Baumer

Programmer’s Guide SDK Reference

Baumer GAPI SDK v2.10.0

Version: v2.10.0
Date: Jan, 14 2020

Table of Contents

1 Introduction

The GenICam™ compliant Baumer GAPI (Generic Application Programming Interface) SDK is designed to easily integrate Baumer cameras in your specific software application. The idea behind the Baumer GAPI is to relive the programmer from defining and instantiating all required objects and to transfer these tasks to the Baumer GAPI. The API consists of five main classes (System, Interface, Device, DataStream, Buffer) and uses the GenTL programming interface. Part of the SDK package is the Camera Explorer test tool for quick and easy camera evaluation - just one mouse click for the first image! Camera selection, access to all camera features, image view and recording can be easily performed.

Baumer Industrial Cameras

2 Module Index

2.1 Modules

Here is a list of all modules:

3 Namespace Index

3.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

4 Hierarchical Index

4.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

5 Class Index

5.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

6 File Index

6.1 File List

Here is a list of all documented files with brief descriptions:

7 Module Documentation

7.1 Main Classes

This group defined the main classes of BGAPI2. These classes represent the fundamental logical and physical components of the image processing system.

Classes

- class BGAPI2::Buffer

  This class realizes the data access to the memory. It contains information about the received data (e.g. image size, pixel format). This class belongs to the BGAPI2 main classes.
- class BGAPI2::DataStream

  This class represents a physical data stream from the device and it is responsible for the buffer handling. This class belongs to the BGAPI2 main classes.
- class BGAPI2::Device

  The class Device is used to retrieve information (e.g. model, manufacturer, access modes) of the device (camera) and also to control the device. This class belongs to the BGAPI2 main classes.
- class BGAPI2::Interface

  The class Interface represents a physical interface, e.g. GEV or a logical interface, such as USB and belongs to the BGAPI2 main classes.
- class BGAPI2::System

  The class System is the abstraction of a Producer and belongs to the BGAPI2 main classes.

7.1.1 Detailed Description

This group defined the main classes of BGAPI2. These classes represent the fundamental logical and physical components of the image processing system.

7.2 List Classes

This classes aim to the discovery and listing of the main objects.

Classes

- class BGAPI2::SystemList

  This class is used to search and list system objects and may be instantiated only once.
- class BGAPI2::InterfaceList

  This class is used to search and list interface objects.
- class BGAPI2::DeviceList

  This class is used to discover and list device objects.
- class BGAPI2::DataStreamList

  This class is used to discover and list data stream objects.
- class BGAPI2::BufferList

  This class is used for discovery and listing of buffer objects.

7.2.1 Detailed Description

This classes aim to the discovery and listing of the main objects.

7.3 Interface Classes

This group defined fundamental functions which are used by the main classes.

Classes

- class BGAPI2::Node

  The class Node represent one feature from the provided node list based on the underlying XML definition.
- class BGAPI2::NodeMap

  The class NodeMap represents a collection of Node objects based on the underlying XML definition file. This class supports two representation forms, a unstructured list and a tree structure.
- class BGAPI2::INode

  The class INode act as base for of the main classes and provided the access to the node objects (features).
- class BGAPI2::Events::EventControl

  The class EventControl provided access to custom events as well as the event mode.
- class BGAPI2::Events::PnPEvent

  The class PnPEvent represented a plug'n play event and provides access to the event information.
- class BGAPI2::Events::InterfaceEventControl

  The class InterfaceEventControl provides access to interface specific events, e.g. plug'n play event.
- class BGAPI2::Events::DeviceEvent

  This class represents an device event which was received from the host. Use this class to get event information.
- class BGAPI2::Events::DeviceEventControl

  The class DeviceEventControl provides access to standard events transmitted from the device.
- class BGAPI2::Events::DataStreamEventControl

  The class DataStreamEventControl provides the new buffer event which is used for fetching images.

7.3.1 Detailed Description

This group defined fundamental functions which are used by the main classes.

7.4 Additional Classes

This group provided additional functions to extend the application spectrum of BGAPI2.

Classes

- class BGAPI2::Trace

  The class Trace offers the possibility to monitor the program flow and detect errors. This class belongs to the additional classes.
- class BGAPI2::Image

  The class Image provides the ability of image transformation. This class belongs to the additional classes.
- class BGAPI2::ImageProcessor

  The task of the class ImageProcessor are the creation of image objects and the transformation of pixel formats.

7.4.1 Detailed Description

This group provided additional functions to extend the application spectrum of BGAPI2.

7.5 Exception Classes

This group provided the BGAPI2 exeption handling.

Classes

- class BGAPI2::Exceptions::IException

  This class is responable for the exception handling and represents the parent class of all exception classes.
- class BGAPI2::Exceptions::ErrorException

  General purpose exception.
- class BGAPI2::Exceptions::NotInitializedException

  The requested object is not initialized/opened.
- class BGAPI2::Exceptions::NotImplementedException

  The requested function/feature is not implemented.
- class BGAPI2::Exceptions::ResourceInUseException

  The requested object is already used.
- class BGAPI2::Exceptions::AccessDeniedException

  The requested operation is not allowed/possible, e.g. lose the connection to the device.
- class BGAPI2::Exceptions::InvalidHandleException

  (Given handle does not support the operation.)
- class BGAPI2::Exceptions::NoDataException

  An event contains no event data.
- class BGAPI2::Exceptions::InvalidParameterException

  One of the parameter given was not valid or out of range.
- class BGAPI2::Exceptions::AbortException

  An operation has been aborted before it could be completed.
- class BGAPI2::Exceptions::InvalidBufferException

  Invalid buffer is used. The used Buffer object is not valid.
- class BGAPI2::Exceptions::NotAvailableException

  The requested resource or information is not available at a given time in a current state.
- class BGAPI2::Exceptions::ObjectInvalidException

  The referenced object is not a valid object of BGAPI2.
- class BGAPI2::Exceptions::LowLevelException

  Exception thrown by deeper software layers like GenTL producer.

7.5.1 Detailed Description

This group provided the BGAPI2 exeption handling.

8 Namespace Documentation

8.1 BGAPI2 Namespace Reference

The global namespace of Baumer GAPI SDK 2.

Namespaces

- Events

  The namespace Events consists of classes which belongs to the event interface.
- Exceptions

  The namespace Exceptions consists of classes which are responsible for exception handling.

Classes

- class _pairb
- class _paird
- class _pairds
- class _pairi
- class _pairn
- class _pairnm
- class _pairs
- struct bo_tHistRecords
- struct bo_tRGB16QUAD
- class Buffer

  This class realizes the data access to the memory. It contains information about the received data (e.g. image size, pixel format). This class belongs to the BGAPI2 main classes.
- class BufferList

  This class is used for discovery and listing of buffer objects.
- class DataStream

  This class represents a physical data stream from the device and it is responsible for the buffer handling. This class belongs to the BGAPI2 main classes.
- class DataStreamList

  This class is used to discover and list data stream objects.
- class Device

  The class Device is used to retrieve information (e.g. model, manufacturer, access modes) of the device (camera) and also to control the device. This class belongs to the BGAPI2 main classes.
- class DeviceList

  This class is used to discover and list device objects.
- class Image

The class Image provides the ability of image transformation. This class belongs to the additional classes.

- class ImageProcessor

  The task of the class ImageProcessor are the creation of image objects and the transformation of pixel formats.
- class INode

  The class INode act as base for of the main classes and provided the access to the node objects (features).
- class Interface

  The class Interface represents a physical interface, e.g. GEV or a logical interface, such as USB and belongs to the BGAPI2 main classes.
- class InterfaceList

  This class is used to search and list interface objects.
- class Node

  The class Node represent one feature from the provided node list based on the underlying XML definition.
- class NodeMap

  The class NodeMap represents a collection of Node objects based on the underlying XML definition file. This class supports two representation forms, a unstructured list and a tree structure.
- class Polarizer

  Provides functionality to calculate several different formats out of the raw polarized camera data.
- class String
- class System

  The class System is the abstraction of a Producer and belongs to the BGAPI2 main classes.
- class SystemList

  This class is used to search and list system objects and may be instantiated only once.
- class Trace

  The class Trace offers the possibility to monitor the program flow and detect errors. This class belongs to the additional classes.

Typedefs

- typedef struct BGAPI2::_sSystemListData **tSystemListData**
- typedef struct BGAPI2::_sSystemListData ∗ **ptSystemListData**
- typedef struct BGAPI2::_sInterfaceListData **tInterfaceListData**
- typedef struct BGAPI2::_sInterfaceListData ∗ **ptInterfaceListData**
- typedef struct BGAPI2::_sSystemData **tSystemData**
- typedef struct BGAPI2::_sSystemData ∗ **ptSystemData**
- typedef struct BGAPI2::_sDeviceListData **tDeviceListData**
- typedef struct BGAPI2::_sDeviceListData ∗ **ptDeviceListData**
- typedef struct BGAPI2::_sDataStreamListData **tDataStreamListData**
- typedef struct BGAPI2::_sDataStreamListData ∗ **ptDataStreamListData**
- typedef struct BGAPI2::_sBufferListData **tBufferListData**
- typedef struct BGAPI2::_sBufferListData ∗ **ptBufferListData**
- typedef struct BGAPI2::_sBufferData **tBufferData**
- typedef struct BGAPI2::_sBufferData ∗ **ptBufferData**
- typedef struct BGAPI2::_sDataStreamData **tDataStreamData**
- typedef struct BGAPI2::_sDataStreamData ∗ **ptDataStreamData**
- typedef struct BGAPI2::_sDeviceData **tDeviceData**
- typedef struct BGAPI2::_sDeviceData ∗ **ptDeviceData**
- typedef struct BGAPI2::_sInterfaceData **tInterfaceData**
- typedef struct BGAPI2::_sInterfaceData ∗ **ptInterfaceData**
- typedef struct BGAPI2::_sImageData **tImageData**
- typedef struct BGAPI2::_sImageData ∗ **ptImageData**
- typedef struct BGAPI2::_sImageProcessorData **tImageProcessorData**
- typedef struct BGAPI2::_sImageProcessorData ∗ **ptImageProcessorData**
- typedef struct BGAPI2::_sBrightnessAutoData **tBrightnessAutoData**
- typedef struct BGAPI2::_sBrightnessAutoData ∗ **ptBrightnessAutoData**

Functions

- std::ostream & **operator**<< (std::ostream &out, BGAPI2::String const &ExStr) BGAPI2_DECL

8.1.1 Detailed Description

The global namespace of Baumer GAPI SDK 2.

8.2 BGAPI2::Events Namespace Reference

The namespace Events consists of classes which belongs to the event interface.

Classes

- class DataStreamEventControl

  The class DataStreamEventControl provides the new buffer event which is used for fetching images.
- class DeviceEvent

  This class represents an device event which was received from the host. Use this class to get event information.
- class DeviceEventControl

  The class DeviceEventControl provides access to standard events transmitted from the device.
- class EventControl

  The class EventControl provided access to custom events as well as the event mode.
- class InterfaceEventControl

  The class InterfaceEventControl provides access to interface specific events, e.g. plug'n play event.
- class PnPEvent

  The class PnPEvent represented a plug'n play event and provides access to the event information.

Typedefs

- typedef void(BGAPI2CALL * PnPEventHandler) (void *callBackOwner, PnPEvent *pBuffer)

  Function pointer for pnp event notification, which points to a user defined handler.
- typedef void(BGAPI2CALL * DeviceEventHandler) (void *callBackOwner, DeviceEvent *pDeviceEvent)

  Function pointer for device event notification, which points to a user defined handler.
- typedef void(BGAPI2CALL * NewBufferEventHandler) (void *callBackOwner, Buffer *pBuffer)

  Function pointer for buffer notification, which points to a user defined handler.

Enumerations

- enum EventMode { EVENTMODE_UNREGISTERED = 0, EVENTMODE_POLLING = 1, EVENTMODE_EVENT_HANDLER = 2 }

  Enumeration, which defines kinds of event modes.
- enum PnPType { PNPTYPE_DEVICEREMOVED = 0, PNPTYPE_DEVICEADDED = 1 }

  Enumeration, which defines kinds of PnP events.

8.2.1 Detailed Description

The namespace Events consists of classes which belongs to the event interface.

8.2.2 Enumeration Type Documentation

8.2.2.1 EventMode

`enum BGAPI2::Events::EventMode`

Enumeration, which defines kinds of event modes.

Enumerator

EVENTMODE_UNREGISTERED	Event handling is disabled. No events can be retrieved.
EVENTMODE_POLLING	This EventMode allows the retrieval of occurred events by a function call of the corresponding get function.
EVENTMODE_EVENT_HANDLER	This EventMode allows the retrieval of occurred events by a previously registered callback function.

Definition at line 1769 of file bgapi2_genicam.hpp.

8.2.2.2 PnPType

`enum BGAPI2::Events::PnPType`

Enumeration, which defines kinds of PnP events.

Enumerator

PNPTYPE_DEVICEREMOVED	Represents a device removed PnP event.
PNPTYPE_DEVICEADDED	Represents a device add PnP event.

Definition at line 1787 of file bgapi2_genicam.hpp.

8.3 BGAPI2::Exceptions Namespace Reference

The namespace Exceptions consists of classes which are responsible for exception handling.

Classes

- class AbortException

  An operation has been aborted before it could be completed.
- class AccessDeniedException

  The requested operation is not allowed/possible, e.g. lose the connection to the device.
- class ErrorException

  General purpose exception.
- class IException

  This class is responable for the exception handling and represents the parent class of all exception classes.
- class InvalidBufferException

  Invalid buffer is used. The used Buffer object is not valid.
- class InvalidHandleException

  (Given handle does not support the operation.)
- class InvalidParameterException

  One of the parameter given was not valid or out of range.
- class LowLevelException

  Exception thrown by deeper software layers like GenTL producer.
- class NoDataException

  An event contains no event data.
- class NotAvailableException

  The requested resource or information is not available at a given time in a current state.
- class NotImplementedException

  The requested function/feature is not implemented.
- class NotInitializedException

  The requested object is not initialized/opened.
- class ObjectInvalidException

  The referenced object is not a valid object of BGAPI2.
- class ResourceInUseException

  The requested object is already used.

8.3.1 Detailed Description

The namespace Exceptions consists of classes which are responsible for exception handling.

9 Class Documentation

9.1 BGAPI2::_pairb Class Reference

Public Attributes

- String **first**
- Buffer * **second**

9.1.1 Detailed Description

Definition at line 119 of file bgapi2_def.h.

The documentation for this class was generated from the following file:

- bgapi2_def.h

9.2 BGAPI2::_paird Class Reference

Public Attributes

- String **first**
- Device * **second**

9.2.1 Detailed Description

Definition at line 105 of file bgapi2_def.h.

The documentation for this class was generated from the following file:

- bgapi2_def.h

9.3 BGAPI2::_pairds Class Reference

Public Attributes

- String **first**
- DataStream ∗ **second**

9.3.1 Detailed Description

Definition at line 112 of file bgapi2_def.h.

The documentation for this class was generated from the following file:

- bgapi2_def.h

9.4 BGAPI2::_pairi Class Reference

Public Attributes

- String **first**
- Interface ∗ **second**

9.4.1 Detailed Description

Definition at line 98 of file bgapi2_def.h.

The documentation for this class was generated from the following file:

- bgapi2_def.h

9.5 BGAPI2::_pairn Class Reference

Public Attributes

- bo_int64 **first**
- Node ∗ **second**

9.5.1 Detailed Description

Definition at line 130 of file bgapi2_def.h.

The documentation for this class was generated from the following file:

- bgapi2_def.h

9.6 BGAPI2::_pairnm Class Reference

Public Attributes

- String **first**
- _pairn **second**

9.6.1 Detailed Description

Definition at line 136 of file bgapi2_def.h.

The documentation for this class was generated from the following file:

- bgapi2_def.h

9.7 BGAPI2::_pairs Class Reference

Public Attributes

- String **first**
- System ∗ **second**

9.7.1 Detailed Description

Definition at line 91 of file bgapi2_def.h.

The documentation for this class was generated from the following file:

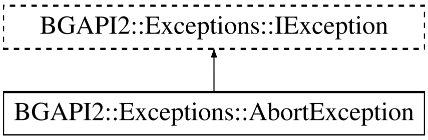
- bgapi2_def.h

9.8 BGAPI2::Exceptions::AbortException Class Reference

An operation has been aborted before it could be completed.

```
#include <bgapi2_genicam.hpp>
```

Inheritance diagram for BGAPI2::Exceptions::AbortException:


BGAPI2::Exceptions::IException
BGAPI2::Exceptions::AbortException

Additional Inherited Members

9.8.1 Detailed Description

An operation has been aborted before it could be completed.

See also

BGAPI2::Events::DataStreamEventControl::CancelGetFilledBuffer and BGAPI2::Events::DataStreamEventControl::GetFilledBuffer

Definition at line 3743 of file bgapi2_genicam.hpp.

The documentation for this class was generated from the following file:

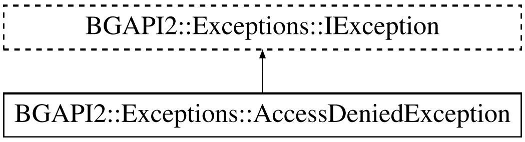
- bgapi2_genicam.hpp

9.9 BGAPI2::Exceptions::AccessDeniedException Class Reference

The requested operation is not allowed/possible, e.g. lose the connection to the device.

`#include <bgapi2_genicam.hpp>`

Inheritance diagram for BGAPI2::Exceptions::AccessDeniedException:


BGAPI2::Exceptions::IException
BGAPI2::Exceptions::AccessDeniedException

Additional Inherited Members

9.9.1 Detailed Description

The requested operation is not allowed/possible, e.g. lose the connection to the device.

Definition at line 3706 of file bgapi2_genicam.hpp.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.10 BGAPI2::bo_tHistRecords Struct Reference

Public Attributes

- bo_ushort * **pcRed**
- bo_ushort * **pcGreen**
- bo_ushort * **pcBlue**
- bo_ushort * **pcLuma**
- int **length**
- int * **pSizeFilled**
- bool **ThresEnable**
- int **ThresMin**
- int **ThresMax**
- int * **red_under**
- int * **red_over**
- int * **green_under**
- int * **green_over**
- int * **blue_under**
- int * **blue_over**

9.10.1 Detailed Description

Definition at line 149 of file bgapi2_def.h.

The documentation for this struct was generated from the following file:

- bgapi2_def.h

9.11 BGAPI2::bo_tRGB16QUAD Struct Reference

Public Attributes

- bo_ushort **rgbBlue**
- bo_ushort **rgbGreen**
- bo_ushort **rgbRed**
- bo_ushort **rgbReserved**

9.11.1 Detailed Description

Definition at line 172 of file bgapi2_def.h.

The documentation for this struct was generated from the following file:

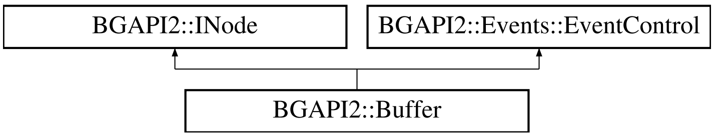
- bgapi2_def.h

9.12 BGAPI2::Buffer Class Reference

This class realizes the data access to the memory. It contains information about the received data (e.g. image size, pixel format). This class belongs to the BGAPI2 main classes.

```
#include <bgapi2_genicam.hpp>
```

Inheritance diagram for BGAPI2::Buffer:



Public Member Functions

- Buffer ()

  Parameterless constructor for creating of a Buffer-Object. When using this constructor, the buffer object takes care of memory management. The necessary memory for this Buffer-Object will only be allocated when it is added to a Buffer list.
- Buffer (void *pUserObj)

  Constructor for creating of a Buffer-Object. When using this constructor, the buffer object takes care of memory management. The necessary memory for this Buffer-Object will only be allocated when added to a Buffer list.
- Buffer (void *pUserBuffer, bo_uint64 uUserBufferSize, void *pUserObj)

  Constructor for creating of a Buffer-Object. When using this constructor, the user takes care of allocation of the necessary memory.
- ∼Buffer ()

  Destructor to destroy an Buffer object.
- String GetID ()

  This function delivers the unique string identifier of the Buffer object, which is used in the BufferList.
- void QueueBuffer ()

  This function moves a Buffer object of the BufferList into the input buffer queue and make it available for the image acquisition.
- void * GetMemPtr ()

  This function delivers a pointer to the memory of the Buffer object.
- bo_uint64 GetMemSize ()

  This function delivers the size of the allocated memory.
- void * GetUserObj ()

  *This function delivers a pointer to a user allocated memory. See the construcors Buffer(void * pUserObj) and Buffer(void *pUserBuffer, bo_uint64 uUserBufferSize, void *pUserObj).*
- bo_uint64 GetTimestamp ()

  This function delivers the timestamp of the buffer obtained by the camera. The basic requirement for this function is that the Buffer object needs valid image data. See also the exception description of Exceptions::LowLevelException.
- bo_uint64 GetHostTimestamp ()

  This function returns the host time stamp of the first received data packet of a new image obtained by a steady clock. The basic requirement for this function is that the Buffer object needs valid image data. See also the exception description of Exceptions::LowLevelException.
- bo_bool GetNewData ()

  This function delivers a flag that indicates whether the object buffer contains new data.

- bo_bool GetIsQueued ()

  This function delivers a flag that indicates whether the Buffer object has been queued.
- bo_bool GetIsAcquiring ()

  This function delivers a flag that indicates whether the Buffer object is being filled.
- bo_bool GetIsIncomplete ()

  This function delivers a flag that indicates whether the expected data has been completely transferred.
- String GetTLType ()

  This function delivers the transport layer type with which the data were transmitted.
- bo_uint64 GetSizeFilled ()

  This function delivers the number of transferred bytes.
- bo_uint64 GetWidth ()

  This function delivers the width of the image in pixel.
- bo_uint64 GetHeight ()

  This function delivers the height of the image in pixel.
- bo_uint64 GetXOffset ()

  This function delivers the offset in X direction in pixel.
- bo_uint64 GetYOffset ()

  This function delivers the offset in Y direction in pixel.
- bo_uint64 GetXPadding ()

  This function delivers the number of extra bytes transmitted at the end of each line and it is only specified for the payload types Image and Extended Chunk Data. See function Buffer::GetPayloadType.
- bo_uint64 GetYPadding ()

  This function delivers the number of extra bytes transmitted at the end of the image and it is only specified for the payload types Image and Extended Chunk Data. See function Buffer::GetPayloadType.
- bo_uint64 GetFrameID ()

  This function delivers a sequentially incremented number of the image. The number refers e.g. for GigE Vision and for USB3 Vision to the block ID of the stream.
- bo_bool GetImagePresent ()

  This function delivers true, if the Buffer object includes image data.
- bo_uint64 GetImageOffset ()

  This function returns the offset into the memory of the Buffer object to the begin of the image data.
- String GetPayloadType ()

  This function delivers the payload type of the Buffer object. See the payload type definitions in bgapi2↩_def.h (BGAPI2_PAYLOADTYPE_xxx).
- String GetPixelFormat ()

  This function delivers the pixel format of the Buffer object.
- bo_uint64 GetDeliveredImageHeight ()

  This function delivers the transmitted lines of an image. Only valid for the payload types 'Image' and 'ImageExt'.
- bo_uint64 GetDeliveredChunkPayloadSize ()

  This function delivers the number of transmitted bytes. Only valid for the chunk payload types 'Chunk↩Data' and 'ImageExt'.
- bo_bool GetContainsChunk ()

  This function delivers true, if the Buffer object includes chunk data.
- bo_uint64 GetChunkLayoutID ()

  This function delivers a value representing the current structure of the chunk data. This value changes when changing the structure of the chunk data.
- NodeMap * GetChunkNodeList ()

  This function delivers a NodeMap of available chunk information of the Buffer object. Only valid for chunk payload types 'ChunkData' and 'ImageExt'.
- String GetFileName ()

  This function delivers the file name of the Buffer object. Only valid for payload types 'File'.

- DataStream ∗ GetParent ()

  This function delivers the superordinate DataStream object but only if the Buffer object was added to a BufferList.
- void ∗ GetReserved ()

  Undocumented function.

Friends

- class **BufferList**
- class **DataStream**

9.12.1 Detailed Description

This class realizes the data access to the memory. It contains information about the received data (e.g. image size, pixel format). This class belongs to the BGAPI2 main classes.

Definition at line 2153 of file bgapi2_genicam.hpp.

9.12.2 Constructor & Destructor Documentation

9.12.2.1 Buffer() [1/2]

```
BGAPI2::Buffer::Buffer (
            void ∗ pUserObj )
```

Constructor for creating of a Buffer-Object. When using this constructor, the buffer object takes care of memory management. The necessary memory for this Buffer-Object will only be allocated when added to a Buffer list.

Parameters

pUserObj	A pointer to a user allocated memory. This pointer is stored in the Buffer-Object and can be queried at any time. See function Buffer::GetUserObj.

9.12.2.2 Buffer() [2/2]

```
BGAPI2::Buffer::Buffer (
            void ∗ pUserBuffer,
            bo_uint64 uUserBufferSize,
            void ∗ pUserObj )
```

Constructor for creating of a Buffer-Object. When using this constructor, the user takes care of allocation of the necessary memory.

To use the actual necessary memory size the functions Device::GetPayloadSize and DataStream::↵ GetPayloadSize are used respectively. To use the maximum required memory size of a device the maximum of the 'PayloadSize' feature is queried. See Device::GetRemoteNode and Node::GetIntMax.

Parameters

pUserBuffer	A pointer to a user allocated data buffer.
uUserBufferSize	The size of the user allocated data buffer.
pUserObj	A pointer to a user allocated memory. This pointer is stored in the Buffer-Object and can be queried at any time. See function Buffer::GetUserObj.

9.12.3 Member Function Documentation

9.12.3.1 GetChunkLayoutID()

```
BGAPI2::Buffer::GetChunkLayoutID ( )
```

This function delivers a value representing the current structure of the chunk data. This value changes when changing the structure of the chunk data.

The change in this value initiates a new internal parsing of the chunk data. Only valid for chunk payload types 'ChunkData' and 'ImageExt'.

Returns

bo_uint64 ID of the chunk data layout delivered in the buffer

Exceptions

Exceptions::NotAvailableException	The Buffer object is not added to a BufferList.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Possible reasons for this exception are a incomplete transferred image and a wrong payload type. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.2 GetChunkNodeList()

```
BGAPI2::Buffer::GetChunkNodeList ( )
```

This function delivers a NodeMap of available chunk information of the Buffer object. Only valid for chunk payload types 'ChunkData' and 'ImageExt'.

Returns

NodeMap∗ The list of all available chunk information of the Buffer object.

Exceptions

Exceptions::NotAvailableException	The Buffer object is not added to a BufferList. Wrong payload type used.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.12.3.3 GetContainsChunk()

```
BGAPI2::Buffer::GetContainsChunk ( )
```

This function delivers true, if the Buffer object includes chunk data.

Exceptions

Exceptions::NotAvailableException	The Buffer object is not added to a BufferList.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Possible reasons for this exception are a incomplete transferred image and a wrong payload type. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.4 GetDeliveredChunkPayloadSize()

```
BGAPI2::Buffer::GetDeliveredChunkPayloadSize ( )
```

This function delivers the number of transmitted bytes. Only valid for the chunk payload types 'ChunkData' and 'ImageExt'.

Returns

bo_uint64 The number of transmitted bytes.

Exceptions

Exceptions::NotAvailableException	The Buffer object is not added to a BufferList.
Exceptions::ObjectInvalidException	The calling object is not valid.

Exceptions

Exceptions::LowLevelException	GenTL Producer error. Possible reasons for this exception are a incomplete transferred image and a wrong payload type. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.5 GetDeliveredImageHeight()

```
BGAPI2::Buffer::GetDeliveredImageHeight ( )
```

This function delivers the transmitted lines of an image. Only valid for the payload types 'Image' and 'ImageExt'.

Returns

bo_uint64 The transmitted lines of an image.

Exceptions

Exceptions::NotAvailableException	The Buffer object is not added to a BufferList.
Exceptions::LowLevelException	GenTL Producer error. Possible reasons for this exception are a incomplete transferred image and a wrong payload type. Use IException::GetErrorDescription or check trace output for more detailed error information.

Remarks

For matrix cameras this usually is the image height. This function finds application for variable size linescan images.

9.12.3.6 GetFileName()

```
BGAPI2::Buffer::GetFileName ( )
```

This function delivers the file name of the Buffer object. Only valid for payload types 'File'.

Returns

String The file name of the Buffer object.

