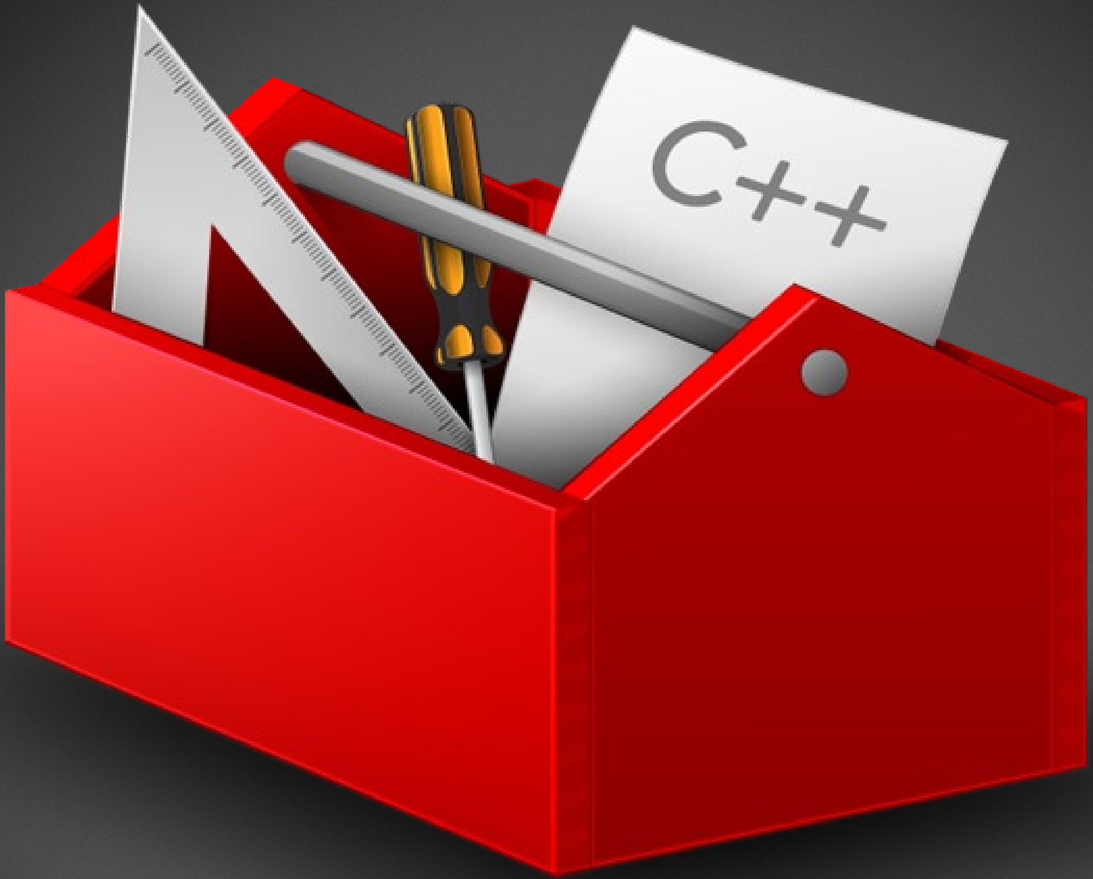
SDK – HDLink
Software Developers Kit

Blackmagicdesign



Mac OS X™
Windows™
November 2009

TABLE OF CONTENTS

1 Introduction 4

1.1 Welcome 4
1.2 Overview 4
1.3 API Design 5
1.3.1 Overview 5
1.3.2 Object Model 5
1.3.3 Object Interfaces 5
1.3.4 Reference Counting 6
1.4 Interface Reference 6
1.4.1 IUnknown 6
1.4.1.1 IUnknown::QueryInterface method 7
1.4.2 IUnknown::AddRef method 8
1.4.3 IUnknown::Release method 8

