OpenML V1.0 Specification – Errata
May 2004

This document lists errors and omissions in the “OpenML™ V1.0 Specification” (dated 19 July 2001).

---

**Page 12 – Video Back-end Device Control**
Modify the second sentence to:

It is based on **XSG1ve**, an extension to the X Window System designed by SGI.

**Page 32 – After Table 5.1**
Modify the first sentence to:

…where the *type* suffix is one of **BYTE, INT32, INT64**…

**Pages 33 and 34 – Array Parameters**
Replace “ML_PATH_LUT_REAL64_ARRAY” by “ML_TYPE_REAL64_ARRAY”, and replace “ML_PATH_LUT_INT32_ARRAY” by “ML_TYPE_INT32_ARRAY”.

**Page 39 – Physical Device Capabilities, Table 6.2**
The parameter “DEVICE_INDEX” is of type “**INT32**” (not “**BYTE_ARRAY**”)

**Page 41 – Jack Logical Device Capabilities, Table 6.3**
The parameter “**JACK_DIRECTION**” may also take the value “**ML_JACK_DIRECTION_BOTH**”.

**Page 44 – Obtaining Parameter Capabilities**
The call to “mlPvGetCapabilities” may also return the status code “**ML_STATUS_OUT_OF_MEMORY**”, if the system was not able to allocate sufficient memory for the capabilities array.

**Page 45 – Table 6.7**
In the description for the parameter “**PARAM_TYPE**”, add the following possible value: “**ML_TYPE_BYTE**”.
Page 46 – Table 6.7
In the description for the parameter “PARAM_ENUM_NAMES”, disregard the description for the length of the parameter (second and third lines). Add the following text at the end of the paragraph:
   The length of the parameter is the total length of all the strings, including the NULL separators, and the double-NUL terminator.

Page 58 – Examples
Replace the last line of the example by:
   mlSetControls( someOpenPath, message );

Page 62 – ML_IMAGE_COMPRESSION_INT32
In the third paragraph from the bottom, replace
   ML_IMAGE_SIZE_INT32
by
   ML_IMAGE_BUFFER_SIZE_INT32

Page 75 – ML Program Structure
In the section on querying for individual parameter characteristics, modify the code to:
   mlPvGetCapabilities( objectid, &capabilities );

Page 82 – Set Controls
On the third line, replace “ML_IMAGE_PACKING_IN32” by “ML_IMAGE_PACKING_INT32”.

Page 82 – Get Controls
The “mlGetControls” function may also return the ML_STATUS_INVALID_VALUE error code, if the “controls” argument is not correctly constructed. For instance, if the message contains a control requiring an array, and the array pointer is NULL, the function will return INVALID_VALUE.

Page 84 – Send Buffers
On the first line, replace
   buffers is enqueued…
by
   buffers are enqueued…
Page 84 – Query Controls
Modify the first sentence of the second paragraph to:

`openid` is the identifier, returned by `mlOpen`, of the device whose parameters are to be queried.

Page 87 – Receive Message
The third argument to the “mlReceiveMessage” function is of type “MLpv**”. Thus, the correct prototype for this function is:

```c
MLstatus mlReceiveMessage( Mlopenid openid, MLint32* messageType, MLpv** message );
```

Page 89 – Table 10.7, mlQueryControls Reply Message Types
Add the following message type for replies to “mlQueryControls”:

`ML_QUERY_CONTROLS_FAILED`: Processing of the query controls was aborted due to an error.

Page 91 – Message Name
The last sentence of the description should be:

`NULL` is returned if `messageType` is an invalid message type.

Page 98 – ML_IF_VIDEO_UST_LT
Add the “_INT64” suffix to both control names (e.g., `ML_IF_VIDEO_UST_LT_INT64`).

Add two new controls:

```c
ML_IF_VIDEO_UST_GT_INT64
ML_IF_AUDIO_UST_GT_INT64
```

These controls behave in the same way as the “_UST_LT_INT64” controls, except that the device processes the message only if the UST is greater than the specified time.

Page 117 – mldcQueryAvailableDevices
The second sentence at the top of the page (lines 1—2) should read:

If the `systemHandle` argument is not a valid system handle, …

Page 117 – mldcOpen
The function may fail with the status “MLDC_STATUS_TOO_MANY_OPEN_DEVICES”.

Page 121 – Setting Parameters, Table 15.1
Change the event message type “MLDC_INPUT_SYNCSOURCE_NOTIFY” (2 occurrences) to “MLDC_INPUT_SYNC_SOURCE_NOTIFY”.

Page 123 – mldcSetEventMask
Change the event mask name “MLDC_INPUT_SYNCSOURCE_NOTIFY_MASK” to “MLDC_INPUT_SYNC_SOURCE_NOTIFY_MASK”

Page 126 – mldcSetWindowsMessageQueue
The second-last paragraph should be:
This function is only viable on a Windows system. On a system using another windowing system, this function is not defined.

Page 128 – MLdc Event Message Structures
Make the following corrections to the event type names:
MLDC_VIDEOFORMAT_NOTIFY → MLDC_VIDEO_FORMAT_NOTIFY
MLDC_CHANNEL_INPUTRECTANGLE_NOTIFY → MLDC_CHANNEL_INPUT_RECTANGLE_NOTIFY
MLDC_INPUT_SYNCSOURCE_NOTIFY → MLDC_INPUT_SYNC_SOURCE_NOTIFY

The event types are specified using the data type “MLDCEventType”, rather than “MLDCint32”. Replace all occurrences of:
MLDCint32 mldcType;
by:
MLDCEventType mldcType;

Page 144 – mldcListVideoFormats
The return type of this function should be “MLDCstatus” (rather than “MLDCstatus*”).

Page 155 – mldcQueryGammaColors
The “requestedComponent” argument to this function is of type “MLDCbitfield” rather than “MLDCint32”.

Page 157 – mldcStoreGammaColors16
In the description of the “loadTables” argument, the first sentence should be:
Specifies the tables or color components that should be loaded.

Page 158 – mldcSetChannelGammaMap
In the description of the function arguments, the third argument should be “gammaMap”.
Page 159 – Output Gain
The last two sentences of the paragraph should be:

    The MLDC_CIF_PER_COMPONENT_GAIN in the channelFlag value returned by mldeQueryChannelInfo indicates the capability of the device. This flag is set when the device supports independent gain adjustment of each color component.

Page 159 – mldeSetOutputGain
The “componentID” argument to this function is of type “MLDCbitfield” rather than “MLDCint32”.

Page 160 – mldeQueryOutputGain
The “componentID” argument to this function is of type “MLDCbitfield” rather than “MLDCint32”.

Page 180 – mldeQueryMonitorName
If no monitor is connected, or if it does not support the command, the function will return “MLDC_STATUS_NO_MONITOR” (rather than “NO_MONITOR_NAME”).

Page 181 – mldeSendMonitorCommand
If no monitor is connected, or if it does not support the command, the function will return “MLDC_STATUS_NO_MONITOR”.