Exceptions

Exceptions::NotAvailableException	The Buffer object is not added to a BufferList.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	Baumer GenTL producer does not implement this feature. GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.7 GetFrameID()

```
BGAPI2::Buffer::GetFrameID ( )
```

This function delivers a sequentially incremented number of the image. The number refers e.g. for GigE Vision and for USB3 Vision to the block ID of the stream.

Returns

bo_uint64 The sequentially incremented number of the image.

Exceptions

Exceptions::NotAvailableException	The Buffer object is not added to a BufferList.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.8 GetHeight()

```
BGAPI2::Buffer::GetHeight ( )
```

This function delivers the height of the image in pixel.

Returns

bo_uint64 The height of the image in pixel.

Exceptions

Exceptions::NotAvailableException	Possible reasons for this exceptions are the Buffer object is not added to a BufferList, or when using the payload type chunk the chunk feature 'ChunkHeight' is not available.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.9 GetHostTimestamp()

```
BGAPI2::Buffer::GetHostTimestamp ( )
```

This function returns the host time stamp of the first received data packet of a new image obtained by a steady clock. The basic requirement for this function is that the Buffer object needs valid image data. See also the exception description of Exceptions::LowLevelException.

Returns

bo_uint64 The timestamp of the image in ns.

Exceptions

Exceptions::NotAvailableException	The Buffer object is not added to a BufferList.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.12.3.10 GetID()

```
BGAPI2::Buffer::GetID ( )
```

This function delivers the unique string identifier of the Buffer object, which is used in the BufferList.

Returns

String The unique string identifier.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.12.3.11 GetImageOffset()

```
BGAPI2::Buffer::GetImageOffset ( )
```

This function returns the offset into the memory of the Buffer object to the begin of the image data.

Exceptions

Exceptions::NotAvailableException	The Buffer object doesn't include image data. The possible reason for this exception is, the image chunk on the device is not enabled.

9.12.3.12 GetImagePresent()

```
BGAPI2::Buffer::GetImagePresent ( )
```

This function delivers true, if the Buffer object includes image data.

Exceptions

Exceptions::NotAvailableException	This function is currently not suported.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.13 GetIsAcquiring()

```
BGAPI2::Buffer::GetIsAcquiring ( )
```

This function delivers a flag that indicates whether the Buffer object is being filled.

The flag is set when data is written in the memory of the Buffer object. The flag is reset when the Buffer object is filled.

Returns

bo_bool The flag that indicates whether the Buffer object is being filled.

Exceptions

Exceptions::NotAvailableException	The Buffer object is not added to a BufferList.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. A possible reason for this exception is the use of this function with the Baumer Filter Driver. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.14 GetIsIncomplete()

```
BGAPI2::Buffer::GetIsIncomplete ( )
```

This function delivers a flag that indicates whether the expected data has been completely transferred.

The flag is set when the expected data was not completely transferred. The flag is reset when all expected data was transferred.

Returns

bo_bool The flag that indicates whether the expected data has been completely transferred.

Exceptions

Exceptions::NotAvailableException	The Buffer object is not added to a BufferList.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.15 GetIsQueued()

```
BGAPI2::Buffer::GetIsQueued ( )
```

This function delivers a flag that indicates whether the Buffer object has been queued.

The flag is set when the Buffer object is moved into the input buffer queue. See functions Buffer::QueueBuffer, BufferList::FlushAllToInputQueue and BufferList::FlushUnqueuedToInputQueue. The flag is reset when the Buffer object is fetched by DataStreamEventControl::GetFilledBuffer function.

Returns

bo_bool The flag that indicates whether the Buffer object has been queued.

Exceptions

Exceptions::NotAvailableException	The Buffer object is not added to a BufferList.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.16 GetMemPtr()

```
BGAPI2::Buffer::GetMemPtr ( )
```

This function delivers a pointer to the memory of the Buffer object.

Returns

void∗ The pointer to the memory of Buffer object.

Exceptions

Exceptions::NotAvailableException	The Buffer object is currently no memory allocated, because it was not added to a BufferList. This exception is thrown only when using the constructors Buffer() and Buffer(void ∗ pUserObj).
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information. This exception is thrown only when using the constructors Buffer() and Buffer(void ∗ pUserObj).

9.12.3.17 GetMemSize()

```
BGAPI2::Buffer::GetMemSize ( )
```

This function delivers the size of the allocated memory.

Returns

bo_uint64 The size of the allocated memory.

Exceptions

Exceptions::NotAvailableException	The Buffer object is currently no memory allocated, because it was not added to a BufferList. This exception is thrown only when using the constructors Buffer() and Buffer(void ∗ pUserObj).
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information. This exception is thrown only when using the constructors Buffer() and Buffer(void ∗ pUserObj).

9.12.3.18 GetNewData()

```
BGAPI2::Buffer::GetNewData ( )
```

This function delivers a flag that indicates whether the object buffer contains new data.

The flag is set when the Buffer object is moved into the output buffer queue. The flag is reset when the Buffer object moved into the input buffer queue. See functions Buffer::QueueBuffer, BufferList::FlushAllToInputQueue and BufferList::FlushUnqueuedToInputQueue.

Returns

bo_bool The flag that indicates whether the object buffer contains new data.

Exceptions

Exceptions::NotAvailableException	The Buffer object is not added to a BufferList.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.19 GetParent()

```
BGAPI2::Buffer::GetParent ( )
```

This function delivers the superordinate DataStream object but only if the Buffer object was added to a BufferList.

Returns

DataStream∗ A pointer to the superordinate DataStream object or NULL if the Buffer object is not added to a BufferList.

9.12.3.20 GetPayloadType()

```
BGAPI2::Buffer::GetPayloadType ( )
```

This function delivers the payload type of the Buffer object. See the payload type definitions in bgapi2_def.h (BGAPI2_PAYLOADTYPE_xxx).

Returns

String The payload type of the Buffer object.

Exceptions

Exceptions::NotAvailableException	The Buffer object is not added to a BufferList.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. A possible reason for this exception is a incomplete transferred image. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.21 GetPixelFormat()

```
BGAPI2::Buffer::GetPixelFormat ( )
```

This function delivers the pixel format of the Buffer object.

Returns

String The pixelformat of the Buffer object.

Exceptions

Exceptions::NotAvailableException	The Buffer object is not added to a BufferList, or when using the payload type chunk the chunk feature 'ChunkPixelFormat' is not available.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. A possible reason for this exception is a incomplete transferred image. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.22 GetSizeFilled()

```
BGAPI2::Buffer::GetSizeFilled ( )
```

This function delivers the number of transferred bytes.

Returns

bo_uint64 The number of transferred bytes.

Exceptions

Exceptions::NotAvailableException	The Buffer object is not added to a BufferList.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.23 GetTimestamp()

```
BGAPI2::Buffer::GetTimestamp ( )
```

This function delivers the timestamp of the buffer obtained by the camera. The basic requirement for this function is that the Buffer object needs valid image data. See also the exception description of Exceptions::LowLevelException.

Returns

bo_uint64 The timestamp of the image.

Exceptions

Exceptions::NotAvailableException	The Buffer object is not added to a BufferList.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Possible reasons of this exception are: the Buffer object is still empty, the Buffer object is moved into the input buffer queue with Buffer::QueueBuffer, the Buffer object is filled, but is not fetched with DataStreamEventControl::GetFilledBuffer, the payload type of the received data is not supported by the GenTL producer. See function Buffer::GetPayloadType. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.24 GetTLType()

```
BGAPI2::Buffer::GetTLType ( )
```

This function delivers the transport layer type with which the data were transmitted.

Returns

String The transport layer type with which the data were transmitted.

Exceptions

Exceptions::NotAvailableException	The Buffer object is not added to a BufferList.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.25 GetUserObj()

```
BGAPI2::Buffer::GetUserObj ( )
```

This function delivers a pointer to a user allocated memory. See the construcors Buffer(void ∗ pUser↩ Obj) and Buffer(void ∗pUserBuffer, bo_uint64 uUserBufferSize, void ∗pUserObj).

Returns

void∗ The pointer to a user allocated memory.

Exceptions

Exceptions::NotAvailableException	The Buffer object was created without user pointer.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.12.3.26 GetWidth()

```
BGAPI2::Buffer::GetWidth ( )
```

This function delivers the width of the image in pixel.

Returns

bo_uint64 The width of the image in pixel.

Exceptions

Exceptions::NotAvailableException	Possible reasons for this exceptions are the Buffer object is not added to a BufferList, or when using the payload type chunk the chunk feature 'ChunkWidth' is not available.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.27 GetXOffset()

```
BGAPI2::Buffer::GetXOffset ( )
```

This function delivers the offset in X direction in pixel.

Returns

bo_uint64 The offset in X direction in pixel.

Exceptions

Exceptions::NotAvailableException	Possible reasons for this exceptions are the Buffer object is not added to a BufferList, or when using the payload type chunk the chunk feature 'ChunkOffsetX' is not available.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.28 GetXPadding()

```
BGAPI2::Buffer::GetXPadding ( )
```

This function delivers the number of extra bytes transmitted at the end of each line and it is only specified for the payload types Image and Extended Chunk Data. See function Buffer::GetPayload↩ Type.

Returns

bo_uint64 The number of extra bytes transmitted at the end of each line.

Exceptions

Exceptions::NotAvailableException	Possible reasons for this exceptions are the Buffer object is not added to a BufferList, when using the wrong the payload type.
Exceptions::NotAvailableException	Possible reasons for this exceptions are the Buffer object is not added to a BufferList, and when using the payload type chunk the chunk feature 'ChunkOffsetX' is not available.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.29 GetYOffset()

```
BGAPI2::Buffer::GetYOffset ( )
```

This function delivers the offset in Y direction in pixel.

Returns

bo_uint64 The offset in Y direction in pixel.

Exceptions

Exceptions::NotAvailableException	Possible reasons for this exceptions are the Buffer object is not added to a BufferList, or when using the payload type chunk the chunk feature 'ChunkOffsetY' is not available.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.30 GetYPadding()

```
BGAPI2::Buffer::GetYPadding ( )
```

This function delivers the number of extra bytes transmitted at the end of the image and it is only specified for the payload types Image and Extended Chunk Data. See function Buffer::GetPayloadType.

Returns

bo_uint64 The number of extra bytes transmitted at the end of the image.

Exceptions

Exceptions::NotAvailableException	Possible reasons for this exceptions are the Buffer object is not added to a BufferList, when using the wrong the payload type.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.12.3.31 QueueBuffer()

```
BGAPI2::Buffer::QueueBuffer ( )
```

This function moves a Buffer object of the BufferList into the input buffer queue and make it available for the image acquisition.

If the image aquisition is done the filled Buffer object is moved into the output buffer queue and is fetched with function DataStreamEventControl::GetFilledBuffer. See also the functions of the BufferList, to move the Buffer object between the internal lists. The Buffer object has to previously added with BufferList::Add to a BufferList, otherwise the function raises an exception.

Exceptions

Exceptions::NotAvailableException	The Buffer object is not added to a BufferList.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.13 BGAPI2::BufferList Class Reference

This class is used for discovery and listing of buffer objects.

`#include <bgapi2_genicam.hpp>`

Classes

- class iterator

  This class provides a iterator that can read or modify any element in the list.

Public Member Functions

- void Add (Buffer ∗pBuffer)

  This function adds a Buffer object to the buffer list. If the Buffer object is created without external memory, this function allocates the necessary memory.
- void ∗ RevokeBuffer (Buffer ∗pBuffer)

  This function removes a Buffer object from the buffer list.
- void clear ()

  This function removes all Buffer objects from the Buffer list.
- bo_uint64 size ()

  This function delivers the number of Buffer objects in the Buffer list.
- Buffer ∗ operator[ ] (const String &bufid)

  This operator allows the direct access to an object of the Buffer list.
- void FlushInputToOutputQueue ()

  This function moves all Buffer objects from the input buffer queue to the output buffer queue. See function Buffer::QueueBuffer and DataStreamEventControl::GetFilledBuffer.
- void FlushAllToInputQueue ()

  This function moves all Buffer objects of the Buffer list to the input buffer queue even those in the output buffer queue. See function Buffer::QueueBuffer and DataStreamEventControl::GetFilledBuffer.
- void FlushUnqueuedToInputQueue ()

  This function moves all free (not queued) Buffer objects of the Buffer list to the input buffer queue. See function Buffer::QueueBuffer.
- void DiscardOutputBuffers ()

  This function discards all Buffer objects in the output buffer queue. The discarded Buffer objects are free.
- void DiscardAllBuffers ()

  This function discards all Buffer objects in the input buffer queue and output buffer queue. The discarded Buffer objects are free.
- bo_uint64 GetDeliveredCount ()

  This function delivers the number of Buffer objects that have been delivered since the start of the DataStream object.
- bo_uint64 GetUnderrunCount ()

  This function delivers the number of lost frames due to queue underrun since the DataStream object was started. This counter is incremented every time the data could not be acquired because there was no Buffer object in the input buffer queue.
- bo_uint64 GetAnnouncedCount ()

  This function delivers the number of Buffer objects in the Buffer list. Same as BufferList::size.
- bo_uint64 GetQueuedCount ()

This function delivers the number of Buffer objects in the input buffer queue. See function Buffer::QueueBuffer.

- bo_uint64 GetAwaitDeliveryCount ()

  This function delivers the number of Buffer objects in the output buffer queue. See function DataStreamEventControl::GetFilledBuffer.
- bo_uint64 GetStartedCount ()

  This function delivers the number of Buffer objects which are currently being filled.
- iterator begin ()

  This function delivers an iterator on the top of the Buffer list.
- iterator end ()

  This function delivers an iterator at the end of the Buffer list.
- iterator find (const String &_keyval)

  This function delivers an iterator on an object to be found. The object is not found, this functions delivers an end-iterator.
- void ∗ GetReserved ()

  Undocumented function.

Friends

- class **DataStream**

9.13.1 Detailed Description

This class is used for discovery and listing of buffer objects.

Definition at line 677 of file bgapi2_genicam.hpp.

9.13.2 Member Function Documentation

9.13.2.1 Add()

```
BGAPI2::BufferList::Add (
            Buffer ∗ pBuffer )
```

This function adds a Buffer object to the buffer list. If the Buffer object is created without external memory, this function allocates the necessary memory.

See the Buffer constructors Buffer::Buffer(), Buffer::Buffer(void ∗ pUserObj) and Buffer::Buffer(void ∗pUserBuffer, bo_uint64 uUserBufferSize, void ∗pUserObj).

Parameters

pBuffer	The Buffer object to be added.

Exceptions

Exceptions::InvalidParameterException	The passed parameter is not a valid Buffer object.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.13.2.2 begin()

```
BGAPI2::BufferList::begin ( )
```

This function delivers an iterator on the top of the Buffer list.

Returns

iterator The iterator on the top of the Buffer list.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.13.2.3 clear()

```
BGAPI2::BufferList::clear ( )
```

This function removes all Buffer objects from the Buffer list.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.13.2.4 DiscardAllBuffers()

```
BGAPI2::BufferList::DiscardAllBuffers ( )
```

This function discards all Buffer objects in the input buffer queue and output buffer queue. The discarded Buffer objects are free.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.13.2.5 DiscardOutputBuffers()

```
BGAPI2::BufferList::DiscardOutputBuffers ( )
```

This function discards all Buffer objects in the output buffer queue. The discarded Buffer objects are free.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.13.2.6 end()

```
BGAPI2::BufferList::end ( )
```

This function delivers an iterator at the end of the Buffer list.

Returns

iterator The iterator at the end of the Buffer list.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.13.2.7 find()

```
BGAPI2::BufferList::find (
            const String & _keyval )
```

This function delivers an iterator on an object to be found. The object is not found, this functions delivers an end-iterator.

Parameters

_keyval	The ID to the object to be found.

Returns

iterator The iterator to the found object.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.13.2.8 FlushAllToInputQueue()

```
BGAPI2::BufferList::FlushAllToInputQueue ( )
```

This function moves all Buffer objects of the Buffer list to the input buffer queue even those in the output buffer queue. See function Buffer::QueueBuffer and DataStreamEventControl::GetFilledBuffer.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.13.2.9 FlushInputToOutputQueue()

```
BGAPI2::BufferList::FlushInputToOutputQueue ( )
```

This function moves all Buffer objects from the input buffer queue to the output buffer queue. See function Buffer::QueueBuffer and DataStreamEventControl::GetFilledBuffer.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.13.2.10 FlushUnqueuedToInputQueue()

`BGAPI2::BufferList::FlushUnqueuedToInputQueue ( )`

This function moves all free (not queued) Buffer objects of the Buffer list to the input buffer queue. See function Buffer::QueueBuffer.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.13.2.11 GetAnnouncedCount()

`BGAPI2::BufferList::GetAnnouncedCount ( )`

This function delivers the number of Buffer objects in the Buffer list. Same as BufferList::size.

Returns

bo_uint64 The number of Buffer objects in the Buffer list.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.13.2.12 GetAwaitDeliveryCount()

`BGAPI2::BufferList::GetAwaitDeliveryCount ( )`

This function delivers the number of Buffer objects in the output buffer queue. See function Data↵ StreamEventControl::GetFilledBuffer.

Returns

bo_uint64 The number of Buffer objects in the output buffer queue.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.13.2.13 GetDeliveredCount()

```
BGAPI2::BufferList::GetDeliveredCount ( )
```

This function delivers the number of Buffer objects that have been delivered since the start of the DataStream object.

Returns

bo_uint64 The number of Buffer object that have been delivered since the start of the DataStream object.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.13.2.14 GetQueuedCount()

```
BGAPI2::BufferList::GetQueuedCount ( )
```

This function delivers the number of Buffer objects in the input buffer queue. See function Buffer::QueueBuffer.

Returns

bo_uint64 The number of Buffer objects in the input buffer queue.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.13.2.15 GetStartedCount()

```
BGAPI2::BufferList::GetStartedCount ( )
```

This function delivers the number of Buffer objects which are currently being filled.

Returns

bo_uint64 The number of Buffer objects which are currently being filled.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.13.2.16 GetUnderrunCount()

```
BGAPI2::BufferList::GetUnderrunCount ( )
```

This function delivers the number of lost frames due to queue underrun since the DataStream object was started. This counter is incremented every time the data could not be acquired because there was no Buffer object in the input buffer queue.

Returns

bo_uint64 The number of lost frames.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.13.2.17 operator[]()

```
BGAPI2::BufferList::operator[] (
            const String & bufid )
```

This operator allows the direct access to an object of the Buffer list.

Parameters

bufid	For this ID, the associated Buffer object is delivered.

Returns

Buffer∗ The requested Buffer object.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::InvalidParameterException	No object in the Buffer list has the passed ID.

9.13.2.18 RevokeBuffer()

```
BGAPI2::BufferList::RevokeBuffer (
             Buffer ∗ pBuffer )
```

This function removes a Buffer object from the buffer list.

Parameters

pBuffer	The Buffer object to be removed.

Returns

void∗ This function delivers the user pointer of the Buffer object. See constructor of Buffer class.

Exceptions

Exceptions::InvalidParameterException	The passed parameter is not a valid Buffer object.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.13.2.19 size()

```
BGAPI2::BufferList::size ( )
```

This function delivers the number of Buffer objects in the Buffer list.

Returns

bo_uint64 The number of Buffer objects in the Buffer list.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.14 BGAPI2::Polarizer::formatlist::const_iterator Class Reference

This class provides a iterator that can read or modify any element in the list.

```
#include <bgapi2_genicam.hpp>
```

Public Member Functions

- bo_bool operator== (const const_iterator &right) const
  This operator compares their operands for equality.
- bo_bool operator!= (const const_iterator &right) const
  This operator compares their operands for inequality.
- const_iterator operator++ ()
  Postfix increment operator.
- const_iterator operator++ (int dummy)
  Prefix increment operator.
- const Formats & operator∗ ()
  The indirection operator dereferences the list pointer.
- const Formats ∗ operator-> ()
  Member access operator. This operator dereferences the list pointer.

Friends

- class **formatlist**

9.14.1 Detailed Description

This class provides a iterator that can read or modify any element in the list.

Definition at line 3531 of file bgapi2_genicam.hpp.

9.14.2 Member Function Documentation

9.14.2.1 operator"!=()

```
BGAPI2::Polarizer::formatlist::const_iterator::operator!= (
               const const_iterator & right ) const
```

This operator compares their operands for inequality.

Parameters

right	The second operand.

Returns

bo_bool The result of comparison.

9.14.2.2 operator*()

```
BGAPI2::Polarizer::formatlist::const_iterator::operator* ( )
```

The indirection operator dereferences the list pointer.

Returns

Formats.

9.14.2.3 operator++() [1/2]

```
BGAPI2::Polarizer::formatlist::const_iterator::operator++ ( )
```

Postfix increment operator.

Returns

const_iterator The iterator of the next list element.

9.14.2.4 operator++() [2/2]

```
BGAPI2::Polarizer::formatlist::const_iterator::operator++ (
            int dummy )
```

Prefix increment operator.

Returns

const_iterator The iterator of the next list element.

9.14.2.5 operator->()

```
BGAPI2::Polarizer::formatlist::const_iterator::operator-> ( )
```

Member access operator. This operator dereferences the list pointer.

Returns

Formats.

9.14.2.6 operator==()

```
BGAPI2::Polarizer::formatlist::const_iterator::operator== (
            const const_iterator & right ) const
```

This operator compares their operands for equality.

Parameters

right	The second operand.

Returns

bo_bool The result of comparison.

The documentation for this class was generated from the following file:

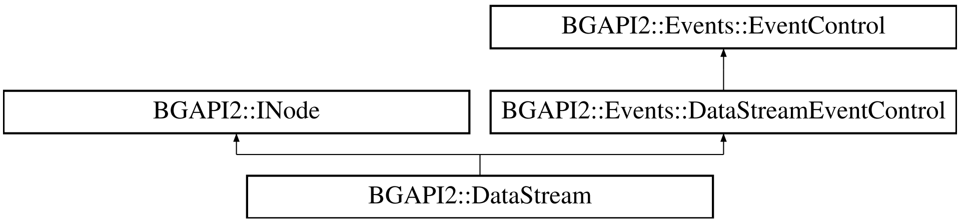
- bgapi2_genicam.hpp

9.15 BGAPI2::DataStream Class Reference

This class represents a physical data stream from the device and it is responsible for the buffer handling. This class belongs to the BGAPI2 main classes.

```
#include <bgapi2_genicam.hpp>
```

Inheritance diagram for BGAPI2::DataStream:


BGAPI2::Events::EventControl
BGAPI2::INode
BGAPI2::Events::DataStreamEventControl
BGAPI2::DataStream

Public Member Functions

- void Open ()

  This function opens the DataStream object and makes their functions available. An exception are the info-functions, which are already available before opening the DataStream object.
- void Close ()

  This function closes the DataStream object and releases the used resources.
- bo_bool IsOpen ()

  This function delivers true, if the data stream is opened.
- BufferList ∗ GetBufferList ()

  This function delivers the buffer list of the DataStream object.
- String GetID ()

  This function delivers the unique string identifier of the DataStream object, which is used in the DataStream list.
- String GetTLType ()

  This function delivers the transport layer type of the DataStream object.
- bo_bool GetDefinesPayloadSize ()

  This function delivers true, if the DataStream object supports the function DataStream::GetPayloadSize.
- bo_uint64 GetPayloadSize ()

  This function delivers the size of the expecting data block of this DataStream object in bytes based on the current device settings and including all control data (e.g. chunk header).
- bo_bool GetIsGrabbing ()

  This function delivers true, if the DataStream object is started. See functions DataStream::StartAcquisition and DataStream:StartAcquisitionContinuous.
- void StartAcquisition (bo_uint64 iNumToAcquire)

  This function starts the DataStream object. It is ready to receive data blocks. After receiving the specified number of data blocks the DataStrem object is automatically stopped.
- void StartAcquisitionContinuous ()

  This function starts the DataStream object. It is ready to receive data blocks. The DataStream object is started for an undefined number of transmissions until it is stopped by one of the two functions DataStream::StopAcquisition or DataStream::AbortAcquisition.
- void StopAcquisition ()

  This function stops the DataStream object directly if no transmission is active or after finishing a active transmission. It is not longer possible to receive data blocks.
- void AbortAcquisition ()

  This function stops the DataStream object immediately. Active transmissions are aborted. The aborted buffer gets the status 'incomplete'. See function Buffer::GetIsIncomplete.
- Buffer ∗ GetBufferByIndex (bo_uint iIndex)

  This function is deprecated. Please use instead the BufferList class.
- Device ∗ GetParent ()

  This function delivers the superordinate Device object.
- void ∗ GetReserved ()

  Undocumented function.

Friends

- class **DataStreamList**
- class **BufferList**
- class **Buffer**

9.15.1 Detailed Description

This class represents a physical data stream from the device and it is responsible for the buffer handling. This class belongs to the BGAPI2 main classes.

Definition at line 2513 of file bgapi2_genicam.hpp.

9.15.2 Member Function Documentation

9.15.2.1 AbortAcquisition()

```
BGAPI2::DataStream::AbortAcquisition ( )
```

This function stops the DataStream object immediately. Active transmissions are aborted. The aborted buffer gets the status 'incomplete'. See function Buffer::GetIsIncomplete.

Exceptions

Exceptions::NotInitializedException	The DataStream object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.15.2.2 Close()

```
BGAPI2::DataStream::Close ( )
```

This function closes the DataStream object and releases the used resources.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.15.2.3 GetBufferByIndex()

```
BGAPI2::DataStream::GetBufferByIndex (
            bo_uint iIndex )
```

This function is deprecated. Please use instead the BufferList class.

This function delivers a pointer to a specified Buffer object.

Parameters

iIndex	The index of the Buffer object.

Returns

Buffer∗ The requested pointer to the Buffer object.

Exceptions

Exceptions::NotInitializedException	The DataStream object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.15.2.4 GetBufferList()

```
BGAPI2::DataStream::GetBufferList ( )
```

This function delivers the buffer list of the DataStream object.

Returns

BufferList∗ The pointer to the data buffer list of the DataStream object.

Exceptions

Exceptions::NotInitializedException	The DataStream object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.15.2.5 GetDefinesPayloadSize()

```
BGAPI2::DataStream::GetDefinesPayloadSize ( )
```

This function delivers true, if the DataStream object supports the function DataStream::GetPayloadSize.

Returns

bo_bool If the delivered value is true, the DataStream object supports the function DataStream::GetPayloadSize.

Exceptions

Exceptions::NotInitializedException	The DataStream object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.15.2.6 GetID()

```
BGAPI2::DataStream::GetID ( )
```

This function delivers the unique string identifier of the DataStream object, which is used in the DataStream list.

Returns

String The unique string identifier.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.15.2.7 GetIsGrabbing()

```
BGAPI2::DataStream::GetIsGrabbing ( )
```

This function delivers true, if the DataStream object is started. See functions DataStream::StartAcquisition and DataStream:StartAcquisitionContinuous.

Returns

bo_bool If the delivered value is true, the DataStream object is started.

Exceptions

Exceptions::NotInitializedException	The DataStream object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.15.2.8 GetParent()

```
BGAPI2::DataStream::GetParent ( )
```

This function delivers the superordinate Device object.

Returns

Device∗ A pointer to the superordinate Device object.

9.15.2.9 GetPayloadSize()

```
BGAPI2::DataStream::GetPayloadSize ( )
```

This function delivers the size of the expecting data block of this DataStream object in bytes based on the current device settings and including all control data (e.g. chunk header).

This function is mainly used for devices which supports several data streams to allow stream based memory allocation.

Returns

bo_uint64 The size of the expected data block in bytes.

Exceptions

Exceptions::NotInitializedException	The DataStream object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.15.2.10 GetTLType()

```
BGAPI2::DataStream::GetTLType ( )
```

This function delivers the transport layer type of the DataStream object.

Returns

String The transport layer type of DataStream object.

Exceptions

Exceptions::NotInitializedException	The DataStream object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.15.2.11 IsOpen()

```
BGAPI2::DataStream::IsOpen ( )
```

This function delivers true, if the data stream is opened.

Returns

delivers true, if the data stream is open.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.15.2.12 Open()

```
BGAPI2::DataStream::Open ( )
```

This function opens the DataStream object and makes their functions available. An exception are the info-functions, which are already available before opening the DataStream object.

Exceptions

Exceptions::ResourceInUseException	This exception will be thrown if the DataStream object is already open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.15.2.13 StartAcquisition()

```
BGAPI2::DataStream::StartAcquisition (
            bo_uint64 iNumToAcquire )
```

This function starts the DataStream object. It is ready to receive data blocks. After receiving the specified number of data blocks the DataStrem object is automatically stopped.

A data block includes all data of the used payload type, e.g. for 'ChunkData' a data block includes all chunk blocks, for 'Image' a data block includes only image data.