2 HDLink API 9

2.1 Introduction 9
2.1.1 Overview 9
2.1.2 Installation. 10
2.2 Common data types 10
2.3 Class reference 11
2.3.1 CHDLinkDiscovery class 11
2.4 Interfaces reference 11
2.4.1 IHDLinkDiscovery Interface 11
2.4.1.1 IHDLinkDiscovery::CreateHDLinkIterator method 12
2.4.1.2 IHDLinkDiscovery::InstallDeviceAddedNotification method 13
2.4.1.3 IHDLinkDiscovery::UninstallDeviceAddedNotification method 14
2.4.2 IHDLinkIterator Interface 15
2.4.2.1 IHDLinkIterator::Next method 15
2.4.3 IHDLink Interface 16
2.4.3.1 IHDLink::GetModelName method 16
2.4.3.2 IHDLink::InstallDeviceRemovalNotification method 17
2.4.3.3 IHDLink::UninstallDeviceAddedNotification method 18

TABLE OF CONTENTS

2.4.4 I3DLookupTableConfiguration Interface 18
2.4.4.1 I3DLookupTableConfiguration::ReadActiveTable method 19
2.4.4.2 I3DLookupTableConfiguration::WriteTable method 20
2.4.4.3 I3DLookupTableConfiguration::ResetTable method 21
2.4.5 IHDLinkDeviceArrivalCallback interface 21
2.4.5.1 IHDLinkDeviceArrivalCallback::HDLinkDeviceArrival method 22
2.4.6 IHDLinkDeviceRemovalCallback 22
2.4.6.1 IHDLinkDeviceRemovalCallback::HDLinkDeviceRemoval method 23
2.4.7 IHDLinkAttributes Interface 24
2.4.7.1 IHDLinkAttributes::GetFlag method 25
2.4.7.2 IHDLinkAttributes::GetInt method 26
2.4.7.3 IHDLinkAttributes::GetFloat method 27
2.4.7.4 IHDLinkAttributes::GetString method 28
2.4.8 IHDLinkConfiguration interface 29
2.4.8.1 IHDLinkConfiguration::GetFlag method 30
2.4.8.2 IHDLinkConfiguration::SetFlag method 31
2.4.8.3 IHDLinkConfiguration::GetInt method 32
2.4.8.4 IHDLinkConfiguration::SetInt method 33
2.4.8.5 IHDLinkConfiguration::GetFloat method 34
2.4.8.6 IHDLinkConfiguration::SetFloat method 35
2.4.8.7 IHDLinkConfiguration::GetString method 36
2.4.8.8 IHDLinkConfiguration::SetString method 37
2.4.8.9 HDLink Attribute ID 38
2.4.8.10 HDLink Configuration ID 38
2.5 Common Data Types 39
2.5.1 Basic Types 39

SECTION 1

INTRODUCTION

1.1 Welcome

Thanks for downloading the Blackmagic Design Software Developers Kit.

