OPEN
  The device has not been opened with jmLaserOpenDevice() or has been removed.
- JMLASER_ERROR_IO
  IO Error

13. int jmLaserWriteFrame (int handle, JMVectorStruct *vectors, unsigned int count, _ unsigned int speed, unsigned int repetitions)

Write a frame to the output device.

Call this function to write a number of vectors to the output device for laser output.
The vectors are stored in an array of JMVectorStruct data structures. The number of vectors that should be written is provided in the count parameter. The speed parameter specifies the sampling rate with which the array of vectors should be output.
The maximum number of vectors in a frame is limited; its value differs among different types of output devices. Use jmLaserGetFrameSize() to query the maximum number of vectors that a frame may contain.
How often array of vectors is output can be set with the repetitions parameter. A value of 0 means, this frame will be output at least once, then repeated indefinitely until a new frame is written. A value larger then 0 will cause the frame to be displayed exactly the specified number of times, then the output will stop until a new frame is written.
A frame can only be written if the output device has a free buffer. Use jmLaserWaitForDeviceReady() or jmLaserIsDeviceReady() to query, whether the device can accept a new frame.
The output device must have been opened with jmLaserOpenDevice() and the laser output must have been started with jmLaserStartOutput() prior to calling this function. Additionally jmLaserEnumerateDevices() must have been called at least once.

Parameters:

handle	A handle for an open output device as returned by jmLaserOpenDevice().
vectors	An array of JMVectorStruct data structures describing the vector data. The array has count entries.
count	Count of vectors to be written to the device. Range is 1.. the value returned by jmLaserGet-FrameSize().
speed	Sampling rate in points per seconds with which the frame should be output.
repetitions	number of times the frame should be repeated. If this value is 0, the frame will be display at least once and repeat indefinitely until a new frame is written. If this value is larger then 0 the frame will be shown exactly the specified number of times and then the output will stop until a new frame is written.

Returns:

This function returns 0 on success. A negative value indicates an error. Possible error codes include

- JMLASER_ERROR_NOT_ENUMERATED
  jmLaserEnumerateDevices() has not been called.
- JMLASER_ERROR_INVALID_HANDLE
  The handle parameter is invalid.
- JMLASER_ERROR_DEVICE_NOT_OPEN
  The device has not been opened with jmLaserOpenDevice() or has been removed.
- JMLASER_ERROR_OUTPUT_NOT_STARTED
  The laser output has not been started (successfully) with jmLaserStartOutput().
- JMLASER_ERROR_DEVICE_BUSY
  Frame could not be written because device is busy.
- JMLASER_ERROR_IO
  IO Error

14. Struct reference: JMVectorStruct

Public Attributes:
- int x
- int y
- unsigned short r
- unsigned short g
- unsigned short b
- unsigned short i
- unsigned short deepblue
- unsigned short yellow
- unsigned short cyan
- unsigned short user4

16bit hardware resolution X/Y deflection value.
Represented in 32bit range: -2147483648 - 2147483647

15. int jmLaserWaitForDeviceReady (int handle)

Block thread until device can accept a new frame.

This function blocks the calling thread until jmLaserWriteFrame() is ready to accept a new frame. Use this function in a multi-threaded application when you have one thread for every output device.
The output device must have been opened with jmLaserOpenDevice() and the laser output must have been started with jmLaserStartOutput() prior to calling this function. Additionally jmLaserEnumerateDevices() must have been called at least once.

Parameters:

handle A handle for an open output device as returned by jmLaserOpenDevice().

Returns:

This function returns JMLASER_DEVICE_READY on success. A negative value indicates an error. Possible error codes include:
- JMLASER_ERROR_NOT_ENUMERATED
  jmLaserEnumerateDevices() has not been called.
- JMLASER_ERROR_INVALID_HANDLE
  The handle parameter is invalid.
- JMLASER_ERROR_DEVICE_NOT_OPEN
  The device has not been opened with jmLaserOpenDevice() or has been removed.
- JMLASER_ERROR_OUTPUT_NOT_STARTED
  The laser output has not been started (successfully) with jmLaserStartOutput().
- JMLASER_ERROR_IO
  IO Error

16. int jmLaserIsDeviceReady (int handle)

Query if the device can accept a new frame.