Parameters

iNumToAcquire	The number of expected data blocks.

Exceptions

Exceptions::NotInitializedException	The DataStream object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.15.2.14 StartAcquisitionContinuous()

```
BGAPI2::DataStream::StartAcquisitionContinuous ( )
```

This function starts the DataStream object. It is ready to receive data blocks. The DataStream object is started for an undefined number of transmissions until it is stopped by one of the two functions DataStream::StopAcquisition or DataStream::AbortAcquisition.

Exceptions

Exceptions::NotInitializedException	The DataStream object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.15.2.15 StopAcquisition()

```
BGAPI2::DataStream::StopAcquisition ( )
```

This function stops the DataStream object directly if no transmission is active or after finishing a active transmission. It is not longer possible to receive data blocks.

Exceptions

Exceptions::NotInitializedException	The DataStream object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

The documentation for this class was generated from the following file:

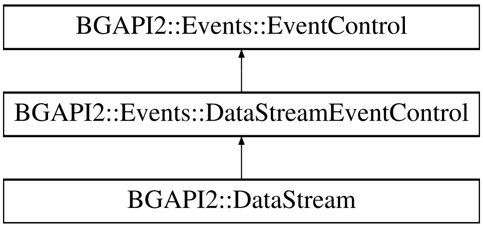
- bgapi2_genicam.hpp

9.16 BGAPI2::Events::DataStreamEventControl Class Reference

The class DataStreamEventControl provides the new buffer event which is used for fetching images.

`#include <bgapi2_genicam.hpp>`

Inheritance diagram for BGAPI2::Events::DataStreamEventControl:


BGAPI2::Events::EventControl
BGAPI2::Events::DataStreamEventControl
BGAPI2::DataStream

Public Member Functions

- void RegisterNewBufferEvent (EventMode eventMode)

  This function registers manually a NewBufferEvent at a DataStream object to detect the receiving of new images.
- void UnregisterNewBufferEvent ()

  This function unregisters manually a NewBufferEvent at a DataStream object.
- Buffer ∗ GetFilledBuffer (bo_uint64 iTimeout)

  This function fetches a new image from the DataStream object and removes it from the output buffer queue.
- void CancelGetFilledBuffer ()

  This function cancels the current waiting operation on the GetFilledBuffer function and not the image acquisition in the camera.
- void RegisterNewBufferEventHandler (void ∗callBackOwner, NewBufferEventHandler pFunc)

  Register a callback function which is called any time a new buffer was received.

9.16.1 Detailed Description

The class DataStreamEventControl provides the new buffer event which is used for fetching images.

Definition at line 2085 of file bgapi2_genicam.hpp.

9.16.2 Member Function Documentation

9.16.2.1 CancelGetFilledBuffer()

```
BGAPI2::Events::DataStreamEventControl::CancelGetFilledBuffer ( )
```

This function cancels the current waiting operation on the GetFilledBuffer function and not the image acquisition in the camera.

No image acquisition is aborted and an the data transfer will be finished. It is usefull to reduce stop/close operations for process and task.

Exceptions

Exceptions::NotInitializedException	The BGAPI object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.16.2.2 GetFilledBuffer()

```
BGAPI2::Events::DataStreamEventControl::GetFilledBuffer (
            bo_uint64 iTimeout )
```

This function fetches a new image from the DataStream object and removes it from the output buffer queue.

If the output buffer queue is empty after the timeout, the function delivers NULL.

Parameters

iTimeout	After this time, the function delivers latest.

Returns

Buffer∗ The buffer object which includes the new image or NULL in case of timeout.

Exceptions

Exceptions::NotInitializedException	The BGAPI object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::AbortException	The waiting operation of this function was canceled. See function CancelGetFilledBuffer.

9.16.2.3 RegisterNewBufferEvent()

```
BGAPI2::Events::DataStreamEventControl::RegisterNewBufferEvent (
            EventMode eventMode )
```

This function registers manually a NewBufferEvent at a DataStream object to detect the receiving of new images.

This function is currently not implemented. The NewBuffer event is already registered internally, when opening the DataStream object.

Parameters

eventMode	The event mode specifies the way how to fetch an incoming event.

Exceptions

Exceptions::NotImplementedException	This function is currently not implemented.
Exceptions::NotInitializedException	The object is not initialized properly.

9.16.2.4 RegisterNewBufferEventHandler()

```
BGAPI2::Events::DataStreamEventControl::RegisterNewBufferEventHandler (
            void * callBackOwner,
            NewBufferEventHandler pFunc )
```

Register a callback function which is called any time a new buffer was received.

Parameters

callBackOwner	Any object, stays at it is.
pFunc	Callback event handler.

Returns

void

Exceptions

Exceptions::NotInitializedException	The BGAPI object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.16.2.5 UnregisterNewBufferEvent()

```
BGAPI2::Events::DataStreamEventControl::UnregisterNewBufferEvent ( )
```

This function unregisters manually a NewBufferEvent at a DataStream object.

The NewBuffer event is already unregistered internally, when closing the DataStream object. This function is currently not implemented. If NewBufferEventHandler was registered, Thread will be shutdown, handler removed and all buffers discarded.

Exceptions

Exceptions::NotImplementedException	This function is currently not implemented.
Exceptions::NotInitializedException	The object is not initialized properly.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.17 BGAPI2::DataStreamList Class Reference

This class is used to discover and list data stream objects.

```
#include <bgapi2_genicam.hpp>
```

Classes

- class iterator

  This class provides an iterator that can read or modify any element in the list.

Public Member Functions

- void Refresh ()

  This function provides a list of DataStream objects.
- bo_uint64 size ()

  This function delivers the number of DataStream objects in the DataStream list.
- void clear ()

  This function removes all DataStream objects from the DataStream list.
- DataStream * operator[] (const String &streamid)

  This operator allows the direct access to an object of the DataStream list.
- iterator begin ()

  This function delivers an iterator on the top of the DataStream list.
- iterator end ()

  This function delivers an iterator at the end of the DataStream list.
- iterator find (const String &_keyval)

  This function delivers an iterator on an object to be found. If the object is not found, this functions delivers an end-iterator.

Friends

- class **Device**

9.17.1 Detailed Description

This class is used to discover and list data stream objects.

Definition at line 533 of file bgapi2_genicam.hpp.

9.17.2 Member Function Documentation

9.17.2.1 begin()

```
BGAPI2::DataStreamList::begin ( )
```

This function delivers an iterator on the top of the DataStream list.

Returns

iterator The iterator on the top of the DataStream list.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.17.2.2 clear()

```
BGAPI2::DataStreamList::clear ( )
```

This function removes all DataStream objects from the DataStream list.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.17.2.3 end()

```
BGAPI2::DataStreamList::end ( )
```

This function delivers an iterator at the end of the DataStream list.

Returns

iterator The iterator at the end of the DataStream list.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.17.2.4 find()

```
BGAPI2::DataStreamList::find (
            const String & _keyval )
```

This function delivers an iterator on an object to be found. If the object is not found, this functions delivers an end-iterator.

Parameters

_keyval	The ID to the object to be found.

Returns

iterator The iterator to the found object.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.17.2.5 operator[]()

```
BGAPI2::DataStreamList::operator[] (
            const String & streamid )
```

This operator allows the direct access to an object of the DataStream list.

Parameters

streamid	For this ID, the associated DataStream object is delivered.

Returns

DataStream∗ The requested DataStream object.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::InvalidParameterException	No object in the DataStream list has the passed ID.

9.17.2.6 Refresh()

```
BGAPI2::DataStreamList::Refresh ( )
```

This function provides a list of DataStream objects.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.17.2.7 size()

```
BGAPI2::DataStreamList::size ( )
```

This function delivers the number of DataStream objects in the DataStream list.

Returns

bo_uint64 The number of DataStream objects in the DataStream list.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

The documentation for this class was generated from the following file:

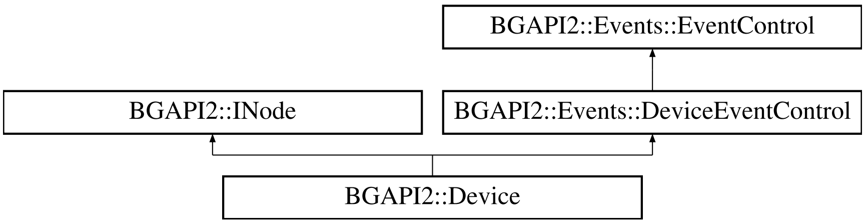
- bgapi2_genicam.hpp

9.18 BGAPI2::Device Class Reference

The class Device is used to retrieve information (e.g. model, manufacturer, access modes) of the device (camera) and also to control the device. This class belongs to the BGAPI2 main classes.

`#include <bgapi2_genicam.hpp>`

Inheritance diagram for BGAPI2::Device:


BGAPI2::Events::EventControl
BGAPI2::INode
BGAPI2::Events::DeviceEventControl
BGAPI2::Device

Public Member Functions

- void Open ()

  This function opens the device object and makes their functions available. Exceptions are the info-functions, which are already available before opening the device.
- void OpenExclusive ()

  This function opens the device object exclusive and makes their functions available. Exceptions are the info-functions, which are already available before opening the device.
- void OpenReadOnly ()

  This function opens the device object only for reading and make their functions available. Exceptions are the info-functions, which are already available before opening the device.
- void Close ()

  This function closes the device object and releases the used resources.
- bo_bool IsOpen ()

  This function delivers true, if the device is opened.
- void StartStacking (bo_bool bReplaceMode)

  This function starts the stacked mode for write commands (control commands). See remarks.
- void WriteStack ()

  This function writes the collected values to the device and stops the stacked mode.
- void CancelStack ()

  This function drops the collected values and stops the stacked mode.
- DataStreamList ∗ GetDataStreams ()

  This function delivers the data stream list of the device.
- String GetID ()

  This function delivers the unique string identifier of the Device, which is used in the DeviceList.
- String GetVendor ()

  This function delivers the name of the device vendor.
- String GetModel ()

  This function delivers the name of the device.
- String GetSerialNumber ()

  This function delivers the serial number of the device.
- String GetTLType ()

  This function delivers the transport layer type of the Device.
- String GetDisplayName ()

  This function delivers a meaningful name of the Device for display only.
- String GetAccessStatus ()

  This function delivers the access mode to the device. If this function is used before opening the device, it provides the possible access mode. The following possibilities can occur before opening the device.
- bo_uint64 GetPayloadSize ()

If the device supports a streaming channel for data transfer, this function delivers the size of the expecting data block in bytes based on the current device settings and including all control data (e.g. chunk header).

- Node ∗ GetRemoteNode (String name)

  The remote functions of the Device class are used to get access to the features of the physical (remote) device.
- NodeMap ∗ GetRemoteNodeTree ()

  The remote functions of the Device class are used to get access to the features of the physical (remote) device.
- NodeMap ∗ GetRemoteNodeList ()

  The remote functions of the Device class are used to get access to the features of the physical (remote) device.
- String GetRemoteConfigurationFile ()

  The remote functions of the Device class are used to get access to the features of the physical (remote) device.
- void SetRemoteConfigurationFile (String sConfigFile)

  The remote functions of the Device class are used to get access to the features of the physical (remote) device.
- bo_bool IsUpdateModeAvailable ()

  The update functions of the Device class are used to get access to firmware of the physical (remote) device.
- bo_bool IsUpdateModeActive ()

  The update functions of the Device class are used to get access to firmware of the physical (remote) device.
- void SetUpdateMode (bool bActive, String pcCustomKey)

  The update functions of the Device class are used to get access to firmware of the physical (remote) device.
- Node ∗ GetUpdateNode (String name)

  The update functions of the Device class are used to get access to the update features of the physical (remote) device.
- NodeMap ∗ GetUpdateNodeTree ()

  The update functions of the Device class are used to get access to the update features of the physical (remote) device.
- NodeMap ∗ GetUpdateNodeList ()

  The update functions of the Device class are used to get access to the update features of the physical (remote) device.
- String GetUpdateConfigurationFile ()

  The update functions of the Device class are used to get access to the update features of the physical (remote) device.
- Interface ∗ GetParent ()

  This function delivers the superordinate Interface object.
- void ∗ GetReserved ()

  Undocumented function.

Friends

- class **DeviceList**
- class **DataStreamList**
- class **DataStream**

9.18.1 Detailed Description

The class Device is used to retrieve information (e.g. model, manufacturer, access modes) of the device (camera) and also to control the device. This class belongs to the BGAPI2 main classes.

Definition at line 2678 of file bgapi2_genicam.hpp.

9.18.2 Member Function Documentation

9.18.2.1 CancelStack()

```
BGAPI2::Device::CancelStack ( )
```

This function drops the collected values and stops the stacked mode.

Exceptions

Exceptions::NotInitializedException	The device object is not open.
Exceptions::NoDataException	The stacked mode is not started or no control commands were written after calling the function Device::StartStacking.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.18.2.2 Close()

```
BGAPI2::Device::Close ( )
```

This function closes the device object and releases the used resources.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.18.2.3 GetAccessStatus()

```
BGAPI2::Device::GetAccessStatus ( )
```

This function delivers the access mode to the device. If this function is used before opening the device, it provides the possible access mode. The following possibilities can occur before opening the device.

If the function delivers with "RW" before opening, the device can be opened with one of the functions Device::Open or Device::OpenExclusive. If the function delivers with "RO" before opening, the device can be opend only with function Device::OpenReadOnly. If the function delivers with "NA" before opening, the device can not be opened. If the function delivers with "Unknown" before opening, the access mode could not be determined. The call to an open function may throw an exception. Is the device already opened, this function delivers the current access mode. The following variants can occur after opening the device. If the device was opened with the function Device::Open or Device::OpenExclusive, the function delivers "RW". If the device was opened with the function Device::OpenReadOnly, the function delivers "RO".

Returns

String "RW" - read and write access
String "RO" - only read access
String "NA" - No access, e.g. the device is not reachable (GEV).
String "Unknown" - The access mode could not be determined.

Exceptions

Exceptions::NotAvailableException	The GenTL producer delivers an unexpected access mode.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.18.2.4 GetDataStreams()

```
BGAPI2::Device::GetDataStreams ( )
```

This function delivers the data stream list of the device.

Returns

DataStreamList∗ The pointer to the data stream list of the device.

Exceptions

Exceptions::NotInitializedException	The interface object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.18.2.5 GetDisplayName()

```
BGAPI2::Device::GetDisplayName ( )
```

This function delivers a meaningful name of the Device for display only.

For Baumer devices this function delivers the value of the feature 'DeviceUserID'. If this feature is not available or empty the model name of the device is delivered instead.

Returns

String The meaningful name of the Device.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.18.2.6 GetID()

```
BGAPI2::Device::GetID ( )
```

This function delivers the unique string identifier of the Device, which is used in the DeviceList.

Returns

String The unique string identifier.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.18.2.7 GetModel()

```
BGAPI2::Device::GetModel ( )
```

This function delivers the name of the device.

Returns

String The name of the device.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.18.2.8 GetParent()

```
BGAPI2::Device::GetParent ( )
```

This function delivers the superordinate Interface object.

Returns

Interface∗ A pointer to the superordinate Interface object.

9.18.2.9 GetPayloadSize()

```
BGAPI2::Device::GetPayloadSize ( )
```

If the device supports a streaming channel for data transfer, this function delivers the size of the expecting data block in bytes based on the current device settings and including all control data (e.g. chunk header).

This function is mainly used to determine the size of image buffers to allocate. See also the functions DataStream::GetPayloadSize and DataStream::GetDefinesPayloadSize.

Returns

bo_uint64 The size of the expected data block in bytes.

Exceptions

Exceptions::NotInitializedException	The device object is not open.
Exceptions::NotAvailableException	This function is not supported.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.18.2.10 GetRemoteConfigurationFile()

```
BGAPI2::Device::GetRemoteConfigurationFile ( )
```

The remote functions of the Device class are used to get access to the features of the physical (remote) device.

This function delivers the GenICam XML File of the remote device.

Returns

String The GenICam XML file of the remote device.

Exceptions

Exceptions::NotInitializedException	The device object is not open.
Exceptions::NotAvailableException	The GenICam XML file of the remote device is not available.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.18.2.11 GetRemoteNode()

```
BGAPI2::Device::GetRemoteNode (
            String name )
```

The remote functions of the Device class are used to get access to the features of the physical (remote) device.

This function allows the access to a requested feature of the remote device. If the requested feature is not available, an exception is thrown.

Parameters

name	The requested feature of the remote device. For standardized features the predefined items in bgapi2_featurenames.h should be used.

Returns

Node∗ The requested feature of the remote device in form of a Node object.

Exceptions

Exceptions::NotInitializedException	The device object is not open.
Exceptions::NotAvailableException	The GenICam XML file of the remote device is not available.
Exceptions::InvalidParameterException	The requested feature is not available.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.18.2.12 GetRemoteNodeList()

```
BGAPI2::Device::GetRemoteNodeList ( )
```

The remote functions of the Device class are used to get access to the features of the physical (remote) device.

This function provides the features of the remote device as unstructured list in form of a NodeMap pointer. For a detailed description of the unstructured list representation refer to the class description of NodeMap.

Returns

NodeMap∗ The list of all features of the remote device in form of a NodeMap object.

Exceptions

Exceptions::NotInitializedException	The device object is not open.
Exceptions::NotAvailableException	The GenICam XML file of the remote device is not available.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.18.2.13 GetRemoteNodeTree()

```
BGAPI2::Device::GetRemoteNodeTree ( )
```

The remote functions of the Device class are used to get access to the features of the physical (remote) device.

This function provides the features of the remote device as tree structure in form of a NodeMap pointer. For a detailed description of the tree structure representation refer to the class description of NodeMap. The access to the features (elements) of the subordinated levels of hierarchy the functions Node::GetNodeTree and Node::GetNodeList should be used respectively.

Returns

NodeMap∗ The tree structure of all features of the remote device in form of a NodeMap object.

Exceptions

Exceptions::NotInitializedException	The device object is not open.
Exceptions::NotAvailableException	The GenICam XML file of the remote device is not available.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.18.2.14 GetSerialNumber()

```
BGAPI2::Device::GetSerialNumber ( )
```

This function delivers the serial number of the device.

Returns

String The serial number of the device.

Exceptions

Exceptions::NotImplementedException	This function is currently not implemented.

9.18.2.15 GetTLType()

```
BGAPI2::Device::GetTLType ( )
```

This function delivers the transport layer type of the Device.

Returns

String The name of the device.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.18.2.16 GetUpdateConfigurationFile()

```
BGAPI2::Device::GetUpdateConfigurationFile ( )
```

The update functions of the Device class are used to get access to the update features of the physical (remote) device.

This function delivers the GenICam XML File of the update features.

Returns

String The GenICam XML file of the update features.

Exceptions

Exceptions::NotInitializedException	The device object is not open.
Exceptions::NotAvailableException	The GenICam XML file of the remote device is not available.
Exceptions::ObjectInvalidException	The calling object is not valid.

Remarks

The update configuration file is only available if the update mode is active.

9.18.2.17 GetUpdateNode()

```
BGAPI2::Device::GetUpdateNode (
            String name )
```

The update functions of the Device class are used to get access to the update features of the physical (remote) device.

This function allows the access to a requested update feature of the remote device. If the requested update feature is not available, an exception is thrown.

Parameters

name	The requested update feature of the remote device.

Returns

Node∗ The requested update feature of the remote device in form of a Node object.

Exceptions

Exceptions::NotInitializedException	The device object is not open.
Exceptions::NotAvailableException	The GenICam XML file of the remote device is not available.
Exceptions::InvalidParameterException	The requested feature is not available.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

Remarks

The update nodes are only available if the update mode is active.

9.18.2.18 GetUpdateNodeList()

```
BGAPI2::Device::GetUpdateNodeList ( )
```

The update functions of the Device class are used to get access to the update features of the physical (remote) device.

This function provides the update features of the remote device as unstructured list in form of a NodeMap pointer. For a detailed description of the unstructured list representation refer to the class description of NodeMap.

Returns

NodeMap* The list of all update features of the remote device in form of a NodeMap object.

Exceptions

Exceptions::NotInitializedException	The device object is not open.
Exceptions::NotAvailableException	The GenICam XML file of the remote device is not available.
Exceptions::ObjectInvalidException	The calling object is not valid.

Remarks

The update node list is only available if the update mode is active.

9.18.2.19 GetUpdateNodeTree()

```
BGAPI2::Device::GetUpdateNodeTree ( )
```

The update functions of the Device class are used to get access to the update features of the physical (remote) device.

This function provides the update features of the remote device as tree structure in form of a NodeMap pointer. For a detailed description of the tree structure representation refer to the class description of NodeMap. The access to the update features (elements) of the subordinated levels of hierarchy the functions Node::GetNodeTree and Node::GetNodeList should be used respectively.

Returns

NodeMap* The tree structure of all update features of the remote device in form of a NodeMap object.

Exceptions

Exceptions::NotInitializedException	The device object is not open.
Exceptions::NotAvailableException	The GenICam XML file of the remote device is not available.
Exceptions::ObjectInvalidException	The calling object is not valid.

Remarks

The update node tree is only available if the update mode is active.

9.18.2.20 GetVendor()

```
BGAPI2::Device::GetVendor ( )
```

This function delivers the name of the device vendor.

Returns

String The name of the device vendor.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.18.2.21 IsOpen()

```
BGAPI2::Device::IsOpen ( )
```

This function delivers true, if the device is opened.

Returns

bo_bool delivers true, if the device is open.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.18.2.22 IsUpdateModeActive()

```
BGAPI2::Device::IsUpdateModeActive ( )
```

The update functions of the Device class are used to get access to firmware of the physical (remote) device.

This function delivers true if the update mode is active.

Returns

bo_bool delivers true, if the UpdateMode is active.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

Remarks

The update mode allows an firmware update of the device. The remote device features will not be available in the update mode.
In order to enable the update mode all data streams of the device have to be closed.
The update mode requires the BO_UpdateAPI2 library.

9.18.2.23 IsUpdateModeAvailable()

```
BGAPI2::Device::IsUpdateModeAvailable ( )
```

The update functions of the Device class are used to get access to firmware of the physical (remote) device.

This function delivers true if the update mode is available.

Returns

bo_bool delivers true, if the UpdateMode is available.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

Remarks

The update mode allows an firmware update of the device. The remote device features will not be available in the update mode.
In order to enable the update mode all data streams of the device have to be closed.
The update mode requires the BO_UpdateAPI2 library.

9.18.2.24 Open()

```
BGAPI2::Device::Open ( )
```

This function opens the device object and makes their functions available. Exceptions are the info-functions, which are already available before opening the device.

Exceptions

Exceptions::ResourceInUseException	This exception will be thrown if the device object is already opened.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

Remarks

GigEVision (GEV) devices supports several access methods for opening. The several methods are read access, read and write access and exclusive read and write access. If a device is opened for reading another application can get read and write access. If a device is opened for reading and writing one or more other applications only can get read access. If a device is opened for exclusive reading and writing the access from other applications is not possible.
Devices of other transport layer types (e.g. USB3Vision) supports only the exclusive read and write access, that means the open function opens the device for reading and writing. The access from other applications is not possible.

9.18.2.25 OpenExclusive()

```
BGAPI2::Device::OpenExclusive ( )
```

This function opens the device object exclusive and makes their functions available. Exceptions are the info-functions, which are already available before opening the device.

Exceptions

Exceptions::ResourceInUseException	This exception will be thrown if the device object is already opened.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

Remarks

GigEVision (GEV) devices supports several access methods for opening. The several methods are read access, read and write access and exclusive read and write access. If a device is opened for reading another application can get read and write access. If a device is opened for reading and writing one or more other applications only can get read access. If a device is opened for exclusive reading and writing the access from other applications is not possible.
Devices of other transport layer types (e.g. USB3Vision) supports only the exclusive read and write access, that means the open function opens the device for reading and writing. The access from other applications is not possible.

9.18.2.26 OpenReadOnly()

```
BGAPI2::Device::OpenReadOnly ( )
```

This function opens the device object only for reading and make their functions available. Exceptions are the info-functions, which are already available before opening the device.

Exceptions

Exceptions::ResourceInUseException	This exception will be thrown if the device object is already opened in exclusive mode.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

Remarks

GigEVision (GEV) devices supports several access methods for opening. The several methods are read access, read and write access and exclusive read and write access. If a device is opened for reading another application can get read and write access. If a device is opened for reading and writing one or more other applications only can get read access. If a device is opened for exclusive reading and writing the access from other applications is not possible.
Devices of other transport layer types (e.g. USB3Vision) supports only the exclusive read and write access, that means the open function opens the device for reading and writing. The access from other applications is not possible.

9.18.2.27 SetRemoteConfigurationFile()

```
BGAPI2::Device::SetRemoteConfigurationFile (
            String sConfigFile )
```

The remote functions of the Device class are used to get access to the features of the physical (remote) device.

This function sets a custom GenICam XML File of the remote device.

Parameters

sConfigFile	is a coustom config file of the remote device. Set to "" to use the default file of the device.

Exceptions

Exceptions::ResourceInUseException	The device object is open.
Exceptions::ObjectInvalidException	The calling object is not valid.

Remarks

In order to set the configuration file the device has to be closed.

9.18.2.28 SetUpdateMode()

```
BGAPI2::Device::SetUpdateMode (
            bool bActive,
            String pcCustomKey )
```

The update functions of the Device class are used to get access to firmware of the physical (remote) device.

This function enables or disables the update mode. See remarks.

Parameters

bActive	Enable or disable the update mode. This mode will control whether the nodemap contains update or bgapi features.
pcCustomKey	is reserved for future use and should be set to "".

Exceptions

Exceptions::NotInitializedException	The device object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::ResourceInUseException	This exception will be thrown if the update mode is already in the target state.

Remarks

The update mode allows an firmware update of the device. The remote device features will not be available in the update mode.
In order to enable the update mode all data streams of the device have to be closed.
The update mode requires the BO_UpdateAPI2 library.
If you perform an update, previously requested node objects of the remote device will become invalid.

9.18.2.29 StartStacking()

```
BGAPI2::Device::StartStacking (
            bo_bool bReplaceMode )
```

This function starts the stacked mode for write commands (control commands). See remarks.

Parameters

bReplaceMode	Enable or disable the replace mode. This mode will control whether all accesses to a register will be transferred to the device or only the last one.

Exceptions

Exceptions::NotInitializedException	The device object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.

Remarks

> The stacked mode is a high-performance method for transmitting write commands (control commands) to the device. The control commands were packed into one packet and transmitted as a concatenated command. The stacked mode is started by calling the function Device::Start↵ Stacking. All following control commands were collected and will not be transmitted until the calling of Device::WriteStack. If the replace mode is activated, only the last write acess will be transmitzted. After calling Device::WriteStack the stacked mode is stopped.

9.18.2.30 WriteStack()

```
BGAPI2::Device::WriteStack ( )
```

This function writes the collected values to the device and stops the stacked mode.

Exceptions

Exceptions::NotInitializedException	The device object is not open.
Exceptions::NoDataException	The stacked mode is not started or no control commands were written after calling the function Device::StartStacking.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

The documentation for this class was generated from the following file:

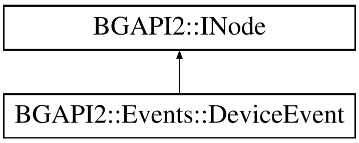
- bgapi2_genicam.hpp

9.19 BGAPI2::Events::DeviceEvent Class Reference

This class represents an device event which was received from the host. Use this class to get event information.

`#include <bgapi2_genicam.hpp>`

Inheritance diagram for BGAPI2::Events::DeviceEvent:



Public Member Functions

- DeviceEvent ()

    Constructor for creating an DeviceEvent object.
- ~DeviceEvent ()

    Destructor for deleting an DeviceEvent object.
- String GetName ()

    This function delivers the name of the DeviceEvent, which was specified in the XML description of the event.
- String GetDisplayName ()

    This function delivers the display name of the DeviceEvent, which was specified in the XML description of the event.
- bo_uint64 GetTimeStamp ()

    This function delivers the timestamp of the DeviceEvent.
- String GetId ()

    This function delivers the ID of the DeviceEvent, which was specified in the XML description of the event.

Friends

- class **DeviceEventControl**

9.19.1 Detailed Description

This class represents an device event which was received from the host. Use this class to get event information.

Definition at line 1951 of file bgapi2_genicam.hpp.

9.19.2 Member Function Documentation

9.19.2.1 GetDisplayName()

```
BGAPI2::Events::DeviceEvent::GetDisplayName ( )
```

This function delivers the display name of the DeviceEvent, which was specified in the XML description of the event.

Returns

String The display name of the device event.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::NotAvailableException	The display name was not included in the XML description of the event.