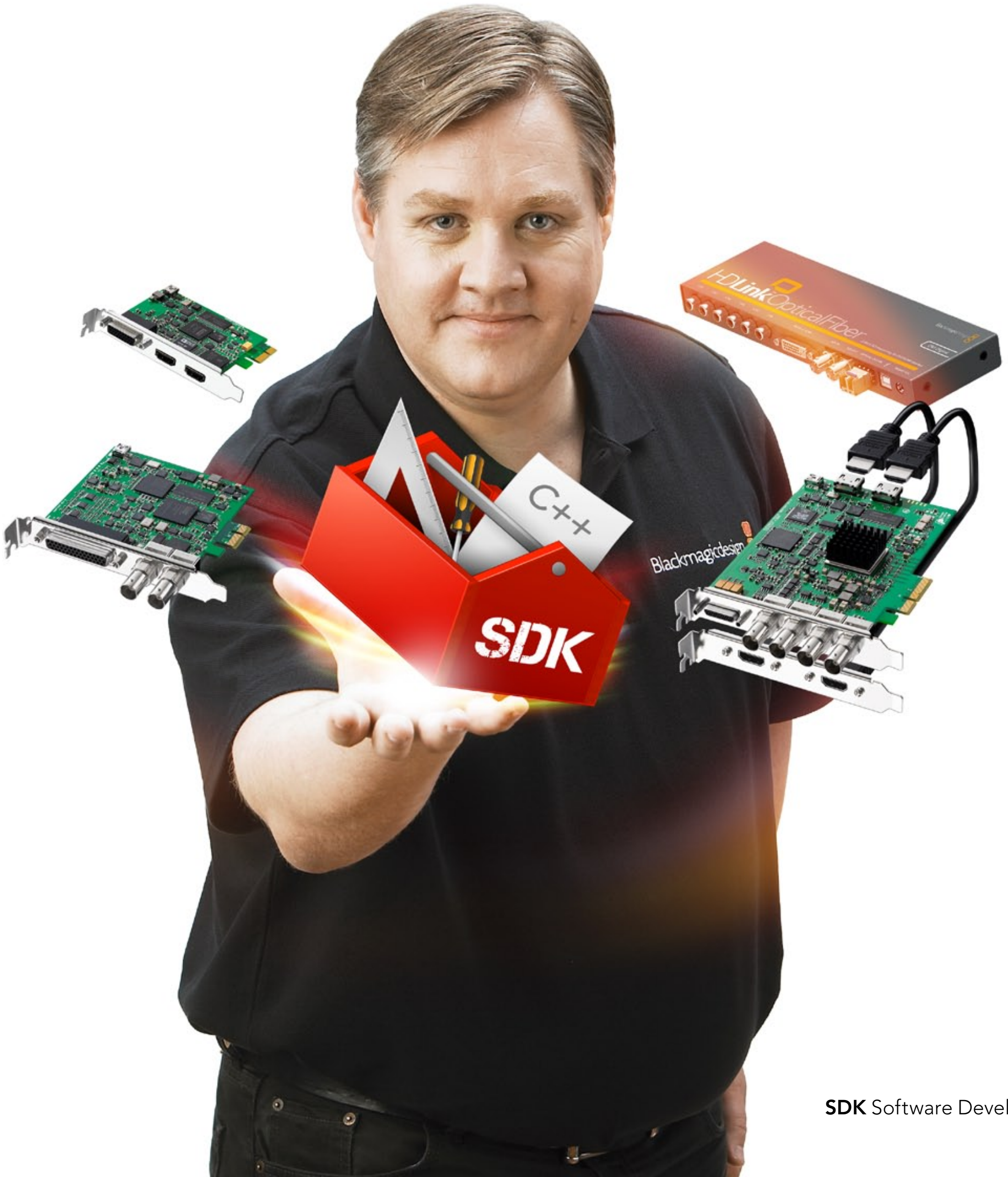
1.2 Overview

The HDLink SDK provides a stable, cross-platform interface to the Blackmagic Design HDLink product line.

The SDK consists of a set of interface descriptions & sample applications which demonstrate the use of the basic features of the hardware. The details of the SDK are described in this document.

The SDK supports Microsoft Windows and Mac OS X platforms.

You can download the SDK from **www.blackmagic-design.com/support**



1.3.1 Overview

The libraries supporting the Blackmagic SDK are shipped as part of the product installers for each supported product line. Applications built against the interfaces shipped in the SDK will dynamically link against the library installed on the end-user's system.

1.3.2 Object Model

The SDK interface is modeled on Microsoft's Component Object Model (COM).
On Microsoft Windows platforms, it is provided as a native COM interface registered with the operating system. On other platforms application code is provided to allow the same COM style interface to be used.

The COM model provides a paradigm for creating flexible and extensible interfaces without imposing much overhead or baggage.

1.3.3 Object Interfaces

The HDLink API provides programmatic access to all current Blackmagic Design HDLink products.

The API provides high-level interfaces to allow 3D LUT configuration as low-level interfaces for controlling features available on different HDLink models.

Functionality within the API is accessed via "object interfaces". Each object in the system may inherit from and be accessed via a number of object interfaces. Typically the developer is able to interact with object interfaces and leave the underlying objects to manage themselves.

Each object interface class has a Globally Unique ID (GUID) called an "Interface ID".
On platforms with native COM support, an IID may be used to obtain a handle to an exported interface object from the OS, which is effectively an entry point to an installed API.

Each interface may have related interfaces that are accessed by providing an IID to an existing object interface (see IUnknown::QueryInterface). This mechanism allows new interfaces to be added to the API without breaking API or ABI compatibility.

1.3.4 Reference Counting

The API uses reference counting to manage the life cycle of object interfaces. The developer may need to add or remove references on object interfaces (see **IUnknown::AddRef** and **IUnknown::Release**) to influence their life cycle as appropriate in the application.