This function is used to query if the output device will be ready to accept a new frame with jmLaserWriteFrame(). Use this function in a single-threaded application to poll whether an output device can accept a new frame. If this is the case, JMLASER_DEVICE_READY is returned, else JMLASER_ERROR_DEVICE_BUSY will be returned.
The output device must have been opened with jmLaserOpenDevice() and the laser output must have been started with jmLaserStartOutput() prior to calling this function. Additionally jmLaserEnumerateDevices() must have been called at least once.

Parameters:

handle A handle for an open output device as returned by jmLaserOpenDevice().

Returns:

This function returns JMLASER_DEVICE_READY or JMLASER_ERROR_DEVICE_BUSY on success. A negative value indicates an error.
Possible error codes include:

- JMLASER_ERROR_NOT_ENUMERATED
  jmLaserEnumerateDevices() has not been called.
- JMLASER_ERROR_INVALID_HANDLE
  The handle parameter is invalid.
- JMLASER_ERROR_DEVICE_NOT_OPEN
  The device has not been opened with jmLaserOpenDevice() or has been removed.
- JMLASER_ERROR_OUTPUT_NOT_STARTED
  The laser output has not been started (successfully) with jmLaserStartOutput().
- JMLASER_ERROR_IO
  IO Error

17. int jmLaserStopOutput (int handle)

Stop the laser output.

Call this function just to stop the laser output.
The output device must have been opened with jmLaserOpenDevice() and the laser output must have been started with jmLaserStartOutput() prior to calling this function.
Additionally jmLaserEnumerateDevices() must have been called at least once.

Parameters:

handle A handle for an open output device as returned by jmLaserOpenDevice().

Returns:

This function returns 0 on success. A negative value indicates an error. Possible error codes include

- JMLASER_ERROR_NOT_ENUMERATED
  jmLaserEnumerateDevices() has not been called.
- JMLASER_ERROR_INVALID_HANDLE
  The handle parameter is invalid.
- JMLASER_ERROR_DEVICE_NOT_OPEN
  The device has not been opened with jmLaserOpenDevice() or has been removed.
- JMLASER_ERROR_OUTPUT_NOT_STARTED
  The laser output has not been started (successfully) with jmLaserStartOutput().
- JMLASER_ERROR_IO
  IO Error

18. int jmLaserCloseDll ()

DLL de-initialisation.

Call jmLaserCloseDll once just before you close the DLL or exit you application program.

Returns:

0 on success, a value less then 0 on error.

19. DLL Error Codes Defines

- #define JMLASER_ERROR_NOT_ENUMERATED -1
- #define JMLASER_ERROR_INVALID_HANDLE -2
- #define JMLASER_ERROR_DEVICE_NOT_FOUND -4
- #define JMLASER_ERROR_DEVICE_NOT_OPEN -3
- #define JMLASER_ERROR_OUTPUT_NOT_STARTED -5
- #define JMLASER_ERROR_INVALID_UNIVERSE -6
- #define JMLASER_ERROR_OUT_OF_RANGE -7
- #define JMLASER_ERROR_DEVICE_BUSY -8
- #define JMLASER_ERROR_IO -9

12.1 Detailed Description

These are the Error Codes that are returned by the various API functions.

12.1.1. #define JMLASER_ERROR_DEVICE_BUSY -8
Error when the device is busy.

12.1.2. #define JMLASER_ERROR_DEVICE_NOT_FOUND -4
Error when a device with the given name was not found.

12.1.3. #define JMLASER_ERROR_DEVICE_NOT_OPEN -3
Error when the device is not open or was removed.

12.1.4. #define JMLASER_ERROR_INVALID_HANDLE -2
Error when the passed handle value is invalid.

12.1.5. #define JMLASER_ERROR_INVALID_UNIVERSE -6
Error when the universe index is invalid.

12.1.6. #define JMLASER_ERROR_IO -9
Error when an IO Error occured.

12.1.7. #define JMLASER_ERROR_NOT_ENUMERATED -1
Error when jmLaserEnumerateDevices() was not called.

12.1.8. #define JMLASER_ERROR_OUT_OF_RANGE -7
Error when a value/index is out of range.

12.1.9. #define JMLASER_ERROR_OUTPUT_NOT_STARTED -5
Error when the laser output was not started