9.19.2.2 GetId()

```
BGAPI2::Events::DeviceEvent::GetId ( )
```

This function delivers the ID of the DeviceEvent, which was specified in the XML description of the event.

Returns

String The ID of the device event in String format.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::NoDataException	The event includes no data.

9.19.2.3 GetName()

```
BGAPI2::Events::DeviceEvent::GetName ( )
```

This function delivers the name of the DeviceEvent, which was specified in the XML description of the event.

Returns

String The name of the device event.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::NotAvailableException	The name was not included in the event.

9.19.2.4 GetTimeStamp()

```
BGAPI2::Events::DeviceEvent::GetTimeStamp ( )
```

This function delivers the timestamp of the DeviceEvent.

Returns

bo_uint64 The timestamp of the device event.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::NotAvailableException	The timestamp was not included in the event.

The documentation for this class was generated from the following file:

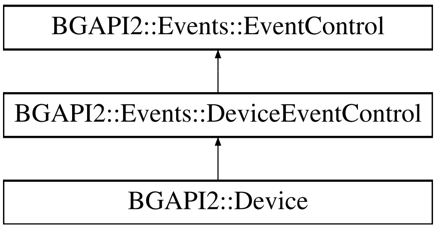
- bgapi2_genicam.hpp

9.20 BGAPI2::Events::DeviceEventControl Class Reference

The class DeviceEventControl provides access to standard events transmitted from the device.

`#include <bgapi2_genicam.hpp>`

Inheritance diagram for BGAPI2::Events::DeviceEventControl:


BGAPI2::Events::EventControl
BGAPI2::Events::DeviceEventControl
BGAPI2::Device

Public Member Functions

- void RegisterDeviceEvent (EventMode eventMode)

  This function registers a DeviceEvent at an Device object to detect asynchronous events from the physical connected device.
- void UnregisterDeviceEvent ()

  This function unregisters a DeviceEvent at an Device object.
- bo_bool GetDeviceEvent (DeviceEvent ∗pDeviceEvent, bo_uint64 iTimeout)

  This function fetches a DeviceEvent from the Device object.
- void CancelGetDeviceEvent ()

  This function cancels the current waiting operation on the GetDeviceEvent function.
- void RegisterDeviceEventHandler (void ∗callBackOwner, DeviceEventHandler pFunc)

  Register a callback function which is called any time a new device event was received.

9.20.1 Detailed Description

The class DeviceEventControl provides access to standard events transmitted from the device.

Definition at line 2015 of file bgapi2_genicam.hpp.

9.20.2 Member Function Documentation

9.20.2.1 CancelGetDeviceEvent()

```
BGAPI2::Events::DeviceEventControl::CancelGetDeviceEvent ( )
```

This function cancels the current waiting operation on the GetDeviceEvent function.

Exceptions

Exceptions::NotInitializedException	The BGAPI object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ErrorException	Unexpected Error. An internal pointer is not valid.

9.20.2.2 GetDeviceEvent()

```
BGAPI2::Events::DeviceEventControl::GetDeviceEvent (
            DeviceEvent * pDeviceEvent,
            bo_uint64 iTimeout )
```

This function fetches a DeviceEvent from the Device object.

Parameters

pDeviceEvent	A pointer to a valid object of type DeviceEvent. See function Events::DeviceEvent::DeviceEvent().
iTimeout	After this time, the function delivers latest.

Returns

bo_bool A flag that indicates whether a DeviceEvent was fetched.

Exceptions

Exceptions::NotInitializedException	The BGAPI object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::InvalidParameterException	The passed DeviceEvent object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.20.2.3 RegisterDeviceEvent()

```
BGAPI2::Events::DeviceEventControl::RegisterDeviceEvent (
            EventMode eventMode )
```

This function registers a DeviceEvent at an Device object to detect asynchronous events from the physical connected device.

Parameters

eventMode	The event mode specifies the way how to fetch an incoming event.

Exceptions

Exceptions::NotInitializedException	The BGAPI object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::InvalidParameterException	Only the event mode polling is available. See enumeration EventMode.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.20.2.4 RegisterDeviceEventHandler()

```
BGAPI2::Events::DeviceEventControl::RegisterDeviceEventHandler (
            void * callBackOwner,
            DeviceEventHandler pFunc )
```

Register a callback function which is called any time a new device event was received.

Parameters

callBackOwner	Any object, stays at it is.
pFunc	Callback event handler.

Exceptions

Exceptions::NotInitializedException	The BGAPI object is not open.
Exceptions::ErrorException	No memory available.

9.20.2.5 UnregisterDeviceEvent()

```
BGAPI2::Events::DeviceEventControl::UnregisterDeviceEvent ( )
```

This function unregisters a DeviceEvent at an Device object.

If DeviceEventHandler was registered, Thread will be shutdown and Handler removed.

Exceptions

Exceptions::NotInitializedException	The BGAPI object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.21 BGAPI2::DeviceList Class Reference

This class is used to discover and list device objects.

`#include <bgapi2_genicam.hpp>`

Classes

- class iterator

  This class provides a iterator that can read or modify any element in the list.

Public Member Functions

- void Refresh (bo_uint64 iTimeout)

  This function starts the search for Devices and provides a list of device objects. The search is controlled by a timeout parameter. This function delivers, if at least one device object were found, but no later than after the set timeout period (see comments).
- bo_uint64 size ()

  This function delivers the number of device objects in the device list.
- void clear ()

  This function removes all device objects from the device list.
- Device ∗ operator[ ] (const String &devid)

  This operator allows the direct access to an object of the device list.
- iterator begin ()

  This function delivers an iterator on the top of the device list.
- iterator end ()

  This function delivers an iterator at the end of the device list.
- iterator find (const String &_keyval)

  This function delivers an iterator on an object to be found. The object is not found, this functions delivers an end-iterator.

Friends

- class **Interface**

9.21.1 Detailed Description

This class is used to discover and list device objects.

Definition at line 386 of file bgapi2_genicam.hpp.

9.21.2 Member Function Documentation

9.21.2.1 begin()

```
BGAPI2::DeviceList::begin ( )
```

This function delivers an iterator on the top of the device list.

Returns

iterator The iterator on the top of the device list.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.21.2.2 clear()

```
BGAPI2::DeviceList::clear ( )
```

This function removes all device objects from the device list.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.21.2.3 end()

```
BGAPI2::DeviceList::end ( )
```

This function delivers an iterator at the end of the device list.

Returns

iterator The iterator at the end of the device list.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.21.2.4 find()

```
BGAPI2::DeviceList::find (
             const String & _keyval )
```

This function delivers an iterator on an object to be found. The object is not found, this functions delivers an end-iterator.

Parameters

_keyval	The ID to the object to be found.

Returns

iterator The iterator to the found object.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.21.2.5 operator[]()

```
BGAPI2::DeviceList::operator[] (
             const String & devid )
```

This operator allows the direct access to an object of the device list.

Parameters

devid	For this ID, the associated device object is delivered.

Returns

Device∗ The requested device object.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::InvalidParameterException	No object in the device list has the passed ID.

9.21.2.6 Refresh()

```
BGAPI2::DeviceList::Refresh (
            bo_uint64 iTimeout )
```

This function starts the search for Devices and provides a list of device objects. The search is controlled by a timeout parameter. This function delivers, if at least one device object were found, but no later than after the set timeout period (see comments).

Parameters

iTimeout	After this time, the function delivers latest.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

Remarks

This function has a minimum execution time, which also works when the timeout set is smaller then the minimum execution time.

9.21.2.7 size()

```
BGAPI2::DeviceList::size ( )
```

This function delivers the number of device objects in the device list.

Returns

bo_uint64 The number of device objects in the device list.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

The documentation for this class was generated from the following file:

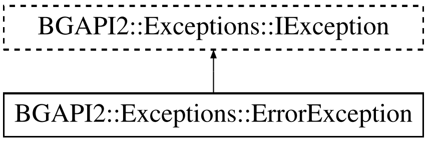
- bgapi2_genicam.hpp

9.22 BGAPI2::Exceptions::ErrorException Class Reference

General purpose exception.

`#include <bgapi2_genicam.hpp>`

Inheritance diagram for BGAPI2::Exceptions::ErrorException:


BGAPI2::Exceptions::IException
BGAPI2::Exceptions::ErrorException

Additional Inherited Members

9.22.1 Detailed Description

General purpose exception.

Definition at line 3670 of file bgapi2_genicam.hpp.

The documentation for this class was generated from the following file:

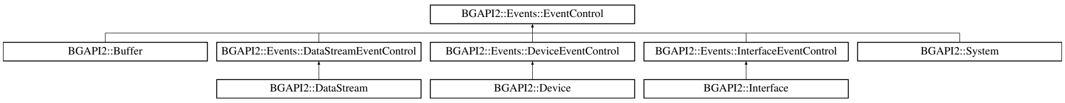
- bgapi2_genicam.hpp

9.23 BGAPI2::Events::EventControl Class Reference

The class EventControl provided access to custom events as well as the event mode.

`#include <bgapi2_genicam.hpp>`

Inheritance diagram for BGAPI2::Events::EventControl:


BGAPI2::Events::EventControl
BGAPI2::Buffer
BGAPI2::Events::DataStreamEventControl
BGAPI2::Events::DeviceEventControl
BGAPI2::Events::InterfaceEventControl
BGAPI2::System
BGAPI2::DataStream
BGAPI2::Device
BGAPI2::Interface

Public Member Functions

- EventMode GetEventMode ()

  This function delivers the current event mode setting. The event mode is controlled by the event register functions. See also enumeration Events::EventMode.
- void ∗ GetBase ()

  Undocumented function.

9.23.1 Detailed Description

The class EventControl provided access to custom events as well as the event mode.

Definition at line 1802 of file bgapi2_genicam.hpp.

9.23.2 Member Function Documentation

9.23.2.1 GetBase()

```
BGAPI2::Events::EventControl::GetBase ( )
```

Undocumented function.

Returns

void ∗

9.23.2.2 GetEventMode()

```
BGAPI2::Events::EventControl::GetEventMode ( )
```

This function delivers the current event mode setting. The event mode is controlled by the event register functions. See also enumeration Events::EventMode.

Returns

EventMode The current event mode.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.24 BGAPI2::Polarizer::formatlist Class Reference

This class provides a iterator that can read or modify any element in the list.

`#include <bgapi2_genicam.hpp>`

Classes

- class const_iterator
  This class provides a iterator that can read or modify any element in the list.

Public Member Functions

- bo_uint64 **size** ()
- const_iterator begin () const
  This function delivers a iterator on the top of the formatlist.
- const_iterator end () const
  This function delivers an iterator at the end of the formatlist.

9.24.1 Detailed Description

This class provides a iterator that can read or modify any element in the list.

Definition at line 3522 of file bgapi2_genicam.hpp.

9.24.2 Member Function Documentation

9.24.2.1 begin()

```
BGAPI2::Polarizer::formatlist::begin ( ) const
```

This function delivers a iterator on the top of the formatlist.

Returns

const_iterator The iterator on the top of the formatlist.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.24.2.2 end()

```
BGAPI2::Polarizer::formatlist::end ( ) const
```

This function delivers an iterator at the end of the formatlist.

Returns

const_iterator The iterator at the end of the formatlist.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

The documentation for this class was generated from the following file:

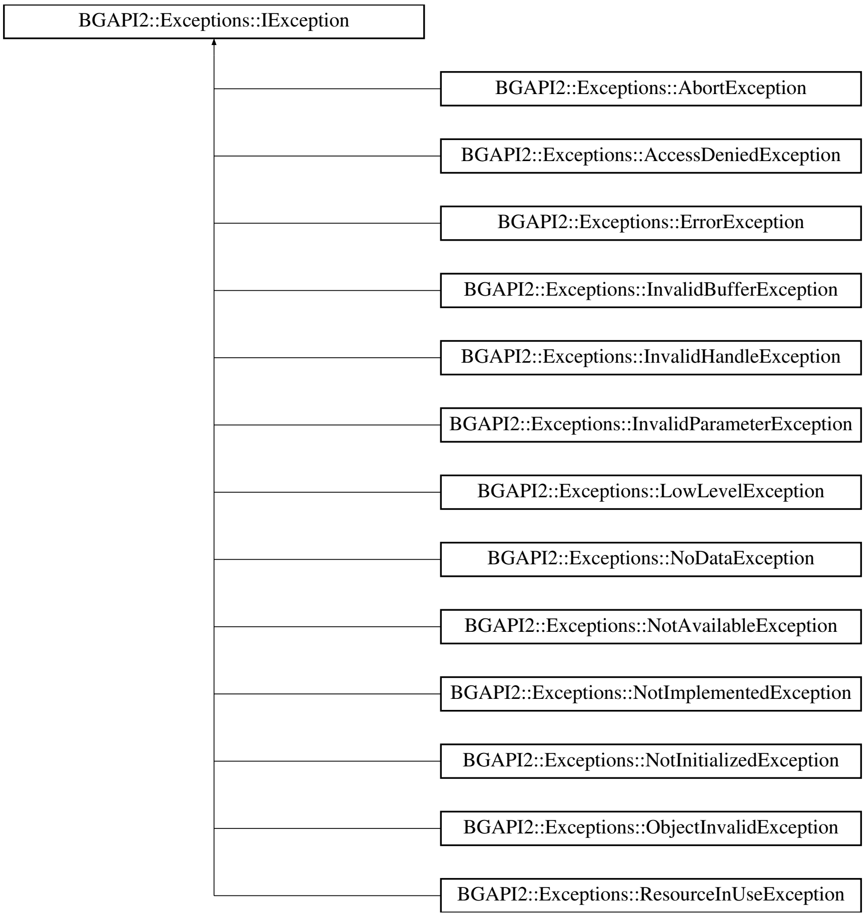
- bgapi2_genicam.hpp

9.25 BGAPI2::Exceptions::IException Class Reference

This class is responable for the exception handling and represents the parent class of all exception classes.

```
#include <bgapi2_genicam.hpp>
```

Inheritance diagram for BGAPI2::Exceptions::IException:


BGAPI2::Exceptions::IException
BGAPI2::Exceptions::AbortException
BGAPI2::Exceptions::AccessDeniedException
BGAPI2::Exceptions::ErrorException
BGAPI2::Exceptions::InvalidBufferException
BGAPI2::Exceptions::InvalidHandleException
BGAPI2::Exceptions::InvalidParameterException
BGAPI2::Exceptions::LowLevelException
BGAPI2::Exceptions::NoDataException
BGAPI2::Exceptions::NotAvailableException
BGAPI2::Exceptions::NotImplementedException
BGAPI2::Exceptions::NotInitializedException
BGAPI2::Exceptions::ObjectInvalidException
BGAPI2::Exceptions::ResourceInUseException

Public Member Functions

- virtual String GetErrorDescription ()=0

  This function delivers a error description of the exception.
- virtual String GetFunctionName ()=0

  This function delivers the name of the function which has thrown the exception.
- virtual String GetType ()=0

  This function delivers the type of the exception. The type of the exception is the name of the exception class.

9.25.1 Detailed Description

This class is responable for the exception handling and represents the parent class of all exception classes.

Definition at line 3641 of file bgapi2_genicam.hpp.

9.25.2 Member Function Documentation

9.25.2.1 GetErrorDescription()

`BGAPI2::Exceptions::IException::GetErrorDescription ( ) [pure virtual]`

This function delivers a error description of the exception.

Returns

String The error description of the exception.

9.25.2.2 GetFunctionName()

`BGAPI2::Exceptions::IException::GetFunctionName ( ) [pure virtual]`

This function delivers the name of the function which has thrown the exception.

Returns

String The name of the function which has thrown the exception.

9.25.2.3 GetType()

`BGAPI2::Exceptions::IException::GetType ( ) [pure virtual]`

This function delivers the type of the exception. The type of the exception is the name of the exception class.

Returns

String The type of the exception.

The documentation for this class was generated from the following file:

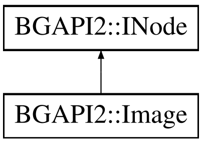
- bgapi2_genicam.hpp

9.26 BGAPI2::Image Class Reference

The class Image provides the ability of image transformation. This class belongs to the additional classes.

`#include <bgapi2_genicam.hpp>`

Inheritance diagram for BGAPI2::Image:



Public Member Functions

- bo_uint GetWidth ()
  This function delivers the width of the image in pixel.
- bo_uint GetHeight ()
  This function delivers the height of the image in pixel.
- String GetPixelformat ()
  This function delivers the pixel format of the Image object.
- void ∗ GetBuffer ()
  This function delivers a pointer to the memory of the Image object.
- bo_uint64 GetTransformBufferLength (String sPixelFormat)
  This function delivers the destination buffer size after the transformation.
- void GetHistogram (bo_tHistRecords tHistogram, bo_uint offsetx, bo_uint offsety, bo_uint width, bo_uint height)
  This function delivers a histogram of a specified region of the Image object.
- void GetHistogram (bo_tHistRecords tHistogram)
  This function delivers a histogram of the complete Image object.
- void Release ()
  This function frees all used resoures.
- void Init (bo_uint width, bo_uint height, String sPixelFormat, void ∗pBuffer, bo_uint64 uBufferSize)
  This function reinitialise an Image object.

Friends

- class **ImageProcessor**
- class **Polarizer**

9.26.1 Detailed Description

The class Image provides the ability of image transformation. This class belongs to the additional classes.

Definition at line 3260 of file bgapi2_genicam.hpp.

9.26.2 Member Function Documentation

9.26.2.1 GetBuffer()

```
BGAPI2::Image::GetBuffer ( )
```

This function delivers a pointer to the memory of the Image object.

Returns

void∗ The pointer to the memory of Image object.

9.26.2.2 GetHeight()

```
BGAPI2::Image::GetHeight ( )
```

This function delivers the height of the image in pixel.

Returns

bo_uint The height of the image in pixel.

9.26.2.3 GetHistogram() [1/2]

```
BGAPI2::Image::GetHistogram (
            bo_tHistRecords tHistogram,
            bo_uint offsetx,
            bo_uint offsety,
            bo_uint width,
            bo_uint height )
```

This function delivers a histogram of a specified region of the Image object.

Parameters

tHistogram	A structure which includes the histogram data.
offsetx	The left coordinate of the ROI.
offsety	The top coordinate of the ROI.
width	The width of the ROI.
height	The height of the ROI.

Exceptions

Exceptions::ErrorException	Function returns with error.

9.26.2.4 GetHistogram() [2/2]

```
BGAPI2::Image::GetHistogram (
            bo_tHistRecords tHistogram )
```

This function delivers a histogram of the complete Image object.

Parameters

tHistogram	A structure which includes the histogram data.

Exceptions

Exceptions::ErrorException	Function returns with error.

9.26.2.5 GetPixelformat()

```
BGAPI2::Image::GetPixelformat ( )
```

This function delivers the pixel format of the Image object.

Returns

String The pixelformat of the Image object.

9.26.2.6 GetTransformBufferLength()

```
BGAPI2::Image::GetTransformBufferLength (
            String sPixelFormat )
```

This function delivers the destination buffer size after the transformation.

Parameters

sPixelFormat	The pixel format for the transformation.

Returns

bo_uint64 The destination buffer size after the transformation.

Exceptions

Exceptions::ErrorException	Function returned with error.

9.26.2.7 GetWidth()

```
BGAPI2::Image::GetWidth ( )
```

This function delivers the width of the image in pixel.

Returns

bo_uint The width of the image in pixel.

9.26.2.8 Init()

```
BGAPI2::Image::Init (
            bo_uint width,
            bo_uint height,
            String sPixelFormat,
            void * pBuffer,
            bo_uint64 uBufferSize )
```

This function reinitialise an Image object.

Parameters

width	The width of the Image object in pixel.
height	The height of the Image object in pixel.
sPixelFormat	The pixelformat of the Image object.
pBuffer	The user defined image buffer.
uBufferSize	The size of the user defined image buffer bytes.

Exceptions

Exceptions::ErrorException	Function returns with error.

The documentation for this class was generated from the following file:

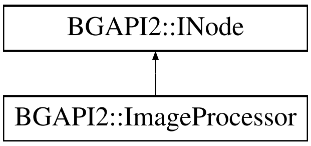
- bgapi2_genicam.hpp

9.27 BGAPI2::ImageProcessor Class Reference

The task of the class ImageProcessor are the creation of image objects and the transformation of pixel formats.

```
#include <bgapi2_genicam.hpp>
```

Inheritance diagram for BGAPI2::ImageProcessor:



Public Member Functions

- ImageProcessor ()

  Parameterless constructor for creating of an ImageProcessor-Object.
- ∼ImageProcessor ()

  destructor to destroy an ImageProcessor object.
- String GetVersion ()

  Get the version number of the ImageProcessor library. The delivered string is composed of these several tags: Major.Minor.LastChanged.RevId.
- Image ∗ CreateImage ()

  This function creates an empty Image object.
- Image ∗ **CreateImage** (bo_uint width, bo_uint height, String pixelformat, void ∗pBuffer, bo_↩ uint64 uBufferSize)
- Image ∗ CreateTransformedImage (Image ∗pInputImage, const char ∗szDestinationPixelformat)

  This function transforms the pixelformat of the passed Image object into a new created Image object.
- void TransformImageToBuffer (Image ∗pInputImage, const char ∗szDestinationPixelformat, void ∗pBuffer, bo_uint64 uBufferSize)

  This function transformes the pixel format of the Image object and writes the data into the passed destination buffer.

9.27.1 Detailed Description

The task of the class ImageProcessor are the creation of image objects and the transformation of pixel formats.

This class belongs to the additional classes. A transformation can be the demosaicing of raw sensor pixel arrangements in a common displayable pixelformat, e.g. RAWBAYER to RGB8 or RGB to YUV. This class is based on DLL "bgapi2_img.dll".

Definition at line 3352 of file bgapi2_genicam.hpp.

9.27.2 Member Function Documentation

9.27.2.1 CreateImage()

```
BGAPI2::ImageProcessor::CreateImage ( )
```

This function creates an empty Image object.

This function creates an Image object according the parameters width, height and pixelformat. The Image object uses a user defines buffer.

Returns

Image∗ The pointer to a new created Image object.

Parameters

width	The width of the Image object in pixel.
height	The height of the Image object in pixel.
pixelformat	The pixelformat of the Image object.
pBuffer	The user defined image buffer.
uBufferSize	The size of the user defined image buffer bytes.

Returns

Image∗ The pointer to a new created Image object.

See also

Image::GetTransformBufferLength

9.27.2.2 CreateTransformedImage()

```
BGAPI2::ImageProcessor::CreateTransformedImage (
            Image * pInputImage,
            const char * szDestinationPixelformat )
```

This function transforms the pixelformat of the passed Image object into a new created Image object.

Parameters

pInputImage	A pointer to a Image object whose image data is to be transformed.
szDestinationPixelformat	The desired destination pixel format of the delivered Image object.

Returns

Image∗ A pointer to a new created Image object which includes the transformed image data.

9.27.2.3 GetVersion()

```
BGAPI2::ImageProcessor::GetVersion ( )
```

Get the version number of the ImageProcessor library. The delivered string is composed of these several tags: Major.Minor.LastChanged.RevId.

Returns

String The version number of the ImageProcessor library.

9.27.2.4 TransformImageToBuffer()

```
BGAPI2::ImageProcessor::TransformImageToBuffer (
            Image * pInputImage,
            const char * szDestinationPixelformat,
            void * pBuffer,
            bo_uint64 uBufferSize )
```

This function transformes the pixel format of the Image object and writes the data into the passed destination buffer.

Parameters

pInputImage	A pointer to an Image object whose image data is to be transformed.
szDestinationPixelformat	The destination pixel format.
pBuffer	The destination buffer.
uBufferSize	The destination buffer size.

Exceptions

Exceptions::ErrorException	Error while transform the image.

The documentation for this class was generated from the following file:

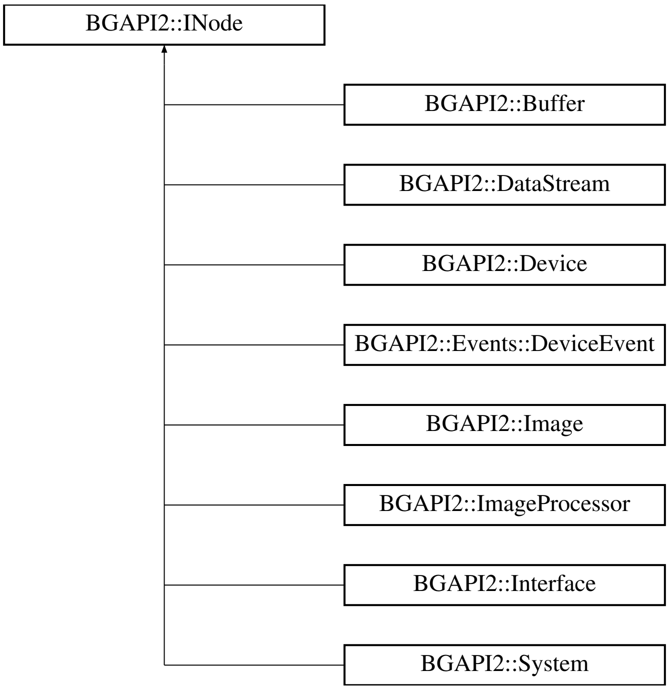
- bgapi2_genicam.hpp

9.28 BGAPI2::INode Class Reference

The class INode act as base for of the main classes and provided the access to the node objects (features).

```
#include <bgapi2_genicam.hpp>
```

Inheritance diagram for BGAPI2::INode:



Public Member Functions

- Node ∗ GetNode (String name)

  This function delivers a certain object of the Node list and provides the access to a node objcet without the detour across a NodeMap object. This function is functional identical to NodeMap::GetNode.
- NodeMap ∗ GetNodeTree ()

  This function provides the features of a object derived from INode interface as tree structure in form of a NodeMap pointer. For a detailed description of the tree structure representation refer to the class description of NodeMap.
- NodeMap ∗ GetNodeList ()

  This function provides the features of a object derived from INode interface as unstructured list in form of a NodeMap pointer. For a detailed description of the unstructured list representation refer to the class description of NodeMap.

9.28.1 Detailed Description

The class INode act as base for of the main classes and provided the access to the node objects (features).

Definition at line 1722 of file bgapi2_genicam.hpp.

9.28.2 Member Function Documentation

9.28.2.1 GetNode()

```
BGAPI2::INode::GetNode (
            String name )
```

This function delivers a certain object of the Node list and provides the access to a node objcet without the detour across a NodeMap object. This function is functional identical to NodeMap::GetNode.

Parameters

name	For this name, the associated Node object is delivered.

Returns

Node∗ The requested Node object.

Exceptions

Exceptions::InvalidParameterException	No object in the Node list has the passed name.
Exceptions::NotInitializedException	The BGAPI object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.28.2.2 GetNodeList()

```
BGAPI2::INode::GetNodeList ( )
```

This function provides the features of a object derived from INode interface as unstructured list in form of a NodeMap pointer. For a detailed description of the unstructured list representation refer to the class description of NodeMap.

Returns

NodeMap∗ The list of all features of a object derived from INode interface.

Exceptions

Exceptions::NotInitializedException	The BGAPI object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.28.2.3 GetNodeTree()

```
BGAPI2::INode::GetNodeTree ( )
```

This function provides the features of a object derived from INode interface as tree structure in form of a NodeMap pointer. For a detailed description of the tree structure representation refer to the class description of NodeMap.

Returns

NodeMap∗ The tree structure of all features of a object derived from INode interface.

Exceptions

Exceptions::NotInitializedException	The BGAPI object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.

The documentation for this class was generated from the following file:

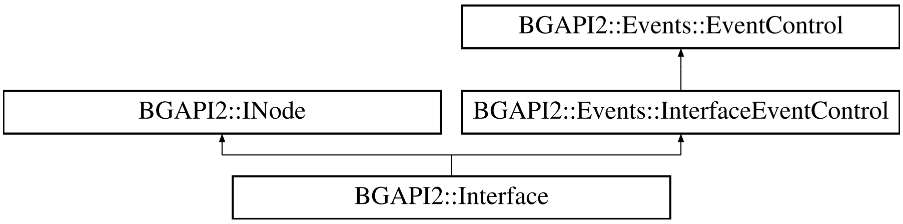
- bgapi2_genicam.hpp

9.29 BGAPI2::Interface Class Reference

The class Interface represents a physical interface, e.g. GEV or a logical interface, such as USB and belongs to the BGAPI2 main classes.

```
#include <bgapi2_genicam.hpp>
```

Inheritance diagram for BGAPI2::Interface:


BGAPI2::Events::EventControl
BGAPI2::INode
BGAPI2::Events::InterfaceEventControl
BGAPI2::Interface

Public Member Functions

- void Open ()

    This function opens the interface object and make their functions available. Exceptions are the info-functions, which are already available before opening the interface.
- void Close ()

    This function closes the interface object and releases the used resources.
- bo_bool IsOpen ()

    This function delivers true, if the interface is opened.
- DeviceList ∗ GetDevices ()

    This function delivers the device list of the interface.
- String GetID ()

    This function delivers the unique string identifier of the Interface, which is used in the InterfaceList.
- String GetDisplayName ()

    This function delivers a meaningful name of the Interface for display only.
- String GetTLType ()

    This function delivers the transport layer type of the Interface.
- System ∗ GetParent ()

    This function delivers the superordinate System object.