1.4 Interface Reference

Every object interface inherits basic functionality from the **IUnknown** interface.

1.4.1 IUnknown

The DeckLink API provides a number of object interfaces for use by developers. Each interface is a subclass of the standard COM base class – **IUnknown**. The **IUnknown** object interface provides reference counting and the ability to look up related interfaces by interface ID. The interface ID mechanism allows interfaces to be added to the API without impacting existing applications.

Public Member Functions

Method	Description
QueryInterface	Provides access to supported child interfaces of the object.
AddRef	Increments the reference count of the object.
Release	Decrements the reference count of the object. When the final reference is removed, the object is freed.

1.4.1.1 IUnknown::QueryInterface method

The **QueryInterface** method looks up a related interface of an object interface.

Syntax

```
HRESULT       QueryInterface(REFIID id, void **outputInterface);
```

Parameters

Name	Direction	Description
`id`	in	Interface ID of interface to lookup
`outputInterface`	out	New object interface or NULL on failure.

Return Values

Value	Description
E_NOINTERFACE	Interface was not found
S_OK	Success

1.4.1.2 IUnknown::AddRef method

The **AddRef** method increments the reference count for an object interface.

Syntax

```
ULONG         AddRef();
```

Parameters

none.

Return Values

Value	Description
Count	New reference count – for debug purposes only.

1.4.1.3 IUnknown::Release method

The **Release** method decrements the reference count for an object interface. When the last reference is removed from an object, the object will be destroyed.

Syntax

```
ULONG         Release();
```

Parameters

none.

Return Values

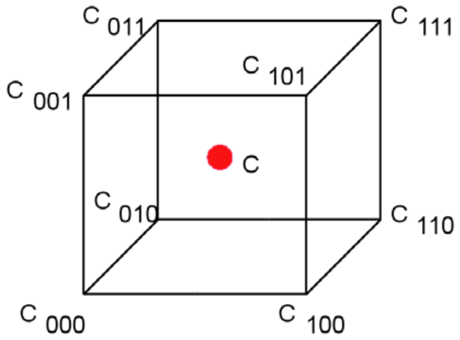
Value	Description
Count	New reference count – for debug purposes only.

HDLink API

2.1 Introduction

2.1.1 Overview

The HDLink API is a set of libraries, interface definitions documents and sample application source code.
It gives programmatic access to the 3D LUT configuration for the HDLink Pro line of products. 3D LUTs are lookup tables that are used to adjust color reproductionon output. A 3D LUT is defined as a 3-dimensional grid where each axis is an entry point for one of the three color components. The 3D LUT output value "C" is calculated by interpolating between the 8 nearest points in the grid.



3D LUT cubes may have various sizes and bit depth. HDLink Pro currently uses a 17 bit LUT. If the imported 3D LUT cube dimensions are different to the internal cube dimensions, the LUT will be resized.

2.1.2 Installation

The HDLink API consists of two parts: The HDLinkAPI library and the HDLink SDK. The HDLinkAPI library contains binary objects and distributed in the HDLink installation package as HDLinkAPI.dll for Windows and HDLink.bundle for Mac. The HDLink SDK is distributed in the Blackmagic Design SDK package and comprised of the required include files and sample applications source code for each platform.

2.2 Common Data Types

The LOOKUPTABLE_COLOR structure is used to store a RGB color value.

Syntax

```
typedef struct
{
    float     red;
    float     green;
    float     blue;

} LOOKUPTABLE_COLOR;
```

2.3 Class Reference

2.3.1 CHDLinkDiscovery class

The **CHDLinkDiscovery** object is the entry point into the HDLink API. The object is the first to be instantiated to start using HDLink API. CHDLinkDescovery class implements interfaces:

2.4 Interfaces reference

2.4.1 IHDLinkDiscovery Interface

The **IHDLinkDiscovery** interface is used to obtain all other HDLink API interfaces. The interface is the first to be obtained to start using HDLink API.

Public Member Functions

Method	Description
CreateHDLinkIterator	Creates HDLink iterator object
InstallDeviceAddedNotification	Install device added notification
UninstallDeviceAddedNotification	Uninstall device added notification

2.4.1.1 IHDLinkDiscovery::CreateHDLinkIterator method

The **CreateHDLinkIterator** method creates an HDLink iterator .

Syntax

```
HRESULT       CreateHDLinkIterator (IHDLinkIterator **iterator);
```

Parameters

Name	Direction	Description
`iterator`	in	Pointer to a pointer that receives address of newly created HDLink iterator object. If iterator creation failed the value will be NULL.

Return Values

Value	Description
E_FAIL	HDLink iterator creation failure.
IS_OK	Success

2.4.1.2 IHDLinkDiscovery::InstallDeviceAddedNotification method

The **InstallDeviceAddedNotification** method installs a "device added" notification callback. The callback is used to inform the host application about an HDLink connection event to a host.

Syntax

```
HRESULT       InstallDeviceAddedNotification
              (IHDLinkDeviceArrivalCallback *theCallback,
              HWND notificationWindow);
```

Parameters

Name	Direction	Description
`theCallback`	in	Pointer to the object of class implementing IHDLinkDeviceArrivalCallback interface.
`notificationwindow`	in	Handle to host application main window.

Return Values

Value	Description
E_FAIL	HDLink device callback registration failure
S_OK	Success

2.4.1.3 IHDLinkDiscovery::UninstallDeviceAddedNotification method

The **UninstallDeviceAddedNotification** uninstalls a previously installed "device added" notification. This method must be called to release the Device added callback notification object.

Syntax

```
HRESULT        UninstallDeviceAddedNotification ();
```

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success

2.4.2 IHDLinkIterator Interface

Public Member Functions	
Method	Description
Next	Creates an object representing the HDLink physical instance connected to the host and obtains pointer to IHDLink interface corresponding to that object.

2.4.2.1 IHDLinkIterator::Next method

The **Next** method creates an object representing the physical HDLink device and assigns the address of the IHDLink interface of the newly created object to the **hdlinkInstance** paramenter.

Syntax

```
HRESULT        Next (IHDLink **hdlinkInstance);
```

Parameters

Name	Direction	Description
`hdlinkInstance`	out	Pointer to a pointer that receives address of newly instantiated object implementing IHDLink interface, if object creation failed the value will be NULL.

Return Values

Value	Description
E_FAIL	HDLink create failure
S_OK	Success

2.4.3 IHDLink Interface

The **IHDLink** interface represents a physical HDLink device connected to the host computer and can be used to and query other HDLink device interfaces.

Public Member Functions

Method	Description
GetModelName	Method to get HDLink device model name
InstallDeviceRemovalNotification	Install Device Removal Notification to signal host application about physical HDLink device removal from host computer.
UninstallDeviceRemovalNotification	Uninstall Device Removal Notification.

2.4.3.1 IHDLink::GetModelName method

The **GetModelName** method can be used to get the HDLink device model name.

Syntax

```
HRESULT      GetModelName (string *modelName);
```

Parameters

Name	Direction	Description
`modelName`	out	Pointer to an object holding HDLink model name.

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success

2.4.3.2 IHDLink::InstallDeviceRemovalNotification method

The **InstallDeviceRemovalNotification** method is used to specify a callback to receive notification of physical device removal from the host computer.

Syntax

```
HRESULT      InstallDeviceRemovalNotification (
             IHDLinkDeviceRemovalCallback *theCallback,
             HWND notificationWindow);
```

Parameters

Name	Direction	Description
`theCallback`	in	Pointer to an object implementing `IHDLinkDeviceRemovalCallback` interface.
`notificationwindow`	in	Handle to host application main window.

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success

2.4.3.3 IHDLink::UninstallDeviceAddedNotification method

The **UninstallDeviceAddedNotification** method must be called to release a previously installed device added notification object.

Syntax

```
HRESULT       UninstallDeviceAddedNotification ();
```

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success

2.4.4 I3DLookupTableConfiguration Interface

I3DLookupTableConfiguration interface can be used to read, write and reset the 3D LUT to or from the HDLink device. The interface can be obtained by querying the IHDLink interface. If the HDLink Device does not support 3D LUTs, interface query will fail.