Friends

- class **InterfaceList**
- class **DeviceList**
- class **Device**

9.29.1 Detailed Description

The class Interface represents a physical interface, e.g. GEV or a logical interface, such as USB and belongs to the BGAPI2 main classes.

Definition at line 3033 of file bgapi2_genicam.hpp.

9.29.2 Member Function Documentation

9.29.2.1 Close()

```
BGAPI2::Interface::Close ( )
```

This function closes the interface object and releases the used resources.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.29.2.2 GetDevices()

```
BGAPI2::Interface::GetDevices ( )
```

This function delivers the device list of the interface.

Returns

DeviceList* The pointer to the device list of the interface.

Exceptions

Exceptions::NotInitializedException	The interface object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.29.2.3 GetDisplayName()

```
BGAPI2::Interface::GetDisplayName ( )
```

This function delivers a meaningful name of the Interface for display only.

Returns

String The meaningful name of the Interface.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.29.2.4 GetID()

```
BGAPI2::Interface::GetID ( )
```

This function delivers the unique string identifier of the Interface, which is used in the InterfaceList.

Returns

String The unique string identifier.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.29.2.5 GetParent()

```
BGAPI2::Interface::GetParent ( )
```

This function delivers the superordinate System object.

Returns

System∗ A pointer to the superordinate System object.

9.29.2.6 GetTLType()

```
BGAPI2::Interface::GetTLType ( )
```

This function delivers the transport layer type of the Interface.

Returns

String The transport layer type of Interface.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.29.2.7 IsOpen()

```
BGAPI2::Interface::IsOpen ( )
```

This function delivers true, if the interface is opened.

Returns

delivers true, if the interface is open.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.29.2.8 Open()

```
BGAPI2::Interface::Open ( )
```

This function opens the interface object and make their functions available. Exceptions are the info-functions, which are already available before opening the interface.

Exceptions

Exceptions::ResourceInUseException	This exception will be thrown if the interface object is already open.
Exceptions::ObjectInvalidException	The calling object is not valid.

Exceptions

Exceptions::NotInitializedException	The corresponding system object is not initialized.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

The documentation for this class was generated from the following file:

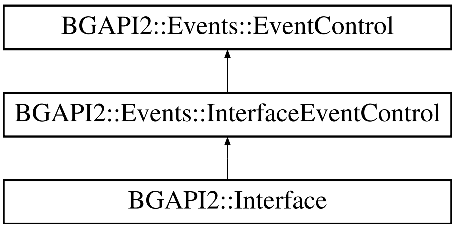
- bgapi2_genicam.hpp

9.30 BGAPI2::Events::InterfaceEventControl Class Reference

The class InterfaceEventControl provides access to interface specific events, e.g. plug'n play event.

`#include <bgapi2_genicam.hpp>`

Inheritance diagram for BGAPI2::Events::InterfaceEventControl:


BGAPI2::Events::EventControl
BGAPI2::Events::InterfaceEventControl
BGAPI2::Interface

Public Member Functions

- void RegisterPnPEvent (EventMode eventMode)

  This function registers a PnPEvent at an Interface object to detect adding and removing of Device objects.
- void UnregisterPnPEvent ()

  This function unregisters a PnPEvent at an Interface object.
- bo_bool GetPnPEvent (PnPEvent ∗pPnPEvent, bo_uint64 iTimeout)

  This function fetches a PnPEvent from the Interface object.
- void CancelGetPnPEvent ()

  This function cancels the current waiting operation on the GetPnPEvent function.
- void RegisterPnPEventHandler (void ∗callBackOwner, PnPEventHandler pFunc)

  Register a callback function which is called any time a new plug 'n play event was received.

9.30.1 Detailed Description

The class InterfaceEventControl provides access to interface specific events, e.g. plug'n play event.

Definition at line 1886 of file bgapi2_genicam.hpp.

9.30.2 Member Function Documentation

9.30.2.1 CancelGetPnPEvent()

```
BGAPI2::Events::InterfaceEventControl::CancelGetPnPEvent ( )
```

This function cancels the current waiting operation on the GetPnPEvent function.

Exceptions

Exceptions::NotInitializedException	The BGAPI object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ErrorException	Unexpected Error. An internal pointer is not valid.

9.30.2.2 GetPnPEvent()

```
BGAPI2::Events::InterfaceEventControl::GetPnPEvent (
            PnPEvent * pPnPEvent,
            bo_uint64 iTimeout )
```

This function fetches a PnPEvent from the Interface object.

Parameters

pPnPEvent	A pointer to a valid object of type PnPEvent. See function Events::PnPEvent::PnpEvent().
iTimeout	After this time, the function delivers latest.

Returns

bo_bool A flag that indicates whether a PnPEvent was fetched.

Exceptions

Exceptions::NotInitializedException	The BGAPI object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::InvalidParameterException	The passed PnPEvent object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.30.2.3 RegisterPnPEvent()

```
BGAPI2::Events::InterfaceEventControl::RegisterPnPEvent (
            EventMode eventMode )
```

This function registers a PnPEvent at an Interface object to detect adding and removing of Device objects.

Parameters

eventMode	The event mode specifies the way how to fetch an incoming event.

Exceptions

Exceptions::NotInitializedException	The BGAPI object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::InvalidParameterException	Only the event mode polling is available. See enumeration EventMode.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.30.2.4 RegisterPnPEventHandler()

```
BGAPI2::Events::InterfaceEventControl::RegisterPnPEventHandler (
            void * callBackOwner,
            PnPEventHandler pFunc )
```

Register a callback function which is called any time a new plug 'n play event was received.

Parameters

callBackOwner	Any object, stays at it is.
pFunc	Callback event handler.

Exceptions

Exceptions::NotInitializedException	The BGAPI object is not open.
Exceptions::ErrorException	No memory available.

9.30.2.5 UnregisterPnPEvent()

```
BGAPI2::Events::InterfaceEventControl::UnregisterPnPEvent ( )
```

This function unregisters a PnPEvent at an Interface object.

If RegisterPnPEventHandler was registered, Thread will be shutdown and Handler removed.

Exceptions

Exceptions::NotInitializedException	The BGAPI object is not open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.31 BGAPI2::InterfaceList Class Reference

This class is used to search and list interface objects.

```
#include <bgapi2_genicam.hpp>
```

Classes

- class iterator

  This class provides a iterator that can read or modify any element in the list.

Public Member Functions

- void Refresh (bo_uint64 iTimeout)

  This function starts the search for Interfaces and provides a list of interface objects. The search is controlled by a timeout parameter. This function delivers, if at least one interface object was found, but not later than after the set timeout period (see comments).
- bo_uint64 size ()

  This function delivers the number of interface objects in the interface list.
- void clear ()

  This function removes all interface objects from the interface list.
- Interface $*$ operator[] (const String &ifaceid)

  This operator allows the direct access to an object of the interface list.
- iterator begin ()

  This functions delivers an iterator on the top of the interface list.
- iterator end ()

  This functions delivers an iterator at the end of the interface list.
- iterator find (const String &_keyval)

  This function delivers an iterator on an object to be found. If the object cannot be found, this functions delivers an end-iterator.

Friends

- class **System**

9.31.1 Detailed Description

This class is used to search and list interface objects.

Definition at line 239 of file bgapi2_genicam.hpp.

9.31.2 Member Function Documentation

9.31.2.1 begin()

```
BGAPI2::InterfaceList::begin ( )
```

This functions delivers an iterator on the top of the interface list.

Returns

iterator The iterator on the top of the interface list.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.31.2.2 clear()

```
BGAPI2::InterfaceList::clear ( )
```

This function removes all interface objects from the interface list.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.31.2.3 end()

```
BGAPI2::InterfaceList::end ( )
```

This functions delivers an iterator at the end of the interface list.

Returns

iterator The iterator at the end of the interface list.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.31.2.4 find()

```
BGAPI2::InterfaceList::find (
             const String & _keyval )
```

This function delivers an iterator on an object to be found. If the object cannot be found, this functions delivers an end-iterator.

Parameters

_keyval	The ID to the object to be found.

Returns

iterator The iterator to the found object.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.31.2.5 operator[]()

```
BGAPI2::InterfaceList::operator[] (
             const String & ifaceid )
```

This operator allows the direct access to an object of the interface list.

Parameters

ifaceid	For this ID, the associated system object is delivered.

Returns

Interface∗ The requested interface object.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::InvalidParameterException	No object in the interface list has the passed ID.

9.31.2.6 Refresh()

```
BGAPI2::InterfaceList::Refresh (
            bo_uint64 iTimeout )
```

This function starts the search for Interfaces and provides a list of interface objects. The search is controlled by a timeout parameter. This function delivers, if at least one interface object was found, but not later than after the set timeout period (see comments).

Parameters

iTimeout	After this time, the function delivers latest.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

Remarks

This function has a minimum execution time, which also works when the timeout set is smaller then the minimum execution time.

9.31.2.7 size()

```
BGAPI2::InterfaceList::size ( )
```

This function delivers the number of interface objects in the interface list.

Returns

bo_uint64 The number of interface objects in the interface list.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

The documentation for this class was generated from the following file:

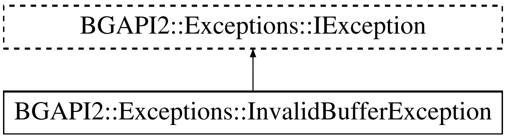
- bgapi2_genicam.hpp

9.32 BGAPI2::Exceptions::InvalidBufferException Class Reference

Invalid buffer is used. The used Buffer object is not valid.

`#include <bgapi2_genicam.hpp>`

Inheritance diagram for BGAPI2::Exceptions::InvalidBufferException:


BGAPI2::Exceptions::IException
BGAPI2::Exceptions::InvalidBufferException

Additional Inherited Members

9.32.1 Detailed Description

Invalid buffer is used. The used Buffer object is not valid.

Definition at line 3752 of file bgapi2_genicam.hpp.

The documentation for this class was generated from the following file:

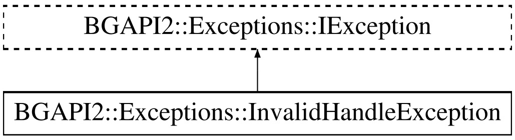
- bgapi2_genicam.hpp

9.33 BGAPI2::Exceptions::InvalidHandleException Class Reference

(Given handle does not support the operation.)

`#include <bgapi2_genicam.hpp>`

Inheritance diagram for BGAPI2::Exceptions::InvalidHandleException:


BGAPI2::Exceptions::IException
BGAPI2::Exceptions::InvalidHandleException

Additional Inherited Members

9.33.1 Detailed Description

(Given handle does not support the operation.)

Definition at line 3715 of file bgapi2_genicam.hpp.

The documentation for this class was generated from the following file:

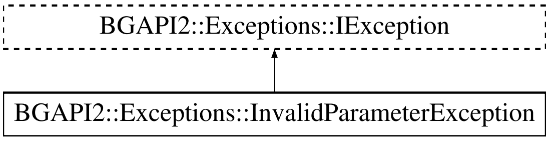
- bgapi2_genicam.hpp

9.34 BGAPI2::Exceptions::InvalidParameterException Class Reference

One of the parameter given was not valid or out of range.

`#include <bgapi2_genicam.hpp>`

Inheritance diagram for BGAPI2::Exceptions::InvalidParameterException:


BGAPI2::Exceptions::IException
BGAPI2::Exceptions::InvalidParameterException

Additional Inherited Members

9.34.1 Detailed Description

One of the parameter given was not valid or out of range.

Definition at line 3733 of file bgapi2_genicam.hpp.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.35 BGAPI2::NodeMap::iterator Class Reference

This class provides a iterator that can read or modify any element in the list.

`#include <bgapi2_genicam.hpp>`

Public Member Functions

- iterator & operator= (const iterator &_iterator)

  Assignment operator. Store the value of the second operand in the object specified by the first operand.
- bool operator!= (const iterator &_right) const

  This operator compare their operands for inequality.
- bool operator== (const iterator &_right) const

  This operator compares their operands for equality.
- iterator & operator++ ()

  Postfix increment operator.
- iterator operator++ (int)

  Prefix increment operator.
- _pairnm ∗ operator-> ()

  Member access operator. This operator dereferences the list pointer.
- _pairnm operator∗ ()

  The indirection operator dereferences the list pointer.

Friends

- class **NodeMap**

9.35.1 Detailed Description

This class provides a iterator that can read or modify any element in the list.

Definition at line 1628 of file bgapi2_genicam.hpp.

9.35.2 Member Function Documentation

9.35.2.1 operator"!=()

```
BGAPI2::NodeMap::iterator::operator!= (
            const iterator & _right ) const
```

This operator compare their operands for inequality.

Parameters

_right	The second operand.

Returns

bool The result of comparison.

9.35.2.2 operator∗()

```
BGAPI2::NodeMap::iterator::operator∗ ( )
```

The indirection operator dereferences the list pointer.

Returns

_pairnm The result is a pair structure with a 'first' and a 'second' value. The 'first' value is the string ID of the object. The 'second' value is the pointer to the BGAPI2 object.

9.35.2.3 operator++() [1/2]

```
BGAPI2::NodeMap::iterator::operator++ ( )
```

Postfix increment operator.

Returns

iterator The iterator of the next list element.

9.35.2.4 operator++() [2/2]

```
BGAPI2::NodeMap::iterator::operator++ (
            int   )
```

Prefix increment operator.

Returns

iterator The iterator of the next list element.

9.35.2.5 operator->()

```
BGAPI2::NodeMap::iterator::operator-> ( )
```

Member access operator. This operator dereferences the list pointer.

Returns

_pairnm The result is a pair structure with a 'first' and a 'second' value. The 'first' value is the string ID of the object. The 'second' value is the pointer to the BGAPI2 object.

9.35.2.6 operator=()

```
BGAPI2::NodeMap::iterator::operator= (
            const iterator & _iterator )
```

Assignment operator. Store the value of the second operand in the object specified by the first operand.

Parameters

_iterator	The second operand.

Returns

iterator The first operand.

9.35.2.7 operator==()

```
BGAPI2::NodeMap::iterator::operator== (
            const iterator & _right ) const
```

This operator compares their operands for equality.

Parameters

_right	The second operand.

Returns

bool The result of comparison.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.36 BGAPI2::DataStreamList::iterator Class Reference

This class provides an iterator that can read or modify any element in the list.

`#include <bgapi2_genicam.hpp>`

Public Member Functions

- iterator & operator= (const iterator &_iterator)

  Assignment operator. Store the value of the second operand in the object specified by the first operand.
- bo_bool operator!= (const iterator &_right) const

  This operator compare their operands for inequality.
- bo_bool operator== (const iterator &_right) const

  This operator compares their operands for equality.
- iterator & operator++ ()

  Postfix increment operator.
- iterator operator++ (int)

  Prefix increment operator.
- _pairds ∗ operator-> ()

  Member access operator. This operator dereferences the list pointer.
- _pairds ∗ operator∗ ()

  The indirection operator dereferences the list pointer.

Friends

- class **DataStreamList**

9.36.1 Detailed Description

This class provides an iterator that can read or modify any element in the list.

Definition at line 573 of file bgapi2_genicam.hpp.

9.36.2 Member Function Documentation

9.36.2.1 operator"!=()

```
BGAPI2::DataStreamList::iterator::operator!= (
            const iterator & _right ) const
```

This operator compare their operands for inequality.

Parameters

_right	The second operand.

Returns

bo_bool The result of comparison.

9.36.2.2 operator∗()

```
BGAPI2::DataStreamList::iterator::operator∗ ( )
```

The indirection operator dereferences the list pointer.

Returns

_pairds The result is a pair structure with a 'first' and a 'second' value. The 'first' value is the string ID of the object. The 'second' value is the pointer to the BGAPI2 object.

9.36.2.3 operator++() [1/2]

```
BGAPI2::DataStreamList::iterator::operator++ ( )
```

Postfix increment operator.

Returns

iterator The iterator of the next list element.

9.36.2.4 operator++() [2/2]

```
BGAPI2::DataStreamList::iterator::operator++ (
            int  )
```

Prefix increment operator.

Returns

iterator The iterator of the next list element.

9.36.2.5 operator->()

```
BGAPI2::DataStreamList::iterator::operator-> ( )
```

Member access operator. This operator dereferences the list pointer.

Returns

_pairds The result is a pair structure with a 'first' and a 'second' value. The 'first' value is the string ID of the object. The 'second' value is the pointer to the BGAPI2 object.

9.36.2.6 operator=()

```
BGAPI2::DataStreamList::iterator::operator= (
            const iterator & _iterator )
```

Assignment operator. Store the value of the second operand in the object specified by the first operand.

Parameters

_iterator	The second operand.

Returns

iterator The first operand.

9.36.2.7 operator==()

```
BGAPI2::DataStreamList::iterator::operator== (
            const iterator & _right ) const
```

This operator compares their operands for equality.

Parameters

_right	The second operand.

Returns

bo_bool The result of comparison.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.37 BGAPI2::DeviceList::iterator Class Reference

This class provides a iterator that can read or modify any element in the list.

`#include <bgapi2_genicam.hpp>`

Public Member Functions

- iterator & operator= (const iterator &_iterator)

  Assignment operator. Store the value of the second operand in the object specified by the first operand.
- bo_bool operator!= (const iterator &_right) const

  This operator compares their operands for inequality.
- bo_bool operator== (const iterator &_right) const

  This operator compares their operands for equality.
- iterator & operator++ ()

  Postfix increment operator.
- iterator operator++ (int)

  Prefix increment operator.
- _paird * operator-> ()

  Member access operator. This operator dereferences the list pointer.
- _paird * operator* ()

  The indirection operator dereferences the list pointer.

Friends

- class **DeviceList**

9.37.1 Detailed Description

This class provides a iterator that can read or modify any element in the list.

Definition at line 429 of file bgapi2_genicam.hpp.

9.37.2 Member Function Documentation

9.37.2.1 operator"!=()

```
BGAPI2::DeviceList::iterator::operator!= (
            const iterator & _right ) const
```

This operator compares their operands for inequality.

Parameters

_right	The second operand.

Returns

bo_bool The result of comparison.

9.37.2.2 operator∗()

```
BGAPI2::DeviceList::iterator::operator∗ ( )
```

The indirection operator dereferences the list pointer.

Returns

_paird The result is a pair structure with a 'first' and a 'second' value. The 'first' value is the string ID of the object. The 'second' value is the pointer to the BGAPI2 object.

9.37.2.3 operator++() [1/2]

```
BGAPI2::DeviceList::iterator::operator++ ( )
```

Postfix increment operator.

Returns

iterator The iterator of the next list element.

9.37.2.4 operator++() [2/2]

```
BGAPI2::DeviceList::iterator::operator++ (
            int   )
```

Prefix increment operator.

Returns

iterator The iterator of the next list element.

9.37.2.5 operator->()

```
BGAPI2::DeviceList::iterator::operator-> ( )
```

Member access operator. This operator dereferences the list pointer.

Returns

_paird The result is a pair structure with a 'first' and a 'second' value. The 'first' value is the string ID of the object. The 'second' value is the pointer to the BGAPI2 object.

9.37.2.6 operator=()

```
BGAPI2::DeviceList::iterator::operator= (
            const iterator & _iterator )
```

Assignment operator. Store the value of the second operand in the object specified by the first operand.

Parameters

_iterator	The second operand.

Returns

iterator The first operand.

9.37.2.7 operator==()

```
BGAPI2::DeviceList::iterator::operator== (
            const iterator & _right ) const
```

This operator compares their operands for equality.

Parameters

_right	The second operand.

Returns

bo_bool The result of comparison.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.38 BGAPI2::BufferList::iterator Class Reference

This class provides a iterator that can read or modify any element in the list.

`#include <bgapi2_genicam.hpp>`

Public Member Functions

- iterator & operator= (const iterator &_iterator)

  Assignment operator. Store the value of the second operand in the object specified by the first operand.
- bo_bool operator!= (const iterator &_right) const

  This operator compares their operands for inequality.
- bo_bool operator== (const iterator &_right) const

  This operator compares their operands for equality.
- iterator & operator++ ()

  Postfix increment operator.
- iterator operator++ (int)

  Prefix increment operator.
- _pairb ∗ operator-> ()

  Member access operator. This operator dereferences the list pointer.
- _pairb ∗ operator∗ ()

  The indirection operator dereferences the list pointer.

Friends

- class **BufferList**

9.38.1 Detailed Description

This class provides a iterator that can read or modify any element in the list.

Definition at line 826 of file bgapi2_genicam.hpp.

9.38.2 Member Function Documentation

9.38.2.1 operator"!=()

```
BGAPI2::BufferList::iterator::operator!= (
            const iterator & _right ) const
```

This operator compares their operands for inequality.

Parameters

_right	The second operand.

Returns

bo_bool The result of comparison.

9.38.2.2 operator∗()

```
BGAPI2::BufferList::iterator::operator∗ ( )
```

The indirection operator dereferences the list pointer.

Returns

_pairb The result is a pair structure with a 'first' and a 'second' value. The 'first' value is the string ID of the object. The 'second' value is the pointer to the BGAPI2 object.

9.38.2.3 operator++() [1/2]

```
BGAPI2::BufferList::iterator::operator++ ( )
```

Postfix increment operator.

Returns

iterator The iterator of the next list element.

9.38.2.4 operator++() [2/2]

```
BGAPI2::BufferList::iterator::operator++ (
            int   )
```

Prefix increment operator.

Returns

iterator The iterator of the next list element.

9.38.2.5 operator->()

```
BGAPI2::BufferList::iterator::operator-> ( )
```

Member access operator. This operator dereferences the list pointer.

Returns

_pairb The result is a pair structure with a 'first' and a 'second' value. The 'first' value is the string ID of the object. The 'second' value is the pointer to the BGAPI2 object.

9.38.2.6 operator=()

```
BGAPI2::BufferList::iterator::operator= (
            const iterator & _iterator )
```

Assignment operator. Store the value of the second operand in the object specified by the first operand.

Parameters

_iterator	The second operand.

Returns

iterator The first operand.

9.38.2.7 operator==()

```
BGAPI2::BufferList::iterator::operator== (
            const iterator & _right ) const
```

This operator compares their operands for equality.

Parameters

_right	The second operand.

Returns

bo_bool The result of comparison.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.39 BGAPI2::InterfaceList::iterator Class Reference

This class provides a iterator that can read or modify any element in the list.

`#include <bgapi2_genicam.hpp>`

Public Member Functions

- iterator & operator= (const iterator &_iterator)

  Assignment operator. Store the value of the second operand in the object specified by the first operand.
- bo_bool operator!= (const iterator &_right) const

  This operator compares their operands for inequality.
- bo_bool operator== (const iterator &_right) const

  This operator compares their operands for equality.
- iterator & operator++ ()

  Postfix increment operator.
- iterator operator++ (int)

  Prefix increment operator.
- _pairi * operator-> ()

  Member access operator. This operator dereferences the list pointer.
- _pairi * operator* ()

  The indirection operator dereferences the list pointer.

Friends

- class **InterfaceList**

9.39.1 Detailed Description

This class provides a iterator that can read or modify any element in the list.

Definition at line 281 of file bgapi2_genicam.hpp.

9.39.2 Member Function Documentation

9.39.2.1 operator"!=()

```
BGAPI2::InterfaceList::iterator::operator!= (
             const iterator & _right ) const
```

This operator compares their operands for inequality.

Parameters

_right	The second operand.

Returns

bo_bool The result of comparison.

9.39.2.2 operator*()

```
BGAPI2::InterfaceList::iterator::operator* ( )
```

The indirection operator dereferences the list pointer.

Returns

_pairi The result is a pair structure with a 'first' and a 'second' value. The 'first' value is the string ID of the object. The 'second' value is the pointer to the BGAPI2 object.

9.39.2.3 operator++() [1/2]

```
BGAPI2::InterfaceList::iterator::operator++ ( )
```

Postfix increment operator.

Returns

iterator The iterator of the next list element.

9.39.2.4 operator++() [2/2]

```
BGAPI2::InterfaceList::iterator::operator++ (
            int   )
```

Prefix increment operator.

Returns

iterator The iterator of the next list element.

9.39.2.5 operator->()

```
BGAPI2::InterfaceList::iterator::operator-> ( )
```

Member access operator. This operator dereferences the list pointer.

Returns

_pairi The result is a pair structure with a 'first' and a 'second' value. The 'first' value is the string ID of the object. The 'second' value is the pointer to the BGAPI2 object.

9.39.2.6 operator=()

```
BGAPI2::InterfaceList::iterator::operator= (
            const iterator & _iterator )
```

Assignment operator. Store the value of the second operand in the object specified by the first operand.

Parameters

_iterator	The second operand.

Returns

iterator The first operand.

9.39.2.7 operator==()

```
BGAPI2::InterfaceList::iterator::operator== (
            const iterator & _right ) const
```

This operator compares their operands for equality.

Parameters

_right	The second operand.

Returns

bo_bool The result of comparison.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.40 BGAPI2::SystemList::iterator Class Reference

This class provides an iterator that can read or modify any object of the list.

`#include <bgapi2_genicam.hpp>`

Public Member Functions

- iterator & operator= (const iterator &_iterator)

  Assignment operator. Store the value of the second operand in the object specified by the first operand.
- bo_bool operator!= (const iterator &_right) const

  This operator compares their operands for inequality.
- bo_bool operator== (const iterator &_right) const

  This operator compares their operands for equality.
- iterator & operator++ ()

  Postfix increment operator.
- iterator operator++ (int)

  Prefix increment operator.
- _pairs ∗ operator-> ()

  Member access operator. This operator dereferences the list pointer.
- _pairs ∗ operator∗ ()

  The indirection operator dereferences the list pointer.

Friends

- class **SystemList**

9.40.1 Detailed Description

This class provides an iterator that can read or modify any object of the list.

Definition at line 136 of file bgapi2_genicam.hpp.

9.40.2 Member Function Documentation

9.40.2.1 operator"!=()

```
BGAPI2::SystemList::iterator::operator!= (
            const iterator & _right ) const
```

This operator compares their operands for inequality.

Parameters

_right	The second operand.

Returns

bo_bool The result of comparison.

9.40.2.2 operator∗()

```
BGAPI2::SystemList::iterator::operator∗ ( )
```

The indirection operator dereferences the list pointer.

Returns

_pairs The result is a pair structure with a 'first' and a 'second' value. The 'first' value is the string ID of the object. The 'second' value is the pointer to the BGAPI2 object.

9.40.2.3 operator++() [1/2]

```
BGAPI2::SystemList::iterator::operator++ ( )
```

Postfix increment operator.

Returns

iterator The iterator of the next list element.

9.40.2.4 operator++() [2/2]

```
BGAPI2::SystemList::iterator::operator++ (
            int   )
```

Prefix increment operator.

Returns

iterator The iterator of the next list element.

9.40.2.5 operator->()

```
BGAPI2::SystemList::iterator::operator-> ( )
```

Member access operator. This operator dereferences the list pointer.

Returns

_pairs The result is a pair structure with a 'first' and a 'second' value. The 'first' value is the string ID of the object. The 'second' value is the pointer to the BGAPI2 object.

9.40.2.6 operator=()

```
BGAPI2::SystemList::iterator::operator= (
            const iterator & _iterator )
```

Assignment operator. Store the value of the second operand in the object specified by the first operand.

Parameters

_iterator	The second operand.

Returns

iterator The first operand.

9.40.2.7 operator==()

```
BGAPI2::SystemList::iterator::operator== (
            const iterator & _right ) const
```

This operator compares their operands for equality.

Parameters

_right	The second operand.

Returns

bo_bool The result of comparison.

The documentation for this class was generated from the following file:

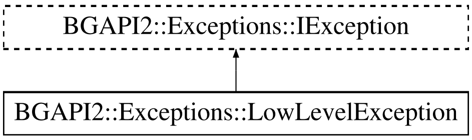
- bgapi2_genicam.hpp

9.41 BGAPI2::Exceptions::LowLevelException Class Reference

Exception thrown by deeper software layers like GenTL producer.

```
#include <bgapi2_genicam.hpp>
```

Inheritance diagram for BGAPI2::Exceptions::LowLevelException:


BGAPI2::Exceptions::IException
BGAPI2::Exceptions::LowLevelException

Additional Inherited Members

9.41.1 Detailed Description

Exception thrown by deeper software layers like GenTL producer.