Public Member Functions

Method	Description
ReadActiveTable	Read 3D LUT table from HDLink device
WriteTable	Write 3D LUT table in to HDLink device
ResetTable	Reset LUT table

2.4.4.1 I3DLookupTableConfiguration::ReadActiveTable method

The **ReadActiveTable** method can be used to read the 3D LUT from the HDLink device.

Syntax

```
HRESULT       ReadActiveTable (LOOKUPTABLE_COLOR **table,
              unsigned long *cubeDimension, string *tableName);
```

Parameters

Name	Direction	Description
`table`	out	Pointer to an object holding HDLink device model name.
`cubeDimension`	out	Pointer to an unsigned long variable receiving value of the LUT cube dimension.
`tableName`	out	Pointer to an object holding LUT table name.

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success

2.4.4.2 I3DLookupTableConfiguration::WriteTable method

The **WriteTable** method can be used to write a 3D LUT table into an HDLink device. If `saveTableToNonvolatileStorage` argument is true, the table will be written into non-volatile memory.

Syntax

```
HRESULT       WriteTable (LOOKUPTABLE_COLOR *table,
              unsigned long cubeDimension, string *tableName, boolean saveTableToNonvolatileStorage);
```

Parameters

Name	Direction	Description
`table`	out	Pointer to a buffer holding 3D LookUp Table
`cubeDimension`	out	LUT cube dimension.
`tableName`	out	Pointer to an object holding LUT table name.
`saveTableToNonvolatileStorage`	in	A boolean value to enable (TRUE) or disable (FALSE) saving 3D LUT into nonvolatile storage.

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success

2.4.4.3 I3DLookupTableConfiguration::ResetTable method

The **ResetTable** method can be used to reset the 3D LUT table in an HDLink device. If `saveTableToNonvolatileStorage` argument is set true, the table will be reset in nonvolatile memory.

Syntax

```
HRESULT      ResetTable (boolean saveTableToNonvolatileStorage);
```

Parameters

Name	Direction	Description
`saveTableToNonvolatileStorage`	in	A Boolean value to enable (TRUE) or disable (FALSE) reset 3D LUT in to nonvolatile storage

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success

2.4.5 IHDLinkDeviceArrivalCallback interface

IHDLinkDeviceArrivalCallback interface can be used to notify application about new HDLink device been connected to a host computer.

Public Member Functions	
Method	Description
HDLinkDeviceArrival	Method defining actions at HDLink device been connected to host computer.

2.4.5.1 IHDLinkDeviceArrivalCallback::HDLinkDeviceArrival method

The **HDLinkDeviceArrival** method is called when an HDLink device is attached to the USB bus. It can be used to request notification when an HDLink device is connected to USB bus. That method is abstract virtual and must be defined by the application developer.

Syntax

```
HRESULT        HDLinkDeviceArrival (IHDLink *device);
```

Parameters

Name	Direction	Description
`device`	out	Pointer to IHDLink interface connected to USB bus.

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success

2.4.6 IHDLinkDeviceRemovalCallback

IHDLinkDeviceRemovalCallback interface can be used to notify the application when an HDLink device has been disconnected from the host computer. The method is abstract virtual and must be defined by the application developer.

Public Member Functions

Method	Description
HDLinkDeviceRemoval	Method defining action at HDLink device been removed from host computer.

2.4.6.1 IHDLinkDeviceRemovalCallback::HDLinkDeviceRemoval method

The **HDLinkDeviceRemoval** method is called when an HDLink device is disconnected from the USB bus. That method is abstract virtual and must be defined by the application developer.

Syntax

```
HRESULT        HDLinkDeviceRemoval (IHDLink *device);
```

Parameters

Name	Direction	Description
`device`	out	Pointer to IHDLink interface of HDLink device removed from USB bus.

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success

2.4.7 IHDLinkAttributes Interface

The **IHDLinkAttributes** object interface provides the capabilities of a HDLink device. The detail types that are available for various capabilities are: flag, int, float, and string. The HDLink Attribute ID section lists the hardware capabilities and associated attributes identifiers that can be queried using this object interface

An **IHDLinkAttributes** object interface can be obtained from the IHDLink interface using QueryInterface.

Related Interfaces

Interface	Interface ID	Description
IHDLink	IID_IHDLink	HDlink device interface

Public Member Functions

Method	Description
GetFlag	Gets a boolean flag corresponding to a **BMDHDLinkAttributeID**
GetInt	Gets an int64_t corresponding to a **BMDHDLinkAttributeID**
GetFloat	Gets a float corresponding to a **BMDHDLinkAttributeID**
GetString	Gets a string corresponding to a **BMDHDLinkAttributeID**

2.4.7.1 IHDLinkAttributes::GetFlag method

The **GetFlag** method gets a boolean flag associated with a given **BMDHDLinkAttributeID**.

Syntax

```
HRESULT       GetFlag (BMDHDLinkAttributeID cfgID, boolean *value);
```

Parameters

Name	Direction	Description
`cfgID`	in	The attribute ID.
`value`	out	The value corresponding to cfgID.

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success
E_INVALIDARG	There is no flag type attribute corresponding to cfgID.

2.4.7.2 IHDLinkAttributes::GetInt method

The **GetInt** method gets an int64_t value associated with a given **BMDHDLinkAttributeID**.

Syntax

```
HRESULT       GetInt (BMDHDLinkAttributeID cfgID, int64_t *value);
```

Parameters

Name	Direction	Description
`cfgID`	in	The attribute ID.
`value`	out	The value corresponding to cfgID.

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success
E_INVALIDARG	There is no int type attribute corresponding to cfgID.

2.4.7.3 IHDLinkAttributes::GetFloat method

The **GetFloat** method gets a float value associated with a given **BMDHDLinkAttributeID**.

Syntax

```
HRESULT      GetFloat (BMDHDLinkAttributeID cfgID, double *value);
```

Parameters

Name	Direction	Description
`cfgID`	in	The attribute ID.
`value`	out	The value corresponding to cfgID.

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success
E_INVALIDARG	There is no float type attribute corresponding to cfgID.

2.4.7.4 IHDLinkAttributes::GetString method

The **GetString** method gets a string value associated with a given **BMDHDLinkAttributeID**.

Syntax

```
HRESULT       GetString (BMDHDLinkAttributeID cfgID, String *value);
```

Parameters

Name	Direction	Description
`cfgID`	in	The attribute ID.
`value`	out	The value corresponding to cfgID.

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success
E_INVALIDARG	There is no string type attribute corresponding to cfgID.

2.4.8 IHDLinkConfiguration interface

The **IHDLinkConfiguration** interface sets and accesses the configuration of a HDLink device.

An **IHDLinkConfiguration** object interface can be obtained from the IHDLink interface using QueryInterface.

Related Interfaces

Interface	Interface ID	Description
IHDLink	IID_IHDLink	HDlink device interface

Public Member Functions

Method	Description
GetFlag	Gets a boolean flag associated with specified **BMDHDLinkConfigurationID**
SetFlag	Sets a boolean associated with specified **BMDHDLinkConfigurationID**
GetInt	Gets an int64_t associated with specified **BMDHDLinkConfigurationID**
SetInt	Set a int64_t associated with specified **BMDHDLinkConfigurationID**
GetFloat	Gets a float associated with specified **BMDHDLinkConfigurationID**
SetFloat	Sets a float associated with specified **BMDHDLinkConfigurationID**
GetString	Gets a string associated with specified **BMDHDLinkConfigurationID**
SetString	Sets a string associated with specified **BMDHDLinkConfigurationID**

2.4.8.1 IHDLinkConfiguration::GetFlag method

The **GetFlag** method gets a boolean flag associated with a given **BMDHDLinkConfigurationID**.

Syntax

```
HRESULT      GetFlag (BMDHDLinkConfigurationID cfgID, boolean *value);
```

Parameters

Name	Direction	Description
`cfgID`	in	The configuration ID.
`value`	out	The value corresponding to cfgID.

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success
E_INVALIDARG	There is no flag type attribute corresponding to cfgID.

2.4.8.2 IHDLinkConfiguration::SetFlag method

The **SetFlag** method sets a boolean flag associated with a given **BMDHDLinkConfigurationID**.

Syntax

```
HRESULT       SetFlag (BMDHDLinkConfigurationID cfgID, boolean value);
```

Parameters

Name	Direction	Description
`cfgID`	in	The configuration ID.
`value`	in	The value corresponding to cfgID.