Definition at line 3779 of file bgapi2_genicam.hpp.

The documentation for this class was generated from the following file:

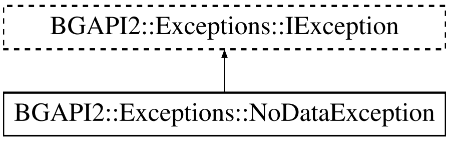
- bgapi2_genicam.hpp

9.42 BGAPI2::Exceptions::NoDataException Class Reference

An event contains no event data.

```
#include <bgapi2_genicam.hpp>
```

Inheritance diagram for BGAPI2::Exceptions::NoDataException:


BGAPI2::Exceptions::IException
BGAPI2::Exceptions::NoDataException

Additional Inherited Members

9.42.1 Detailed Description

An event contains no event data.

Definition at line 3724 of file bgapi2_genicam.hpp.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.43 BGAPI2::Node Class Reference

The class Node represent one feature from the provided node list based on the underlying XML definition.

```
#include <bgapi2_genicam.hpp>
```

Public Member Functions

- String GetInterface ()

  This function delivers the interface type of the Node object. Depending on the interface type the needed access functions are available. The available interface types are defined in header file bgapi2_def.h. See definitions BGAPI2_NODEINTERFACE_xxx.
- String GetToolTip ()

  This function delivers a short description text suitable for a ToolTip representation.
- String GetDescription ()

  This function delivers a description text of the Node object.
- String GetName ()

  This function delivers the name of the Node object.
- String GetDisplayName ()

  This function delivers a name of the node object suitable for displaying.
- String GetVisibility ()

  This function delivers a value representing the visibility of the Node object.
- String GetImposedAccessMode ()

  This function is deprecated. Please use IsReadable or IsWriteable instead.
- String GetCurrentAccessMode ()

  This function delivers a value that represents the current access to the Node object.
- bo_bool IsReadable ()

  This function indicates if a Node object is readable.
- bo_bool IsWriteable ()

  This function indicates if a Node object is writeable.
- String GetAlias ()

  This function delivers the name of a Node object specified as the alias. An alias describes the same feature in a different manner.
- String GetRepresentation ()

  This function delivers a value, which recommends the representation of the Node object on a GUI. Only available for the interface types 'IFloat' and 'IInteger'.
- String GetUnit ()

  This function delivers the physical unit of the Node object. Only available for the interface types 'IFloat' and 'IInteger'.
- bo_bool HasUnit ()

  This function delivers a flag that indicates whether the Node object has a physical unit. Only available for the interface types 'IFloat' and 'IInteger'.
- bo_bool GetImplemented ()

  This function delivers a flag that indicates whether the Node object is implemented.
- bo_bool GetAvailable ()

  This function delivers a flag that indicates whether the Node object is available.
- bo_bool GetLocked ()

  This function is deprecated. Please use isReadable or isWriteable instead.
- bo_int64 GetEventID ()

  This function delivers an ID of a asynchronous event to which the Node object is linked.
- String GetExtension ()

  This function delivers user specific data from the XML definition of this Node object.
- NodeMap * GetEnumNodeList ()

  This function delivers a NodeMap of available enumeration entries.
- String GetValue ()

  This function delivers the value of the Node object in string format.
- void SetValue (String Value)

  This function writes a value in string format to the Node object.

- bo_int64 GetInt ()

  This function delivers the current value of the Node object as Integer.
- void SetInt (bo_int64 value)

  This function writes a integer value to the Node object.
- bo_int64 GetIntMin ()

  This function delivers the minimal allowed value of the Node object as Integer.
- bo_int64 GetIntMax ()

  This function delivers the maximal allowed value of the Node object as Integer.
- bo_int64 GetIntInc ()

  This function delivers the allowed step size for the value of the Node object as Integer.
- bo_double GetDouble ()

  This function delivers the current value of the Node object as floating-point number.
- void SetDouble (bo_double value)

  This function writes a floating-point value to the Node object.
- bo_double GetDoubleMin ()

  This function delivers the minimal allowed value of the Node object as floating-point number.
- bo_double GetDoubleMax ()

  This function delivers the maximal allowed value of the Node object as floating-point number.
- bo_bool HasInc ()

  This function delivers a flag that indicates whether the Node object has an increment. Only available for the interface types 'IFloat' and 'IInteger'.
- bo_double GetDoubleInc ()

  This function delivers the allowed step size for the value of the Node object as Float.
- bo_uint64 GetDoublePrecision ()

  This function delivers the precision for the corresponding double.
- bo_int64 GetMaxStringLength ()

  This function delivers the length of the string.
- String GetString ()

  This function is deprecated. Please use GetValue instead.
- void SetString (String value)

  This function is deprecated. Please use SetValue instead.
- void Execute ()

  This function executes the command of the Node object.
- bo_bool IsDone ()

  This function delivers a flag that indicates whether the command of the Node object has been executed.
- bo_bool GetBool ()

  This function delivers the current value of the Node object as boolean value.
- void SetBool (bo_bool value)

  This function writes a boolean value to the Node object.
- NodeMap ∗ GetNodeTree ()

  This function provides the subordinate features of the Node object as tree structure in form of a NodeMap pointer. For a detailed description of the tree structure representation refer to the class description of NodeMap.
- NodeMap ∗ GetNodeList ()

  This function provides the subordinate features of the Node object as unstructured list in form of a NodeMap pointer. For a detailed description of the unstructured list representation refer to the class description of NodeMap.
- bo_bool IsSelector ()

  This function delivers a flag that indicates whether the Node object is a selector.
- NodeMap ∗ GetSelectedNodeList ()

  This function delivers a list of features that depend on this selector Node.

- bo_uint64 getLength ()

  This function delivers the length in bytes of the memory pointed to by the Node object.
- bo_uint64 getAddress ()

  This function delivers the address of the memory pointed to by the Node object.
- void get (void ∗pBuffer, bo_uint64 len)

  This function reads the memory pointed to by the Node object and writes it into the provided buffer.
- void set (void ∗pBuffer, bo_uint64 len)

  This function writes the memory pointed to by the Node object.

9.43.1 Detailed Description

The class Node represent one feature from the provided node list based on the underlying XML definition.

Definition at line 1016 of file bgapi2_genicam.hpp.

9.43.2 Member Function Documentation

9.43.2.1 Execute()

```
BGAPI2::Node::Execute ( )
```

This function executes the command of the Node object.

Only valid for the interface type 'ICommand'.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. Possible reasons for this exception are the loss of connection to the device and the wrong access mode.
Exceptions::NotImplementedException	This feature is not implemented.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.2 get()

```
BGAPI2::Node::get (
            void ∗ pBuffer,
            bo_uint64 len )
```

This function reads the memory pointed to by the Node object and writes it into the provided buffer.

Only valid for the interface type 'IRegister'.

Parameters

pBuffer	The destination buffer into which the read data is copied.
len	The size of the destination buffer. The function Node::getLength delivers the necessary size.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. Possible reasons for this exception are the loss of connection to the device and the wrong access mode, e.g. "WO".
Exceptions::NotImplementedException	This feature is not implemented.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.3 getAddress()

```
BGAPI2::Node::getAddress ( )
```

This function delivers the address of the memory pointed to by the Node object.

Only valid for the interface type 'IRegister'.

Returns

bo_uint64 The address of the memory pointed to by the Node object.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. A possible reason for this exception is the loss of connection to the device.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.4 GetAlias()

```
BGAPI2::Node::GetAlias ( )
```

This function delivers the name of a Node object specified as the alias. An alias describes the same feature in a different manner.

Returns

String The name of the alias Node object.

Exceptions

Exceptions::NotAvailableException	No alias specified.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.5 GetAvailable()

```
BGAPI2::Node::GetAvailable ( )
```

This function delivers a flag that indicates whether the Node object is available.

This status is equivalent to the access mode 'NA'. See functions Node::GetImposedAccessMode and Node::GetCurrentAccessMode.

Returns

bo_bool The flag that indicates whether the Node object is available.

Exceptions

Exceptions::AccessDeniedException	Error when accessing this function. A possible reason for this exception is the loss of connection to the device.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.6 GetBool()

```
BGAPI2::Node::GetBool ( )
```

This function delivers the current value of the Node object as boolean value.

Only valid for the interface type 'IBoolean'.

Returns

bo_bool The current value of the Node object as boolean value.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. Possible reasons for this exception are the loss of connection to the device and the wrong access mode, e.g. "WO".
Exceptions::NotImplementedException	This feature is not implemented.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.7 GetCurrentAccessMode()

```
BGAPI2::Node::GetCurrentAccessMode ( )
```

This function delivers a value that represents the current access to the Node object.

This value is affected by the change of dependent Node objects. In such a case, the current access mode is different and limited (i.e. less accessible) for general access mode. See also function Node::GetImposedAccessMode(). The available access modes are defined in header file bgapi2_def.h. See definitions BGAPI2_NODEACCESS_xxx.

Returns

String The current access to the Node object.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.8 GetDescription()

```
BGAPI2::Node::GetDescription ( )
```

This function delivers a description text of the Node object.

Returns

String The description text of the Node object.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.9 GetDisplayName()

```
BGAPI2::Node::GetDisplayName ( )
```

This function delivers a name of the node object suitable for displaying.

Returns

String The display name of the Node object.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.10 GetDouble()

```
BGAPI2::Node::GetDouble ( )
```

This function delivers the current value of the Node object as floating-point number.

Valid for the interface types 'IFloat', 'IInteger', 'IEnumeration' and 'IBoolean'.

Returns

bo_double The current value as floating-point number.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. Possible reasons for this exception are the loss of connection to the device and the wrong access mode, e.g. "WO".
Exceptions::NotImplementedException	This feature is not implemented.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.11 GetDoubleInc()

```
BGAPI2::Node::GetDoubleInc ( )
```

This function delivers the allowed step size for the value of the Node object as Float.

Valid for the interface types 'IFloat' and 'IInteger'.

Returns

bo_double The allowed step size for the value of the Node object as Float.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::InvalidParameterException	The Node object has no (valid) increment value.
Exceptions::AccessDeniedException	Error when accessing this function. A possible reason for this exception is the loss of connection to the device.
Exceptions::NotImplementedException	This feature is not implemented.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.12 GetDoubleMax()

```
BGAPI2::Node::GetDoubleMax ( )
```

This function delivers the maximal allowed value of the Node object as floating-point number.

Valid for the interface types 'IFloat' and 'IInteger'.

Returns

bo_double The maximal allowed value as floating-point number.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. A possible reason for this exception is the loss of connection to the device.
Exceptions::NotImplementedException	This feature is not implemented.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.13 GetDoubleMin()

```
BGAPI2::Node::GetDoubleMin ( )
```

This function delivers the minimal allowed value of the Node object as floating-point number.

Valid for the interface types 'IFloat' and 'IInteger'.

Returns

bo_double The minimal allowed value as floating-point number.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. A possible reason for this exception is the loss of connection to the device.
Exceptions::NotImplementedException	This feature is not implemented.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.14 GetDoublePrecision()

```
BGAPI2::Node::GetDoublePrecision ( )
```

This function delivers the precision for the corresponding double.

Only valid for the interface type 'IFloat'.

Returns

bo_uint32 non-negative number for the precision the double is displayed

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. A possible reason for this exception is the loss of connection to the device.
Exceptions::NotImplementedException	This feature is not implemented.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.15 GetEnumNodeList()

```
BGAPI2::Node::GetEnumNodeList ( )
```

This function delivers a NodeMap of available enumeration entries.

Only valid for interface type 'IEnumeration'.

Returns

NodeMap∗ The list of all available enumeration entries of the Node object.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. A possible reason for this exception is the loss of connection to the device.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.16 GetEventID()

```
BGAPI2::Node::GetEventID ( )
```

This function delivers an ID of a asynchronous event to which the Node object is linked.

Returns

bo_int64 The ID of a asynchronous event.

Exceptions

Exceptions::NotAvailableException	The EventID is not specified in the XML description of this Node object.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.17 GetExtension()

```
BGAPI2::Node::GetExtension ( )
```

This function delivers user specific data from the XML definition of this Node object.

Returns

String vendor specific data.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.18 GetImplemented()

```
BGAPI2::Node::GetImplemented ( )
```

This function delivers a flag that indicates whether the Node object is implemented.

Returns

bo_bool The flag that indicates whether the Node object is implemented.

Exceptions

Exceptions::AccessDeniedException	Error when accessing this function. A possible reason for this exception is the loss of connection to the device.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.19 GetImposedAccessMode()

```
BGAPI2::Node::GetImposedAccessMode ( )
```

This function is deprecated. Please use IsReadable or IsWriteable instead.

This function delivers a value that specifies the general access to the Node object. This value is not affected by other Node objects. See also the function Node::GetCurrentAccessMode(). The available access modes are defined in header file bgapi2_def.h. See definitions BGAPI2_NODEACCESS_xxx.

Returns

String The general access to the Node object.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.20 GetInt()

```
BGAPI2::Node::GetInt ( )
```

This function delivers the current value of the Node object as Integer.

Valid for the interface types 'IFloat', 'IInteger', 'IEnumeration' and 'IBoolean'.

Returns

bo_int64 The current value as integer.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. Possible reasons for this exception are the loss of connection to the device and the wrong access mode, e.g. "WO".
Exceptions::NotImplementedException	This feature is not implemented.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.21 GetInterface()

```
BGAPI2::Node::GetInterface ( )
```

This function delivers the interface type of the Node object. Depending on the interface type the needed access functions are available. The available interface types are defined in header file bgapi2_def.h. See definitions BGAPI2_NODEINTERFACE_xxx.

Returns

String The interface type of the Node object.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.22 GetIntInc()

```
BGAPI2::Node::GetIntInc ( )
```

This function delivers the allowed step size for the value of the Node object as Integer.

Valid for the interface types 'IInteger' and 'IFloat'.

Returns

bo_int64 The allowed step size for the value of the Node object as integer.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. A possible reason for this exception is the loss of connection to the device.
Exceptions::NotImplementedException	This feature is not implemented.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.23 GetIntMax()

```
BGAPI2::Node::GetIntMax ( )
```

This function delivers the maximal allowed value of the Node object as Integer.

Valid for the interface types 'IInteger' and 'IFloat'.

Returns

bo_int64 The maximal allowed value as integer.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. A possible reason for this exception is the loss of connection to the device.
Exceptions::NotImplementedException	This feature is not implemented.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.24 GetIntMin()

```
BGAPI2::Node::GetIntMin ( )
```

This function delivers the minimal allowed value of the Node object as Integer.

Valid for the interface types 'IInteger' and 'IFloat'.

Returns

bo_int64 The minimal allowed value as integer.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. A possible reason for this exception is the loss of connection to the device.
Exceptions::NotImplementedException	This feature is not implemented.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.25 getLength()

```
BGAPI2::Node::getLength ( )
```

This function delivers the length in bytes of the memory pointed to by the Node object.

Only valid for the interface type 'IRegister'.

Returns

bo_uint64 The length in bytes of the memory pointed to by the Node object.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. A possible reason for this exception is the loss of connection to the device.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.26 GetLocked()

```
BGAPI2::Node::GetLocked ( )
```

This function is deprecated. Please use isReadable or isWriteable instead.

This function delivers a flag that indicates whether the Node object is locked for writing. This status is equivalent to the access mode 'RO'. See functions Node::GetImposedAccessMode and Node::GetCurrentAccessMode.

Returns

bo_bool The flag that indicates whether the Node object is locked for writing.

Exceptions

Exceptions::AccessDeniedException	Error when accessing this function. A possible reason for this exception is the loss of connection to the device.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.27 GetMaxStringLength()

```
BGAPI2::Node::GetMaxStringLength ( )
```

This function delivers the length of the string.

Only valid for the interface type 'IString'.

Returns

bo_int64 The length of the string.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. A possible reason for this exception is the loss of connection to the device.
Exceptions::NotImplementedException	This feature is not implemented.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.28 GetName()

```
BGAPI2::Node::GetName ( )
```

This function delivers the name of the Node object.

Returns

String The name of the Node object.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.29 GetNodeList()

```
BGAPI2::Node::GetNodeList ( )
```

This function provides the subordinate features of the Node object as unstructured list in form of a NodeMap pointer. For a detailed description of the unstructured list representation refer to the class description of NodeMap.

Only valid for the interface type 'ICategory'.

Returns

NodeMap∗ The list of all features of this Node object.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.30 GetNodeTree()

```
BGAPI2::Node::GetNodeTree ( )
```

This function provides the subordinate features of the Node object as tree structure in form of a NodeMap pointer. For a detailed description of the tree structure representation refer to the class description of NodeMap.

Only valid for the interface type 'ICategory'.

Returns

NodeMap∗ The tree structure of all features of this Node object.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.31 GetRepresentation()

```
BGAPI2::Node::GetRepresentation ( )
```

This function delivers a value, which recommends the representation of the Node object on a GUI. Only available for the interface types 'IFloat' and 'IInteger'.

The available display options are defined in the header file bgapi2_def.h. See the definitions BGAPI2↩ _NODEREPRESENTATION_xxx.

Returns

String The recommended value for the representation of the Node object on a GUI.

Exceptions

Exceptions::NotAvailableException	This exception is thrown when the interface type is not 'IFloat' and not 'IInteger'.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.32 GetSelectedNodeList()

```
BGAPI2::Node::GetSelectedNodeList ( )
```

This function delivers a list of features that depend on this selector Node.

Valid for the interface types 'IInteger' and 'IEnumeration'.

Returns

NodeMap∗ The list of all features that depend on this selector Node.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.33 GetString()

```
BGAPI2::Node::GetString ( )
```

This function is deprecated. Please use GetValue instead.

This function delivers the current value of the Node object as String. Valid for the interface types 'IFloat', 'IInteger', 'IEnumeration', IString', 'IBoolean' and 'ICommand'.

Returns

String The current value of the Node object as String.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. Possible reasons for this exception are the loss of connection to the device and the wrong access mode, e.g. "WO".
Exceptions::NotImplementedException	This feature is not implemented.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.34 GetToolTip()

```
BGAPI2::Node::GetToolTip ( )
```

This function delivers a short description text suitable for a ToolTip representation.

Returns

String The short description text of the Node object.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.35 GetUnit()

```
BGAPI2::Node::GetUnit ( )
```

This function delivers the physical unit of the Node object. Only available for the interface types 'IFloat' and 'IInteger'.

Returns

String The physical unit of the Node object.

Exceptions

Exceptions::NotAvailableException	This exception is thrown when the interface type is not 'IFloat' and not 'IInteger'.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.36 GetValue()

```
BGAPI2::Node::GetValue ( )
```

This function delivers the value of the Node object in string format.

Valid for the interface types 'IFloat', 'IInteger', 'IEnumeration', IString', 'IBoolean' and 'ICommand'.

Returns

String The value of the Node object in string format.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. A possible reason for this exception is the loss of connection to the device.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.37 GetVisibility()

```
BGAPI2::Node::GetVisibility ( )
```

This function delivers a value representing the visibility of the Node object.

Returns

String The visibility of the Node object.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.38 HasInc()

```
BGAPI2::Node::HasInc ( )
```

This function delivers a flag that indicates whether the Node object has an increment. Only available for the interface types 'IFloat' and 'IInteger'.

Returns

bo_bool The flag that indicates whether the Node object has an increment.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.39 HasUnit()

```
BGAPI2::Node::HasUnit ( )
```

This function delivers a flag that indicates whether the Node object has a physical unit. Only available for the interface types 'IFloat' and 'IInteger'.

Returns

bo_bool The flag that indicates whether the Node object has a physical unit.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.40 IsDone()

```
BGAPI2::Node::IsDone ( )
```

This function delivers a flag that indicates whether the command of the Node object has been executed.

Only valid for the interface type 'ICommand'.

Returns

bo_bool The flag that indicates whether the command of the Node object has been executed.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.41 IsReadable()

```
BGAPI2::Node::IsReadable ( )
```

This function indicates if a Node object is readable.

Returns

true when the Node object is readable, otherwise false.

Exceptions

Exceptions::AccessDeniedException	Error when accessing this function. A possible reason for this exception is the loss of connection to the device.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.42 IsSelector()

```
BGAPI2::Node::IsSelector ( )
```

This function delivers a flag that indicates whether the Node object is a selector.

A selector is a posibility to define feature dependencies. The current value of a selector node has an impact on the value of another Node object. Valid for the interface types 'IInteger' and 'IEnumeration'.

Returns

bo_bool The flag that indicates whether the Node object is a selector.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.43 IsWriteable()

```
BGAPI2::Node::IsWriteable ( )
```

This function indicates if a Node object is writeable.

Returns

true when the Node object is writable, otherwise false.

Exceptions

Exceptions::AccessDeniedException	Error when accessing this function. A possible reason for this exception is the loss of connection to the device.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.44 set()

```
BGAPI2::Node::set (
            void * pBuffer,
            bo_uint64 len )
```

This function writes the memory pointed to by the Node object.

Only valid for the interface type 'IRegister'.

Parameters

pBuffer	The buffer for the data to be written.
len	The size of the buffer to the data to be written. The function Node::getLength delivers the necessary size.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. A possible reason for this exception is the loss of connection to the device.
Exceptions::NotImplementedException	This feature is not implemented.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.45 SetBool()

```
BGAPI2::Node::SetBool (
            bo_bool value )
```

This function writes a boolean value to the Node object.

Only valid for the interface type 'IBoolean'.

Parameters

value	The boolean value to be written.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. Possible reasons for this exception are the loss of connection to the device and the wrong access mode.
Exceptions::NotImplementedException	This feature is not implemented.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.46 SetDouble()

```
BGAPI2::Node::SetDouble (
              bo_double value )
```

This function writes a floating-point value to the Node object.

Valid for the interface types 'IFloat', 'IInteger', 'IEnumeration' and 'IBoolean'.

Parameters

value	The floating-point value to be written.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. Possible reasons for this exception are the loss of connection to the device and the wrong access mode.
Exceptions::NotImplementedException	This feature is not implemented.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.47 SetInt()

```
BGAPI2::Node::SetInt (
              bo_int64 value )
```

This function writes a integer value to the Node object.

Valid for the interface types 'IFloat', 'IInteger', 'IEnumeration' and 'IBoolean'.

Parameters

value	The integer value to be written.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. Possible reasons for this exception are the loss of connection to the device and the wrong access mode.
Exceptions::NotImplementedException	This feature is not implemented.

Exceptions

Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.48 SetString()

```
BGAPI2::Node::SetString (
            String value )
```

This function is deprecated. Please use SetValue instead.

This function writes a string value to the Node object. Valid for the interface types 'IFloat', 'IInteger', 'IEnumeration', IString', 'IBoolean' and 'ICommand'.

Parameters

value	The string value to be written.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. Possible reasons for this exception are the loss of connection to the device and the wrong access mode.
Exceptions::NotImplementedException	This feature is not implemented.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.43.2.49 SetValue()

```
BGAPI2::Node::SetValue (
            String Value )
```

This function writes a value in string format to the Node object.

Valid for the interface types 'IFloat', 'IInteger', 'IEnumeration', IString', 'IBoolean' and 'ICommand'.

Parameters

Value	The value to be written in string format.

Exceptions

Exceptions::NotAvailableException	The Node object has the wrong interface type.
Exceptions::AccessDeniedException	Error when accessing this function. A possible reason for this exception is the loss of connection to the device.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.
Exceptions::ObjectInvalidException	The calling object is not valid.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.44 BGAPI2::NodeMap Class Reference

The class NodeMap represents a collection of Node objects based on the underlying XML definition file. This class supports two representation forms, a unstructured list and a tree structure.

`#include <bgapi2_genicam.hpp>`

Classes

- class iterator

  This class provides a iterator that can read or modify any element in the list.

Public Member Functions

- bo_uint64 size ()

  This function delivers the number of Node objects in the Node list.
- bo_bool GetNodePresent (String name)

  This function delivers a flag that indicates whether the NodeMap contains a Node with the specified name.
- Node ∗ operator[] (const String &val)

  This operator allows the direct access to an object of the Node list.
- Node ∗ GetNode (String name)

  This function delivers a certain object of the Node list.
- bo_uint64 GetNodeCount ()

  This function delivers the number of Node objects in the Node list.
- Node ∗ GetNodeByIndex (bo_uint64 iIndex)

  This function delivers a pointer to a specified Node object.
- iterator begin ()

  This function delivers an iterator on the top of the Node list.
- iterator end ()

  This function delivers an iterator at the end of the Node list.
- iterator find (const String &_keyval)

  This function delivers an iterator on an object to be found. The object is not found, this functions delivers an end-iterator.

Friends

- class **Node**

9.44.1 Detailed Description

The class NodeMap represents a collection of Node objects based on the underlying XML definition file. This class supports two representation forms, a unstructured list and a tree structure.

The tree structure is a hierarchical representation. Special kinds of nodes can have subordinate nodes. These nodes are from interface type 'ICategory' (see function Node::GetInterface). The access to its subordinate nodes occurs by calling NodeMap::GetNodeTree. The unstructured list representation is linear. Use the NodeMap::iterator class and the NodeMap::begin and NodeMap::end functions for iterating through the list. All nodes from type 'ICategory' are removed and all its subordinate nodes are moved into the single list. The following functions use the unstructured list representation: INode::GetNodeList, Device::GetRemoteNodeList, Node::GetNodeList, Node::GetEnumNodeList, Node::GetSelectedNodeList The following functions use the tree structure representation: INode::GetNodeTree, Device::GetRemoteNodeTree, Node::GetNodeTree

Definition at line 1573 of file bgapi2_genicam.hpp.

9.44.2 Member Function Documentation

9.44.2.1 begin()

```
BGAPI2::NodeMap::begin ( )
```

This function delivers an iterator on the top of the Node list.

Returns

iterator The iterator on the top of the Node list.

9.44.2.2 end()

```
BGAPI2::NodeMap::end ( )
```

This function delivers an iterator at the end of the Node list.

Returns

iterator The iterator at the end of the Node list.

9.44.2.3 find()

```
BGAPI2::NodeMap::find (
             const String & _keyval )
```

This function delivers an iterator on an object to be found. The object is not found, this functions delivers an end-iterator.

Parameters

_keyval	The ID to the object to be found.

Returns

iterator The iterator to the found object.

9.44.2.4 GetNode()

```
BGAPI2::NodeMap::GetNode (
            String name )
```

This function delivers a certain object of the Node list.

Parameters

name	For this name, the associated Node object is delivered.

Returns

Node∗ The requested Node object.

Exceptions

Exceptions::InvalidParameterException	No object in the Node list has the passed name.

9.44.2.5 GetNodeByIndex()

```
BGAPI2::NodeMap::GetNodeByIndex (
            bo_uint64 iIndex )
```

This function delivers a pointer to a specified Node object.

Parameters

iIndex	The index of the Node object.

Returns

Node∗ The requested Node object.

Exceptions

Exceptions::InvalidParameterException	The passed index is invalid.

9.44.2.6 GetNodeCount()

```
BGAPI2::NodeMap::GetNodeCount ( )
```

This function delivers the number of Node objects in the Node list.

Returns

bo_uint64 The number of Node objects in the Node list.

9.44.2.7 GetNodePresent()

```
BGAPI2::NodeMap::GetNodePresent (
            String name )
```

This function delivers a flag that indicates whether the NodeMap contains a Node with the specified name.

Parameters

name	The Node name to search for.

Returns

bo_bool The flag which indicates whether the NodeMap contains a Node with the specified name.

9.44.2.8 operator[]()

```
BGAPI2::NodeMap::operator[ ] (
            const String & val )
```

This operator allows the direct access to an object of the Node list.

Parameters

val	For this name, the associated Node object is delivered.

Returns

Node∗ The requested Node object.

Exceptions

Exceptions::InvalidParameterException	No object in the Node list has the passed name.

9.44.2.9 size()

```
BGAPI2::NodeMap::size ( )
```

This function delivers the number of Node objects in the Node list.

Returns

bo_uint64 The number of Node objects in the Node list.

The documentation for this class was generated from the following file:

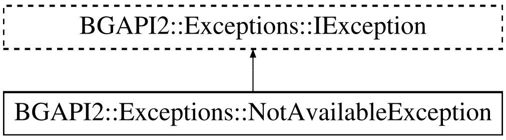
- bgapi2_genicam.hpp

9.45 BGAPI2::Exceptions::NotAvailableException Class Reference

The requested resource or information is not available at a given time in a current state.