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success
E_INVALIDARG	There is no flag type attribute corresponding to cfgID.

2.4.8.3 IHDLinkConfiguration::GetInt method

The **GetInt** method gets an int64_t value associated with a given **BMDHDLinkConfigurationID**.

Syntax

```
HRESULT       GetInt (BMDHDLinkConfigurationID cfgID, int64_t *value);
```

Parameters

Name	Direction	Description
`cfgID`	in	The configuration ID.
`value`	out	The value corresponding to cfgID.

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success
E_INVALIDARG	There is no int type attribute corresponding to cfgID.

2.4.8.4 IHDLinkConfiguration::SetInt method

The **SetInt** method sets a int64_t value associated with a given **BMDHDLinkConfigurationID**.

Syntax

```
HRESULT       SetInt (BMDHDLinkConfigurationID cfgID, int64_t value);
```

Parameters

Name	Direction	Description
`cfgID`	in	The configuration ID.
`value`	in	The value corresponding to cfgID.

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success
E_INVALIDARG	There is no int type attribute corresponding to cfgID.

2.4.8.5 IHDLinkConfiguration::GetFloat method

The **GetFloat** method gets a float value associated with a given **BMDHDLinkConfigurationID**.

Syntax

```
HRESULT      GetFloat (BMDHDLinkConfigurationID cfgID, double *value);
```

Parameters

Name	Direction	Description
`cfgID`	in	The configuration ID.
`value`	out	The value corresponding to cfgID.

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success
E_INVALIDARG	There is no float type attribute corresponding to cfgID.

2.4.8.6 IHDLinkConfiguration::SetFloat method

The **SetFloat** method sets a float associated with a given **BMDHDLinkConfigurationID**.

Syntax

```
HRESULT       SetFloat (BMDHDLinkConfigurationID cfgID, double value);
```

Parameters

Name	Direction	Description
`cfgID`	in	The configuration ID.
`value`	in	The value corresponding to cfgID.

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success
E_INVALIDARG	There is no float type attribute corresponding to cfgID.

2.4.8.7 IHDLinkConfiguration::GetString method

The **GetString** method gets a string value associated with a given **BMDHDLinkConfigurationID**.

Syntax

```
HRESULT       GetString (BMDHDLinkConfigurationID cfgID, string *value);
```

Parameters

Name	Direction	Description
`cfgID`	in	The configuration ID.
`value`	out	The value corresponding to cfgID.

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success
E_INVALIDARG	There is no string type attribute corresponding to cfgID.

2.4.8.8 IHDLinkConfiguration::SetString method

The **SetString** method sets a string value associated with a given **BMDHDLinkConfigurationID**.

Syntax

```
HRESULT      SetString (BMDHDLinkConfigurationID cfgID, string value);
```

Parameters

Name	Direction	Description
`cfgID`	in	The configuration ID.
`value`	in	The value corresponding to cfgID.

Return Values

Value	Description
E_FAIL	Failure
S_OK	Success
E_INVALIDARG	There is no string type attribute corresponding to cfgID.

2.4.8.9 HDLink Attribute ID

BMDHDLinkAttributeID enumerates a set of attributes of a HDLink device which may be queried (see **IHDLinkAttributes** Interface for details).

Name	Type	Description
BMDHDLinkAttribute_Supports3dLUT	Flag	True if 3D Look Up Table is supported.

2.4.8.10 HDLink Configuration ID

BMDHDLinkConfigurationID enumerates a set of attributes of a HDLink device which may be queried (see **IHDLinkConfiguration** Interface for details).

Name	Type	Description
BMDHDLinkConfiguration_ScaleIllegalYUV	Flag	By default, HDLink will scale legal SDI values [64...940] into the display range. If this flag is set, the full range of input values will be scaled into the display range. This will cause normal video to be displayed incorrectly, but it can be useful to view super-black and super-white values.

2.5 Common Data Types

2.5.1 Basic Types

`boolean`

boolean is represented differently on each platform by using its system type:

Windows	BOOL
Mac OS X	OS X bool

`Strings`

Strings are represented differently on each platform, using the most appropriate system type:

Windows	BSTR
Mac OS X	CFStringRef

www.blackmagic-design.com