`#include <bgapi2_genicam.hpp>`

Inheritance diagram for BGAPI2::Exceptions::NotAvailableException:


BGAPI2::Exceptions::IException
BGAPI2::Exceptions::NotAvailableException

Additional Inherited Members

9.45.1 Detailed Description

The requested resource or information is not available at a given time in a current state.

Definition at line 3761 of file bgapi2_genicam.hpp.

The documentation for this class was generated from the following file:

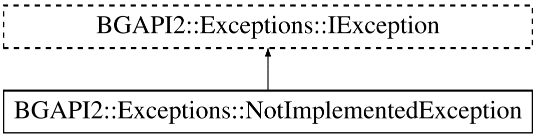
- bgapi2_genicam.hpp

9.46 BGAPI2::Exceptions::NotImplementedException Class Reference

The requested function/feature is not implemented.

`#include <bgapi2_genicam.hpp>`

Inheritance diagram for BGAPI2::Exceptions::NotImplementedException:


BGAPI2::Exceptions::IException
BGAPI2::Exceptions::NotImplementedException

Additional Inherited Members

9.46.1 Detailed Description

The requested function/feature is not implemented.

Definition at line 3688 of file bgapi2_genicam.hpp.

The documentation for this class was generated from the following file:

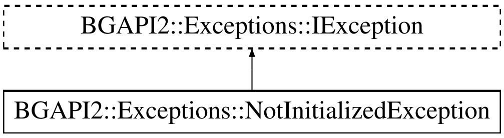
- bgapi2_genicam.hpp

9.47 BGAPI2::Exceptions::NotInitializedException Class Reference

The requested object is not initialized/opened.

`#include <bgapi2_genicam.hpp>`

Inheritance diagram for BGAPI2::Exceptions::NotInitializedException:


BGAPI2::Exceptions::IException
BGAPI2::Exceptions::NotInitializedException

Additional Inherited Members

9.47.1 Detailed Description

The requested object is not initialized/opened.

Definition at line 3679 of file bgapi2_genicam.hpp.

The documentation for this class was generated from the following file:

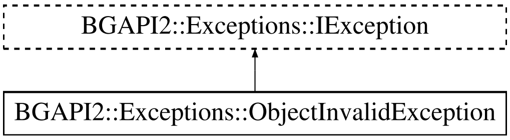
- bgapi2_genicam.hpp

9.48 BGAPI2::Exceptions::ObjectInvalidException Class Reference

The referenced object is not a valid object of BGAPI2.

`#include <bgapi2_genicam.hpp>`

Inheritance diagram for BGAPI2::Exceptions::ObjectInvalidException:


BGAPI2::Exceptions::IException
BGAPI2::Exceptions::ObjectInvalidException

Additional Inherited Members

9.48.1 Detailed Description

The referenced object is not a valid object of BGAPI2.

Definition at line 3770 of file bgapi2_genicam.hpp.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.49 BGAPI2::Events::PnPEvent Class Reference

The class PnPEvent represented a plug'n play event and provides access to the event information.

`#include <bgapi2_genicam.hpp>`

Public Member Functions

- PnPEvent ()

  Constructor for creating an PnPEvent object.
- ∼PnPEvent ()

  Destructor for deleting an PnPEvent object.
- String GetSerialNumber ()

  This function delivers the serial number of the Device object which was added/removed from the Interface object.
- PnPType GetPnPType ()

  This function delivers the kind of the PnPEvent. The two several kinds of PnPEvent are defined in enumeration Events::PnPType.
- String GetId ()

  This function delivers the event ID of the PnPEvent.

Friends

- class **InterfaceEventControl**

9.49.1 Detailed Description

The class PnPEvent represented a plug'n play event and provides access to the event information.

Definition at line 1831 of file bgapi2_genicam.hpp.

9.49.2 Member Function Documentation

9.49.2.1 GetId()

```
BGAPI2::Events::PnPEvent::GetId ( )
```

This function delivers the event ID of the PnPEvent.

Returns

String The ID of the PnPevent in string format.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::NotAvailableException	The ID of the PnPEvent was not included.

9.49.2.2 GetPnPType()

```
BGAPI2::Events::PnPEvent::GetPnPType ( )
```

This function delivers the kind of the PnPEvent. The two several kinds of PnPEvent are defined in enumeration Events::PnPType.

Returns

PnPType The kind of the PnPEvent.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.49.2.3 GetSerialNumber()

```
BGAPI2::Events::PnPEvent::GetSerialNumber ( )
```

This function delivers the serial number of the Device object which was added/removed from the Interface object.

Returns

String The serial number of the Device object which was add/removed from the Interface object.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::NotAvailableException	The serial number was not included in the event.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.50 BGAPI2::Polarizer Class Reference

Provides functionality to calculate several different formats out of the raw polarized camera data.

```
#include <bgapi2_genicam.hpp>
```

Classes

- class formatlist
  This class provides a iterator that can read or modify any element in the list.

Public Types

- enum Formats {
  AOP, DOLP, ADOLP, Intensity,
  Pol0Deg, Pol45Deg, Pol90Deg, Pol135Deg,
  ReflectionMin, ReflectionMax }
  An enumeration containing the string representation of the possible polarization formats.

Public Member Functions

- Polarizer ()
  Parameterless constructor for creating of an Polarizer object.
- ∼Polarizer ()
  Destructor to destroy an Polarizer object.
- void Initialize (BGAPI2::Buffer ∗buffer)
  Initialize the Polarizer and provide the Buffer with the raw polarized data to calculate the.
- void ReadCalibrationData (BGAPI2::Device ∗device)
  Get the calibration data and angle offset from the camera.
- void EnableInterpolation (bo_bool interpolate)
  Keep output image the same size as the input buffer. The default is disabled.
- void Enable (Formats format, bo_bool enable)
  Each component to be calculated must be enabled first.
- void Get (Formats format, BGAPI2::Image ∗image)
  Get the calculated component (AOP, Pol45deg etc.) from the buffer provided through the Initialize()
- String GetFormatString (Formats format)
  Get the string of the polarization format.
- void SetMaxThreads (bo_uint number)
  Set the number of threads the Polarizer can use for calculations.

9.50.1 Detailed Description

Provides functionality to calculate several different formats out of the raw polarized camera data.

Definition at line 3426 of file bgapi2_genicam.hpp.

9.50.2 Member Enumeration Documentation

9.50.2.1 Formats

`enum BGAPI2::Polarizer::Formats`

An enumeration containing the string representation of the possible polarization formats.

Enumerator

AOP	Enum value AOP - Angle of polarization
DOLP	Enum value DOLP - Degree of linear polarization
ADOLP	Enum value ADOLP - Angle and degree of linear polarization
Intensity	Enum value Intensity - Gray scale image
Pol0Deg	Enum value Pol0Deg - The polarization information 0 degrees
Pol45Deg	Enum value Pol45Deg - The polarization information 45 degrees
Pol90Deg	Enum value Pol90Deg - The polarization information 90 degrees
Pol135Deg	Enum value Pol135Deg - The polarization information 135 degrees
ReflectionMin	Enum value ReflectionMin - Image with suppressed reflections
ReflectionMax	Enum value ReflectionMax - Image with enhanced reflections

Definition at line 3475 of file bgapi2_genicam.hpp.

9.50.3 Member Function Documentation

9.50.3.1 Enable()

```
void BGAPI2::Polarizer::Enable (
            Formats format,
            bo_bool enable )
```

Each component to be calculated must be enabled first.

To speed up the calculation of the different components it is necessary to enable them first. This allows for the calculation to re-use and combine some of the necessary calculations.

Parameters

in	*format*	The format to enable or disable
in	*enable*	Set to true to enable or false to disable

See also

BGAPI2::Polarizer::Formats

9.50.3.2 EnableInterpolation()

```
void BGAPI2::Polarizer::EnableInterpolation (
            bo_bool interpolate )
```

Keep output image the same size as the input buffer. The default is disabled.

If enabled, the calculated images will be interpolated to have the same size as the raw image buffer provided.

Parameters

in	*bo_bool*	interpolate If set to true the result images will be interpolated.

9.50.3.3 Get()

```
void BGAPI2::Polarizer::Get (
            Formats format,
            BGAPI2::Image * image )
```

Get the calculated component (AOP, Pol45deg etc.) from the buffer provided through the Initialize()

method.

For performance reasons when handling more than one component, a component must first be enabled via the Polarizer::Enable() method.

Parameters

in	*format*	The format to enable or disable

9.50.3.4 GetFormatString()

```
String BGAPI2::Polarizer::GetFormatString (
            Formats format )
```

Get the string of the polarization format.

Parameters

in	*format*	The polarization format String The string of the polarization format.

See also

BGAPI2::Polarizer::Formats

Exceptions

Exceptions::InvalidParameterException	Invalid parameter

9.50.3.5 Initialize()

```
void BGAPI2::Polarizer::Initialize (
            BGAPI2::Buffer * buffer )
```

Initialize the Polarizer and provide the Buffer with the raw polarized data to calculate the.

different polarized formats from.

Parameters

in	*buffer*	A valid buffer with polarized data acquired by a Baumer camera.

Exceptions

BGAPI2::InvalidParameterException	Invalid Buffer.

9.50.3.6 ReadCalibrationData()

```
void BGAPI2::Polarizer::ReadCalibrationData (
            BGAPI2::Device * device )
```

Get the calibration data and angle offset from the camera.

Reads the calibration matrix and the configured polarization angle offset from the camera device to enhance the calculation of different polarization formats.

Parameters

in	*device*	The BGAPI2::Device∗, a pointer to the polarization camera (opened, must be able to read features from the camera)

Exceptions

Exceptions::InvalidParameterException	The passed BGAPI device is not valid.
Exceptions::NotInitializedException	The BGAPI device is not open.

9.50.3.7 SetMaxThreads()

```
void BGAPI2::Polarizer::SetMaxThreads (
            bo_uint number )
```

Set the number of threads the Polarizer can use for calculations.

To speed up the calculation of components more than one thread can be used internally. The default is 4 threads on processors which have 8 or more logical cores, otherwise half of the logical cores are used. Depending on your application you can change this here.

Parameters

in	*number*	The amount of threads used internally.

The documentation for this class was generated from the following file:

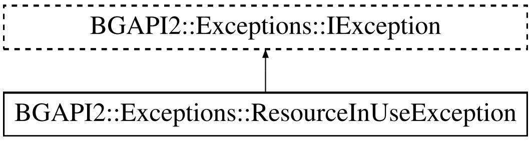
- bgapi2_genicam.hpp

9.51 BGAPI2::Exceptions::ResourceInUseException Class Reference

The requested object is already used.

`#include <bgapi2_genicam.hpp>`

Inheritance diagram for BGAPI2::Exceptions::ResourceInUseException:


BGAPI2::Exceptions::IException
BGAPI2::Exceptions::ResourceInUseException

Additional Inherited Members

9.51.1 Detailed Description

The requested object is already used.

Definition at line 3697 of file bgapi2_genicam.hpp.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.52 BGAPI2::String Class Reference

Public Member Functions

- **String** (const char ∗text)
- **String** (const char ∗text, int length)
- **String** (const char &ch, int length)
- **String** (const String &Obj)
- **String** (const char &ch)
- **operator char** ∗ ()
- **operator const char** ∗ ()
- bool **operator==** (const char ∗text)
- bool **operator==** (String &ExStr) const
- bool **operator!=** (const char ∗text)
- bool **operator!=** (const String &ExStr)
- bool **operator**< (const String &ExStr) const
- const String & **operator=** (char &)
- const String & **operator=** (const char ∗)
- const String & **operator=** (const String &ExStr)
- const char ∗ **get** () const
- void **set** (const char ∗text)
- int **size** ()

Friends

- std::ostream & **operator**<< (std::ostream &out, BGAPI2::String const &ExStr) BGAPI2_DECL

9.52.1 Detailed Description

Definition at line 13 of file bgapi2_def.h.

The documentation for this class was generated from the following file:

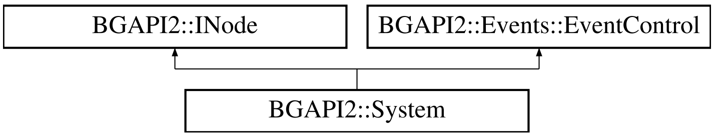
- bgapi2_def.h

9.53 BGAPI2::System Class Reference

The class System is the abstraction of a Producer and belongs to the BGAPI2 main classes.

`#include <bgapi2_genicam.hpp>`

Inheritance diagram for BGAPI2::System:



Public Member Functions

- System (String filepath)

  This constructor sets the path to the producer file.
- ∼System ()

  The destructor.
- void Open ()

  This function opens the system object and makes their functions available. Exceptions are the info-functions, which are already available before opening the system.
- void Close ()

  This function closes the system object and releases the used resources.
- bo_bool IsOpen ()

  This function delivers true, if the system is opened.
- InterfaceList ∗ GetInterfaces ()

  This function delivers the interface list of the system.
- String GetID ()

  This function delivers the unique string identifier of the System which is used in the SystemList.
- String GetVendor ()

  This function delivers the name of the GenTL producer vendor.
- String GetModel ()

  This function delivers the name of the GenTL producer to distinguish different kinds of GenTL Producer implementations.

- String GetVersion ()

  This function delivers the GenTL Producer version.
- String GetTLType ()

  This function delivers the transport layer type of the GenTL Producer.
- String GetFileName ()

  This function delivers the file name of the GenTL Producer.
- String GetPathName ()

  This function delivers the full path to the GenTL Producer including filename.
- String GetDisplayName ()

  This function delivers a meaningful name of the GenTL Producer for display only.
- void ∗ GetReserved ()

  Undocumented function.

Friends

- class **SystemList**
- class **InterfaceList**
- class **Interface**

9.53.1 Detailed Description

The class System is the abstraction of a Producer and belongs to the BGAPI2 main classes.

Definition at line 3119 of file bgapi2_genicam.hpp.

9.53.2 Constructor & Destructor Documentation

9.53.2.1 System()

```
BGAPI2::System::System (
            String filepath )
```

This constructor sets the path to the producer file.

Parameters

filepath	The path including filename of GenTL producer to be loaded.

9.53.3 Member Function Documentation

9.53.3.1 Close()

```
BGAPI2::System::Close ( )
```

This function closes the system object and releases the used resources.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.53.3.2 GetDisplayName()

```
BGAPI2::System::GetDisplayName ( )
```

This function delivers a meaningful name of the GenTL Producer for display only.

Returns

String The meaningful name of the GenTL Producer.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.53.3.3 GetFileName()

```
BGAPI2::System::GetFileName ( )
```

This function delivers the file name of the GenTL Producer.

Returns

String The file name of GenTL Producer.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.53.3.4 GetID()

```
BGAPI2::System::GetID ( )
```

This function delivers the unique string identifier of the System which is used in the SystemList.

Returns

String The unique string identifier.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.53.3.5 GetInterfaces()

```
BGAPI2::System::GetInterfaces ( )
```

This function delivers the interface list of the system.

Returns

InterfaceList* The pointer to the interface list of the system.

Exceptions

Exceptions::NotInitializedException	The system object is not opened.
Exceptions::ObjectInvalidException	The calling object is not valid.

9.53.3.6 GetModel()

```
BGAPI2::System::GetModel ( )
```

This function delivers the name of the GenTL producer to distinguish different kinds of GenTL Producer implementations.

Returns

String The name of the GenTL producer.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.53.3.7 GetPathName()

```
BGAPI2::System::GetPathName ( )
```

This function delivers the full path to the GenTL Producer including filename.

Returns

String The full path to the GenTL Producer including filename.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.53.3.8 GetTLType()

```
BGAPI2::System::GetTLType ( )
```

This function delivers the transport layer type of the GenTL Producer.

Returns

String The transport layer type of GenTL Producer.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.53.3.9 GetVendor()

```
BGAPI2::System::GetVendor ( )
```

This function delivers the name of the GenTL producer vendor.

Returns

String The name of the GenTL producer vendor.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.53.3.10 GetVersion()

```
BGAPI2::System::GetVersion ( )
```

This function delivers the GenTL Producer version.

Returns

String The GenTL producer version.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.53.3.11 IsOpen()

```
BGAPI2::System::IsOpen ( )
```

This function delivers true, if the system is opened.

Returns

delivers true, if the system is open.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.53.3.12 Open()

```
BGAPI2::System::Open ( )
```

This function opens the system object and makes their functions available. Exceptions are the info-functions, which are already available before opening the system.

Exceptions

Exceptions::ResourceInUseException	This exception will be thrown if the system object is already open.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.54 BGAPI2::SystemList Class Reference

This class is used to search and list system objects and may be instantiated only once.

```
#include <bgapi2_genicam.hpp>
```

Classes

- class iterator

  This class provides an iterator that can read or modify any object of the list.

Public Member Functions

- void Add (System ∗pSystem)

  This function adds a GenTL producer to the system list. It can be used to e.g. load GenTL producer which were not found by the searching procedure of the Refresh function.
- void Refresh ()

This functions starts the search for GenTL producers with file extension 'cti' and follows a fixed searching procedure. First, the directory containing the application is searched, then the directory containing the BGAPI GenICam-Consumer and then the directories specified in the GenICam variable (see remarks). By using the function SystemList::CreateInstanceFromPath the GenICam variable will be ignored and the specified path is used instead. The found GenTL producers will be inserted into the SystemList.

- bo_uint64 size ()

  This function delivers the number of GenTL producers in the system list.
- void clear ()

  This function removes all GenTL producer from the system list.
- System ∗ operator[] (const String &systemid)

  This operator allows the direct access to an object of the system list.
- iterator begin ()

  This function delivers an iterator on the top of the system list.
- iterator end ()

  This function delivers an iterator at the end of the system list.
- iterator find (const String &_keyval)

  This function delivers an iterator on an object that is supposed to be found. If the object cannot be found, this functions delivers with an end-iterator.

Static Public Member Functions

- static SystemList ∗ GetInstance ()

  This function creates and delivers a static instance of SystemList. The Instance will be created with the first call of this function. Every following call delivers always the same instance until the function ReleaseInstance is called. This approach makes sure that only one instance of this class can be created.
- static void ReleaseInstance ()

  This function releases the static instance of SystemList.
- static SystemList ∗ CreateInstanceFromPath (String producerpath)

  This function creates and delivers a specified static instance of SystemList. Only the passed path will be used while searching for GenTL producers. The searching procedure described in function SystemList↩ ::Refresh is not used when the Systemlist is created with this function.

Friends

- class **System**

9.54.1 Detailed Description

This class is used to search and list system objects and may be instantiated only once.

Definition at line 54 of file bgapi2_genicam.hpp.

9.54.2 Member Function Documentation

9.54.2.1 Add()

```
BGAPI2::SystemList::Add (
            System ∗ pSystem )
```

This function adds a GenTL producer to the system list. It can be used to e.g. load GenTL producer which were not found by the searching procedure of the Refresh function.

Parameters

pSystem	The GenTL producer to be added.

Returns

void

Exceptions

Exceptions::InvalidParameterException	The passed parameter is not a valid System object.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

9.54.2.2 begin()

```
BGAPI2::SystemList::begin ( )
```

This function delivers an iterator on the top of the system list.

Returns

iterator The iterator on the top of the system list.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.54.2.3 clear()

```
BGAPI2::SystemList::clear ( )
```

This function removes all GenTL producer from the system list.

Returns

void

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.54.2.4 CreateInstanceFromPath()

```
BGAPI2::SystemList::CreateInstanceFromPath (
            String producerpath )  [static]
```

This function creates and delivers a specified static instance of SystemList. Only the passed path will be used while searching for GenTL producers. The searching procedure described in function SystemList::Refresh is not used when the Systemlist is created with this function.

The specified instance of SystemList class is only created when it is used before the first call of function SystemList::GetInstance.

Parameters

producerpath	This is the searching path for GenTL producers.

Returns

The Pointer to the static instance of class SystemList.

Exceptions

Exceptions::ResourceInUseException	This exception will be thrown by repeated use or if the instance was already created by function SystemList::GetInstance.

9.54.2.5 end()

```
BGAPI2::SystemList::end ( )
```

This function delivers an iterator at the end of the system list.

Returns

iterator The iterator at the end of the system list.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.54.2.6 find()

```
BGAPI2::SystemList::find (
            const String & _keyval )
```

This function delivers an iterator on an object that is supposed to be found. If the object cannot be found, this functions delivers with an end-iterator.

Parameters

_keyval	The ID to the object to be found.

Returns

iterator The iterator to the found object.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

9.54.2.7 GetInstance()

```
BGAPI2::SystemList::GetInstance ( )  [static]
```

This function creates and delivers a static instance of SystemList. The Instance will be created with the first call of this function. Every following call delivers always the same instance until the function ReleaseInstance is called. This approach makes sure that only one instance of this class can be created.

If the function CreateInstanceFromPath is used before the first call to this function, this function delivers the pointer to the instance created by CreateInstanceFromPath.

Returns

SystemList ∗ The Pointer to the static instance of class SystemList.

9.54.2.8 operator[]()

```
BGAPI2::SystemList::operator[] (
            const String & systemid )
```

This operator allows the direct access to an object of the system list.

Parameters

systemid	For this ID, the associated system object is delivered.

Returns

System∗ The requested system object.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::InvalidParameterException	No object in the system list has the passed ID.

9.54.2.9 Refresh()

```
BGAPI2::SystemList::Refresh ( )
```

This functions starts the search for GenTL producers with file extension 'cti' and follows a fixed searching procedure. First, the directory containing the application is searched, then the directory containing the BGAPI GenICam-Consumer and then the directories specified in the GenICam variable (see remarks). By using the function SystemList::CreateInstanceFromPath the GenICam variable will be ignored and the specified path is used instead. The found GenTL producers will be inserted into the SystemList.

Returns

void

Exceptions

Exceptions::ErrorException	Internal sytem error, thrown if the GenTL producer couldn't be loaded. Check trace output for a detailed error description.
Exceptions::ObjectInvalidException	The calling object is not valid.
Exceptions::LowLevelException	GenTL Producer error. Use IException::GetErrorDescription or check trace output for more detailed error information.

Remarks

The GenICam variable on 64bit systems is defined as 'GENICAM_GENTL64_PATH', on 32bit systems 'GENICAM_GENTL32_PATH'.
Already found GenTL producers remain unaffected by the Refresh function.
If a GenTL producer is found twice in different directories, it is up to the user to use the correct one.

9.54.2.10 ReleaseInstance()

```
BGAPI2::SystemList::ReleaseInstance ( )  [static]
```

This function releases the static instance of SystemList.

Returns

void

9.54.2.11 size()

```
BGAPI2::SystemList::size ( )
```

This function delivers the number of GenTL producers in the system list.

Returns

bo_uint64 The number of GenTL producers in the system list.

Exceptions

Exceptions::ObjectInvalidException	The calling object is not valid.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.55 BGAPI2::Trace Class Reference

The class Trace offers the possibility to monitor the program flow and detect errors. This class belongs to the additional classes.

`#include <bgapi2_genicam.hpp>`

Static Public Member Functions

- static void Enable (bo_bool benable)

  This function switches the trace on.
- static void ActivateOutputToFile (bo_bool bactive, String tracefilename)

  This function activates the writing of the trace messages into the specified file.
- static void ActivateOutputToDebugger (bo_bool bactive)

  This function activates the writing of the trace messages into the debugger output.

- static void ActivateMaskError (bo_bool bactive)

  This function activates the writing of errors.
- static void ActivateMaskWarning (bo_bool bactive)

  This function activates the writing of warnings.
- static void ActivateMaskInformation (bo_bool bactive)

  This function activates the writing of information.
- static void ActivateOutputOptionTimestamp (bo_bool bactive)

  This function inserts a timestamp into the trace message.
- static void ActivateOutputOptionTimestampDiff (bo_bool bactive)

  This function inserts a timestamp difference to the last message into the current message.
- static void ActivateOutputOptionThreadId (bo_bool bactive)

  This function inserts the thread ident to the last message into the current message.
- static void ActivateOutputOptionPrefix (bo_bool bactive)

  This function inserts a short prefix into the current message which specifies the source of the message, e.g. bgapi2_genicam.dll or a Baumer GenTL producer.

9.55.1 Detailed Description

The class Trace offers the possibility to monitor the program flow and detect errors. This class belongs to the additional classes.

This class consists exclusively of static functions.

Definition at line 932 of file bgapi2_genicam.hpp.

9.55.2 Member Function Documentation

9.55.2.1 ActivateMaskError()

```
BGAPI2::Trace::ActivateMaskError (
            bo_bool bactive )  [static]
```

This function activates the writing of errors.

Parameters

bactive	This flag activates the writing of errors.

9.55.2.2 ActivateMaskInformation()

```
BGAPI2::Trace::ActivateMaskInformation (
            bo_bool bactive )  [static]
```

This function activates the writing of information.

Parameters

bactive	This flag activates the writing of information.

9.55.2.3 ActivateMaskWarning()

```
BGAPI2::Trace::ActivateMaskWarning (
            bo_bool bactive )  [static]
```

This function activates the writing of warnings.

Parameters

bactive	This flag activates the writing of warnings.

9.55.2.4 ActivateOutputOptionPrefix()

```
BGAPI2::Trace::ActivateOutputOptionPrefix (
            bo_bool bactive )  [static]
```

This function inserts a short prefix into the current message which specifies the source of the message, e.g. bgapi2_genicam.dll or a Baumer GenTL producer.

Parameters

bactive	This flag activates the inserting of a short prefix.

9.55.2.5 ActivateOutputOptionThreadId()

```
BGAPI2::Trace::ActivateOutputOptionThreadId (
            bo_bool bactive )  [static]
```

This function inserts the thread ident to the last message into the current message.

Parameters

bactive	This flag activates the inserting of the threadid.

9.55.2.6 ActivateOutputOptionTimestamp()

```
BGAPI2::Trace::ActivateOutputOptionTimestamp (
            bo_bool bactive )  [static]
```

This function inserts a timestamp into the trace message.

Parameters

bactive	This flag activates the inserting of the timestamp.

9.55.2.7 ActivateOutputOptionTimestampDiff()

```
BGAPI2::Trace::ActivateOutputOptionTimestampDiff (
            bo_bool bactive )  [static]
```

This function inserts a timestamp difference to the last message into the current message.

Parameters

bactive	This flag activates the inserting of the timestamp difference.

9.55.2.8 ActivateOutputToDebugger()

```
BGAPI2::Trace::ActivateOutputToDebugger (
            bo_bool bactive )  [static]
```

This function activates the writing of the trace messages into the debugger output.

Parameters

bactive	This flag activates the writing of the trace messages into the debugger output.

9.55.2.9 ActivateOutputToFile()

```
BGAPI2::Trace::ActivateOutputToFile (
            bo_bool bactive,
            String tracefilename )  [static]
```

This function activates the writing of the trace messages into the specified file.

Parameters

bactive	This flag activates the writing of the trace messages into a file.
tracefilename	The name of the file to which the trace messages is written.

9.55.2.10 Enable()

```
BGAPI2::Trace::Enable (
            bo_bool benable )  [static]
```

This function switches the trace on.

Parameters

benable	This flag switches the trace on.

The documentation for this class was generated from the following file:

- bgapi2_genicam.hpp

9.56 tRGB16QUAD Struct Reference

The tRGB16QUAD structure specifies the information for one color look up table entry.

```
#include <bgapi2_def.h>
```

9.56.1 Detailed Description

The tRGB16QUAD structure specifies the information for one color look up table entry.

according to RGBQUAD of WinGdi

The documentation for this struct was generated from the following file:

- bgapi2_def.h

10 File Documentation

10.1 bgapi2_featurenames.h File Reference

Macros

- #define **SFNCVERSION** 1.5
- #define **SFNC_DEVICECONTROL** "DeviceControl"
- #define SFNC_DEVICE_VENDORNAME "DeviceVendorName"

  Name of the manufacturer of the device.
- #define SFNC_DEVICE_MODELNAME "DeviceModelName"

  Model of the device.
- #define SFNC_DEVICE_MANUFACTURERINFO "DeviceManufacturerInfo"

  Manufacturer information about the device.
- #define SFNC_DEVICE_VERSION "DeviceVersion"

  Version of the device.
- #define SFNC_DEVICE_FIRMWAREVERSION "DeviceFirmwareVersion"

  Version of the firmware in the device.
- #define SFNC_DEVICE_SFNCVERSIONMAJOR "DeviceSFNCVersionMajor"

  Major version of the Standard Features Naming Convention that was used to create the device's GenI↵ Cam XML.
- #define SFNC_DEVICE_SFNCVERSIONMINOR "DeviceSFNCVersionMinor"

  Minor version of the Standard Features Naming Convention that was used to create the device's GenI↵ Cam XML.
- #define SFNC_DEVICE_SFNCVERSIONSUBMINOR "DeviceSFNCVersionSubMinor"

  Sub minor version of Standard Features Naming Convention that was used to create the device's Gen↵ ICam XML.
- #define SFNC_DEVICE_MANIFESTENTRYSELECTOR "DeviceManifestEntrySelector"

  Selects the manifest entry to reference.
- #define SFNC_DEVICE_MANIFESTXMLMAJORVERSION "DeviceManifestXMLMajorVersion"

  Indicates the major version number of the GenICam XMLfile of the selected manifest entry.
- #define SFNC_DEVICE_MANIFESTXMLMINORVERSION "DeviceManifestXMLMinorVersion"

  Indicates the minor version number of the GenICam XMLfile of the selected manifest entry.
- #define SFNC_DEVICE_MANIFESTXMLSUBMINORVERSION "DeviceManifestXMLSubMinor↵ Version"

  Indicates the subminor version number of the GenICam XMLfile of the selected manifest entry.
- #define SFNC_DEVICE_MANIFESTSCHEMAMAJORVERSION "DeviceManifestSchemaMajor↵ Version"

  Indicates the major version number of the schema file of the selected manifest entry.
- #define SFNC_DEVICE_MANIFESTSCHEMAMINORVERSION "DeviceManifestSchemaMinor↵ Version"

Indicates the minor version number of the schema file of the selected manifest entry.

- #define SFNC_DEVICE_MANIFESTPRIMARYURL "DeviceManifestPrimaryURL"

  Indicates the first URL to the GenICam XMLdevice description file of the selected manifest entry.
- #define SFNC_DEVICE_MANIFESTSECONDARYURL "DeviceManifestSecondaryURL"

  Indicates the second URL to the GenICam XMLdevice description file of the selected manifest entry.
- #define SFNC_DEVICE_ID "DeviceID"

  depreciated SFNC2.0, replaced with "DeviceSerialNumber"
- #define SFNC_DEVICE_SERIALNUMBER "DeviceSerialNumber"

  Device's serial number. This string is a unique identifier of the device.
- #define SFNC_DEVICE_USERID "DeviceUserID"

  User-programmable device identifier.
- #define SFNC_DEVICE_RESET "DeviceReset"

  Resets the device to its power up state. After reset, the device must be rediscovered.
- #define SFNC_DEVICE_REGISTERSSTREAMINGSTART "DeviceRegistersStreamingStart"

  Prepare the device for registers streaming without checking for consistency.
- #define SFNC_DEVICE_REGISTERSSTREAMINGEND "DeviceRegistersStreamingEnd"

  Announce the end of registers streaming. This will do a register set validation for consistency and activate it. This will also update the DeviceRegistersValidflag.
- #define SFNC_DEVICE_REGISTERSCHECK "DeviceRegistersCheck"

  Perform the validation of the current register setfor consistency.This will update the DeviceRegisters←Validflag.
- #define SFNC_DEVICE_REGISTERSVALID "DeviceRegistersValid"

  Returns if the current register setis valid and consistent.
- #define SFNC_DEVICE_MAXTHROUGHPUT "DeviceMaxThroughput"

  Maximum bandwidth of the data that can be streamed out of the device. This can be used to estimate if the physical connection(s)can sustain transfer of free-running images from the camera at its maximum speed.
- #define SFNC_DEVICE_TEMERATURESELECTOR "DeviceTemperatureSelector"

  Selects the location within the device, where the temperature will be measured.
- #define SFNC_DEVICE_TEMPERATURE "DeviceTemperature"

  Device temperature in degrees Celsius (C). It is measured at the location selected by Device←TemperatureSelector.
- #define SFNC_DEVICE_CLOCKSELECTOR "DeviceClockSelector"

  Selects the clock frequency to access from the device.
- #define SFNC_DEVICE_CLOCKFREQUENCY "DeviceClockFrequency"

  Returns the frequency of the selected Clock.
- #define SFNC_DEVICE_SERIALPORTSELECTOR "DeviceSerialPortSelector"

  Selects which serial port of the device to control.
- #define SFNC_IMAGEFORMATCONTROL "ImageFormatControl"

  Category for Image Format Control features.
- #define SFNC_SENSORWIDTH "SensorWidth"

  Effective width of the sensor in pixels.
- #define SFNC_SENSORHEIGHT "SensorHeight"

  Effective height of the sensor in pixels.
- #define SFNC_SENSORTABS "SensorTaps"

  Number of taps of the camera sensor.
- #define SFNC_SENSORDIGITIZATIONTABS "SensorDigitizationTaps"

  Number of digitized samples outputted simultaneously by the camera A/D conversion stage.
- #define SFNC_WIDTHMAX "WidthMax"

  Maximum width of the image (in pixels). The dimension is calculated after horizontal binning, decimation or any other function changing the horizontal dimension of the image.
- #define SFNC_HEIGHTMAX "HeightMax"

Maximum height of the image (in pixels). This dimension is calculated after vertical binning, decimation or any other function changing the vertical dimension of the image.

- #define SFNC_WIDTH "Width"

  Width of the image provided by the device (in pixels).
- #define SFNC_HEIGHT "Height"

  Height of the image provided by the device (in pixels).
- #define SFNC_OFFSETX "OffsetX"

  Horizontal offset from the origin to the region of interest (in pixels).
- #define SFNC_OFFSETY "OffsetY"

  Vertical offset from the origin to the region of interest (in pixels).
- #define SFNC_LINEPITCH "LinePitch"

  Total number of bytes between the startsof 2 consecutivelines. This feature is used to facilitate alignment of image data.
- #define SFNC_BINNINGHORIZONTAL "BinningHorizontal"

  Number of horizontal photo-sensitive cells to combine together. This reduces the horizontal resolution (width) of the image.
- #define SFNC_BINNINGVERTICAL "BinningVertical"

  Number of vertical photo-sensitive cells to combine together. This reduces the vertical resolution (height) of the image.
- #define SFNC_DECIMATIONHORIZONTAL "DecimationHorizontal"

  Horizontal sub-sampling of the image. This reduces the horizontal resolution (width) of the image by the specified horizontal decimation factor.
- #define SFNC_DECIMATIONVERTICAL "DecimationVertical"

  Vertical sub-sampling of the image. This reduces the vertical resolution (height) of the image by the specified vertical decimation factor.
- #define SFNC_REVERSEX "ReverseX"

  Flip horizontally the image sent by the device. The Region of interest is applied after the flipping.
- #define SFNC_REVERSEY "ReverseY"

  Flip vertically the image sent by the device. The Region of interest is applied after the flipping.
- #define SFNC_PIXELFORMAT "PixelFormat"

  Format of the pixels provided by the device. It represents all the information provided by PixelSize, PixelColorFilter combined in a single feature.
- #define SFNC_PIXELCODING "PixelCoding"

  This feature is deprecated. It represents the coding of the pixels in the image. Raw gives the data in the native format of the sensor.
- #define SFNC_PIXELSIZE "PixelSize"

  Total size in bits of a pixel of the image.
- #define SFNC_PIXELCOLORFILTER "PixelColorFilter"

  Type of color filter that is applied to the image.
- #define SFNC_PIXELDYNAMICRANGEMIN "PixelDynamicRangeMin"

  Minimum value that can be returned during the digitization process. This corresponds to the darkest value of the camera. For color camera, this returns the smallest value that each color component can take.
- #define SFNC_PIXELDYNAMICRANGEMAX "PixelDynamicRangeMax"

  Maximum value that will be returned during the digitization process. This corresponds to the brightest value of the camera. For color camera, this returns the biggest value that each color component can take.
- #define SFNC_TESTIMAGESELECTOR "TestImageSelector"

  This feature is deprecated(See TestPattern). Selects the type of test image that is sent by the device.
- #define SFNC_ACQUISITIONCONTROL "AcquisitionControl"

  Category for the acquisition and trigger control features.
- #define SFNC_ACQUISITION_MODE "AcquisitionMode"

Sets the acquisition mode of the device. It defines mainly the number of frames to capture during an acquisition and the way the acquisition stops.

- #define SFNC_ACQUISITION_START "AcquisitionStart"

  Starts the Acquisition of the device. The number of frames captured is specified by AcquisitionMode.
- #define SFNC_ACQUISITION_STOP "AcquisitionStop"

  Stops the Acquisition of the device at the end of the current Frame. It is mainly used when Acquisition↩ Mode is Continuous but can be used in any acquisition mode.
- #define SFNC_ACQUISITION_ABORT "AcquisitionAbort"

  Aborts the Acquisition immediately. This will end the capture without completing the current Frame or waiting on a trigger. If no Acquisition is in progress, the command is ignored.
- #define SFNC_ACQUISITION_ARM "AcquisitionArm"

  Arms the device before an AcquisitionStartcommand. This optional command validates all the current features for consistency and prepares the device for a fast start of the Acquisition.
- #define SFNC_ACQUISITION_FRAMECOUNT "AcquisitionFrameCount"

  Number of frames to acquire in MultiFrame Acquisition mode.
- #define SFNC_ACQUISITION_FRAMERATE "AcquisitionFrameRate"

  Controls the acquisition rate (in Hertz) at which the frames are captured.
- #define SFNC_ACQUISITION_FRAMERATEABS "AcquisitionFrameRateAbs"

  Deprecated.
- #define SFNC_ACQUISITION_FRAMERATERAW "AcquisitionFrameRateRaw"

  Deprecated.
- #define SFNC_ACQUISITION_LINERATE "AcquisitionLineRate"

  Controls the rate (in Hertz) at which the Lines in a Frame are captured.
- #define SFNC_ACQUISITION_LINERATEABS "AcquisitionLineRateAbs"

  Deprecated.
- #define SFNC_ACQUISITION_LINERATERAW "AcquisitionLineRateRaw"

  Deprecated.
- #define SFNC_ACQUISITION_STATUSSELECTOR "AcquisitionStatusSelector"

  Selects the internal acquisition signal to read using AcquisitionStatus.
- #define SFNC_ACQUISITION_STATUS "AcquisitionStatus"

  Reads the state of the internal acquisition signal selected using AcquisitionStatusSelector.
- #define SFNC_TRIGGERSELECTOR "TriggerSelector"

  Selects the type of trigger to configure.
- #define SFNC_TRIGGERMODE "TriggerMode"

  Controls if the selected trigger is active.
- #define SFNC_TRIGGERSOFTWARE "TriggerSoftware"

  Generates an internal trigger. TriggerSource must be set to Software.
- #define SFNC_TRIGGERSOURCE "TriggerSource"

  Specifies the internal signal or physical input Line to use as the trigger source. The selected trigger must have its TriggerMode set to On.
- #define SFNC_TRIGGERACTIVATION "TriggerActivation"

  Specifies the activation mode of the trigger.
- #define SFNC_TRIGGEROVERLAP "TriggerOverlap"

  Specifies the type trigger overlap permitted with the previous frame or line. This defines when a valid trigger will be accepted (or latched) for a new frame or a new line.
- #define SFNC_TRIGGERDELAY "TriggerDelay"

  Specifies the delay in microseconds (us) to apply after the trigger reception before activating it.
- #define SFNC_TRIGGERDELAYABS "TriggerDelayAbs"

  Deprecated.
- #define SFNC_TRIGGERDELAYRAW "TriggerDelayRaw"

  Deprecated.
- #define SFNC_TRIGGERDIVIDER "TriggerDivider"

Specifies a division factor for the incoming trigger pulses.

- #define SFNC_TRIGGERMULTIPLIER "TriggerMultiplier"

  Specifies a multiplication factor for the incoming trigger pulses.It is used generally used in conjunction with TriggerDividerto control the ratio of triggers that are accepted.
- #define SFNC_EXPOSUREMODE "ExposureMode"

  Sets the operation mode of the Exposure.
- #define SFNC_EXPOSURETIME "ExposureTime"

  Sets the Exposure time when ExposureMode is Timed and ExposureAuto is Off. This controls the duration where the photosensitive cells are exposed to light.
- #define SFNC_EXPOSURETIMEABS "ExposureTimeAbs"

  Deprecated.
- #define SFNC_EXPOSURETIMERAW "ExposureTimeRaw"

  Deprecated.
- #define SFNC_EXPOSUREAUTO "ExposureAuto"

  Sets the automatic exposure mode when ExposureMode is Timed. The exact algorithm used to implement this control is device-specific.
- #define SFNC_DIGITALIOCONTROL "DigitalIOControl"

  Category that contains the digital input and output control features.
- #define SFNC_LINESELECTOR "LineSelector"

  Selects the physical line (or pin) of the external device connector or the virtual line of the Transport Layer to configure.
- #define SFNC_LINEMODE "LineMode"

  Controls if the physical Line is used to Input or Output a signal.
- #define SFNC_LINEINVERTER "LineInverter"

  Controls the inversion of the signal of the selected input or output Line.
- #define SFNC_LINESTATUS "LineStatus"

  Returns the current status of the selected input or output Line.
- #define SFNC_LINESTATUSALL "LineStatusAll"

  Returns the current status of all available Line signals at time of polling in a single bitfield.
- #define SFNC_LINESOURCE "LineSource"

  Selects which internal acquisition or I/O source signal to output on the selected Line. LineMode must be Output.
- #define SFNC_LINEFORMAT "LineFormat"

  Controls the current electrical format of the selected physical input or output Line.
- #define SFNC_USEROUTPUTSELECTOR "UserOutputSelector"

  Selects which bit of the User Output register will be set by UserOutputValue.
- #define SFNC_USEROUTPUTVALUE "UserOutputValue"

  Sets the value of the bit selected by UserOutputSelector.
- #define SFNC_USEROUTPUTVALUEALL "UserOutputValueAll"

  Sets the value of all the bits of the User Output register. It is subject to the UserOutputValueAllMask.
- #define SFNC_USEROUTPUTVALUEALLMASK "UserOutputValueAllMask"

  Sets the write mask to apply to the value specified by UserOutputValueAllbefore writing it in the User Output register. If the UserOutputValueAllMaskfeature is present, setting the user Output register using UserOutputValueAllwill only change the bits that have a corresponding bit in the mask set to one.
- #define SFNC_COUNTERANDTIMERCONTROL "CounterAndTimerControl"

  Category that contains the CounterandTimer control features.
- #define SFNC_COUNTERSELECTOR "CounterSelector"

  Selects which Counter to configure.
- #define SFNC_COUNTEREVENTSOURCE "CounterEventSource"

  Select the events that will be the source to increment the Counter.
- #define SFNC_COUNTEREVENTACTIVATION "CounterEventActivation"

  Selects the Activation mode Event Source signal.

- #define SFNC_COUNTERRESETSOURCE "CounterResetSource"

  Selects the signals that will be the source to reset the Counter.
- #define SFNC_COUNTERRESETACTIVATION "CounterResetActivation"

  Selects the Activation mode of the Counter Reset Source signal.
- #define SFNC_COUNTERRESET "CounterReset"

  Does a software reset of the selected Counter and starts it. The counter starts counting events immediately after the reset unless a Counter trigger is active. CounterReset can be used to reset the Counter independently from the CounterResetSource. To disable the counter temporarily, set CounterEvent↩ Source to Off.
- #define SFNC_COUNTERVALUE "CounterValue"

  Reads or writes the current value of the selected Counter.
- #define SFNC_COUNTERVALUEATRESET "CounterValueAtReset"

  Reads the value of the selected Counter when it was reset by a trigger or by an explicit CounterReset command.
- #define SFNC_COUNTERDURATION "CounterDuration"

  Sets the duration (or number of events) before the CounterEnd event is generated.
- #define SFNC_COUNTERSTATUS "CounterStatus"

  Returns the current statusof the Counter.
- #define SFNC_COUNTERTRIGGERSOURCE "CounterTriggerSource"

  Selects the source to start the Counter.
- #define SFNC_COUNTERTRIGGERACTIVATION "CounterTriggerActivation"

  Selects the activation mode of the trigger to start the Counter.
- #define SFNC_TIMERSELECTOR "TimerSelector"

  Selects which Timer to configure.
- #define SFNC_TIMERDURATION "TimerDuration"

  Sets the duration (in microseconds) of the Timer pulse.
- #define SFNC_TIMERDURATIONABS "TimerDurationAbs"

  Deprecated.
- #define SFNC_TIMERDURATIONRAW "TimerDurationRaw"

  Deprecated.
- #define SFNC_TIMERDELAY "TimerDelay"

  Sets the duration (in microseconds) of the delay to apply at the reception of a trigger before starting the Timer.
- #define SFNC_TIMERDELAYABS "TimerDelayAbs"

  Deprecated.
- #define SFNC_TIMERDELAYRAW "TimerDelayRaw"

  Deprecated.
- #define SFNC_TIMERRESET "TimerReset"

  Does a software reset of the selected timer and starts it. The timer starts immediately after the reset unless a timer trigger is active.
- #define SFNC_TIMERVALUE "TimerValue"

  Reads or writes the current value (in microseconds) of the selected Timer.
- #define SFNC_TIMERVALUEABS "TimerValueAbs"

  Deprecated.
- #define SFNC_TIMERVALUERAW "TimerValueRaw"

  Deprecated.
- #define SFNC_TIMERSTATUS "TimerStatus"

  Returns the current statusof the Timer.
- #define SFNC_TIMERTRIGGERSOURCE "TimerTriggerSource"

  Selects the source of the trigger to start the Timer.
- #define SFNC_TIMERTRIGGERACTIVATION "TimerTriggerActivation"

  Selects the activation mode of the trigger to start the Timer.

- #define SFNC_EVENTCONTROL "EventControl"

  Category that contains Event control features.
- #define SFNC_EVENTSELECTOR "EventSelector"

  Selects which Event to signal to the host application.
- #define SFNC_EVENTNOTIFICATION "EventNotification"

  Activate or deactivate the notification to the host application of the occurrence of the selected Event.
- #define SFNC_EVENT_FRAMETRIGGERDATA "EventFrameTriggerData"

  Category that contains all the data features related to the FrameTrigger Event.
- #define SFNC_EVENT_FRAMETRIGGER "EventFrameTrigger"

  Returns the unique Identifier of the FrameTrigger type of Event. It can be used to register a callback function to be notified of the event occurrence. Its value uniquely identifiesthe type event received.
- #define SFNC_EVENT_FRAMETRIGGERTIMESTAMP "EventFrameTriggerTimestamp"

  Returns the Timestamp of the FrameTrigger Event. It can be used to determine precisely when the event occurred.
- #define SFNC_EVENT_FRAMETRIGGERFRAMEID "EventFrameTriggerFrameID"

  Returns the unique Identifier of the Frame (or image) that generated the FrameTrigger Event.
- #define SFNC_EVENT_EXPOSUREENDDATA "EventExposureEndData"

  Category that contains all the data features related to the ExposureEnd Event.
- #define SFNC_EVENT_EXPOSUREEND "EventExposureEnd"

  Returns the unique identifier of the ExposureEnd type of Event. This feature can be used to register a callback function to be notified of the event occurrence. Its value uniquely identifies the type of event that will be received.
- #define SFNC_EVENT_EXPOSUREENDTIMESTAMP "EventExposureEndTimestamp"

  Returns the Timestamp of the ExposureEnd Event. It can be used to determine precisely when the event occurred.
- #define SFNC_EVENT_EXPOSUREENDFRAMEID "EventExposureEndFrameID"

  Returns the unique Identifier of the Frame (or image) that generated the ExposureEnd Event.
- #define SFNC_EVENT_ERRORDATA "EventErrorData"

  Category that contains all the data features related to the Error Event.
- #define SFNC_EVENT_ERROR "EventError"

  Returns the unique identifier of the Error type of Event. It can be used to register a callbackfunction to be notified of the Error event occurrence. Its value uniquely identifies that the event received was an Error.
- #define SFNC_EVENT_ERRORTIMESTAMP "EventErrorTimestamp"

  Returns the Timestamp of the Error Event. It can be used to determine when the event occurred.
- #define SFNC_EVENT_ERRORFRAMEID "EventErrorFrameID"

  If applicable, returns the unique Identifier of the Frame (or image) that generated the Error Event.
- #define SFNC_EVENT_ERRORCODE "EventErrorCode"

  Returns an error code for the error(s) that happened.
- #define SFNC_ANALOGCONTROL "AnalogControl"

  Category that contains the Analog control features.
- #define SFNC_GAINSELECTOR "GainSelector"

  Selects which Gain is controlled by the various Gain features.
- #define SFNC_GAIN "Gain"

  Controls the selected gain as an absolute physical value. This is an amplification factor applied to the video signal.
- #define SFNC_GAINRAW "GainRaw"

  Deprecated.
- #define SFNC_GAINABS "GainAbs"

  Deprecated.
- #define SFNC_GAINAUTO "GainAuto"

Sets the automatic gain control (AGC) mode. The exact algorithm used to implement AGC is device-specific.

- #define SFNC_GAINAUTOBALANCE "GainAutoBalance"

  Sets the mode for automatic gain balancing between the sensor color channels or taps. The gain coefficients of each channel or tap are adjusted so they are matched.
- #define SFNC_BLACKLEVELSELECTOR "BlackLevelSelector"

  Selects which Black Level is controlled by the various Black Level features.
- #define SFNC_BLACKLEVEL "BlackLevel"

  Controls the analog black level as an absolute physical value. This represents a DC offset applied to the video signal.
- #define SFNC_BLACKLEVELRAW "BlackLevelRaw"

  Deprecated.
- #define SFNC_BLACKLEVELABS "BlackLevelAbs"

  Deprecated.
- #define SFNC_BLACKLEVELAUTO "BlackLevelAuto"

  Controls the mode for automatic black level adjustment. The exact algorithm used to implement this adjustment is device-specific.
- #define SFNC_BLACKLEVELAUTOBALANCE "BlackLevelAutoBalance"

  Controls the mode for automatic black level balancing between the sensor color channels or taps. The-black level coefficients of each channel are adjusted so they are matched.
- #define SFNC_WHITECLIPSELECTOR "WhiteClipSelector"

  Selects which White Clip to control.
- #define SFNC_WHITECLIP "WhiteClip"

  Controls the maximal intensity taken by the video signal before being clipped as an absolute physical value. The video signal will never exceed the white clipping point: it will saturate at that level.
- #define SFNC_WHITECLIPRAW "WhiteClipRaw"

  Deprecated.
- #define SFNC_WHITECLIPABS "WhiteClipAbs"

  Deprecated.
- #define SFNC_BALANCERATIOSELECTOR "BalanceRatioSelector"

  Selects which Balance ratio to control.
- #define SFNC_BALANCERATIO "BalanceRatio"

  Controls ratio of the selected color component to a reference color component. It is used for white balancing.
- #define SFNC_BALANCERATIOABS "BalanceRatioAbs"

  Deprecated.
- #define SFNC_BALANCEWHITEAUTO "BalanceWhiteAuto"

  Controls the mode for automatic white balancing between the color channels. The white balancing ratios are automatically adjusted.
- #define SFNC_GAMMA "Gamma"

  Controls the gamma correction of pixel intensity. This is typically used to compensate for non-linearity of the display system (such as CRT).
- #define SFNC_LUTCONTROL "LUTControl"

  Category that includes the LUT control features.
- #define SFNC_LUTSELECTOR "LUTSelector"

  Selects which LUT to control.
- #define SFNC_LUTENABLE "LUTEnable"

  Activates the selected LUT.
- #define SFNC_LUTINDEX "LUTIndex"

  Control the index (offset) of the coefficient to access in the selected LUT.
- #define SFNC_LUTVALUE "LUTValue"

  Returns the Value at entry LUTIndexof the LUT selected by LUTSelector.

- #define SFNC_LUTVALUEALL "LUTValueAll"

  Accesses all the LUT coefficients in a single access without using individual LUTIndex.
- #define SFNC_ROOT "Root"

  Provides the Root of the GenICam features tree.
- #define SFNC_DEVICE "Device"

  Provides the default GenICam port of the Device.
- #define SFNC_TLPARAMSLOCKED "TLParamsLocked"

  Used by the Transport Layer to prevent critical features from changing during acquisition.
- #define SFNC_TRANSPORTLAYERCONTROL "TransportLayerControl"

  Category that contains the transport Layer control features.
- #define SFNC_PAYLOADSIZE "PayloadSize"

  Provides the number of bytes transferred for each image or chunk on the stream channel. This includes any end-of-line, end-of-frame statistics or other stamp data. This is the total size of data payload for a data block.
- #define SFNC_GEV_VERSIONMAJOR "GevVersionMajor"

  This feature is deprecated (See DeviceTLVersionMajor). It was representing the major version of the specification.
- #define SFNC_GEV_VERSIONMINOR "GevVersionMinor"

  This feature is deprecated (See DeviceTLVersionMinor). It was representing the minor version of the specification.
- #define SFNC_GEV_DEVICEMODEISBIGENDIAN "GevDeviceModeIsBigEndian"

  This feature is deprecated (See DeviceRegistersEndianness). It was representing the Endianness of the device registers.
- #define SFNC_GEV_DEVICECALSS "GevDeviceClass"

  This feature is deprecated (See DeviceType). It was representing the class of the device.
- #define SFNC_GEV_DEVICEMODECHARACTERSET "GevDeviceModeCharacterSet"

  This feature is deprecated (See DeviceCharacterSet). It was representing the character set used by all the strings of the bootstrap registers.
- #define SFNC_GEV_INTERFACESELECTOR "GevInterfaceSelector"

  Selects which logical link to control.
- #define SFNC_GEV_MACADDRESS "GevMACAddress"

  MAC address of the logical link.
- #define SFNC_GEV_SUPPORTEDOPTIONSELECTOR "GevSupportedOptionSelector"

  Selects the GEV option to interrogate for existing support.
- #define SFNC_GEV_SUPPORTEDOPTION "GevSupportedOption"

  Returns if the selected GEV option is supported.
- #define **SFNC_GEV_SUPPORTEDIPCONFIGURATIONLLA** "GevSupportedIPConfigurationLLA"
- #define **SFNC_GEV_SUPPORTEDIPCONFIGURATIONDHCP** "GevSupportedIPConfigurationDHCP"
- #define **SFNC_GEV_SUPPORTEDIPCONFIGURATIONPERSISTENTIP** "GevSupportedIPConfigurationPersistentIP"
- #define **SFNC_GEV_CURRENTIPCONFIGURATION** "GevCurrentIPConfiguration"
- #define **SFNC_GEV_CURRENTIPCONFIGURATIONLLA** "GevCurrentIPConfigurationLLA"
- #define **SFNC_GEV_CURRENTIPCONFIGURATIONDHCP** "GevCurrentIPConfigurationDHCP"
- #define **SFNC_GEV_CURRENTIPCONFIGURATIONPERSISTENTIP** "GevCurrentIPConfigurationPersistentIP"
- #define SFNC_GEV_CURRENTIPADDRESS "GevCurrentIPAddress"

  Reports the IP address for the given logical link.
- #define SFNC_GEV_CURRENTSUBNETMASK "GevCurrentSubnetMask"

  Reports the subnet mask of the given logical link.
- #define SFNC_GEV_CURRENTDEFAULTGATEWAY "GevCurrentDefaultGateway"

  Reports the default gateway IP address to be used on the given logical link.
- #define SFNC_GEV_IPCONFIGURATIONSTATUS "GevIPConfigurationStatus"

Reports the current IP configuration status.

- #define SFNC_GEV_FIRSTURL "GevFirstURL"

  Deprecated! Indicates the first URL to the GenICam XML device description file. The First URL is used as the first choice by the application to retrieve the GenICam XML device description file.
- #define SFNC_GEV_SECONDURL "GevSecondURL"

  Deprecated! Indicates the second URL to the GenICam XML device description file. This URL is an alternative if the application was unsuccessful to retrieve the device description file using the first URL.
- #define SFNC_GEV_NUMBEROFINTERFACES "GevNumberOfInterfaces"

  This feature is deprecated (See DeviceLinkSelector). It was representing the number of logical links supported by this device.
- #define SFNC_GEV_PERSISTENTIPADDRESS "GevPersistentIPAddress"

  Controls the Persistent IP address for this logical link. It is only used when the device boots with the Persistent IP configuration scheme.
- #define SFNC_GEV_PERSISTENTSUBNETMASK "GevPersistentSubnetMask"

  Controls the Persistent subnet mask associated with the Persistent IP address on this logical link. It is only used when the device boots with the Persistent IP configuration scheme.
- #define SFNC_GEV_PERSISTENTDEFAULTGATEWAY "GevPersistentDefaultGateway"

  Controls the persistent default gateway for this logical link. It is only used when the device boots with the Persistent IP configuration scheme.
- #define SFNC_GEV_GEVLINKSPEED "GevLinkSpeed"

  This feature is deprecated (See DeviceLinkSpeed). It was representing the speed of transmission negotiated by the given logical link.
- #define SFNC_GEV_MESSAGECHANNELCOUNT "GevMessageChannelCount"

  This feature is deprecated (See DeviceEventChannelCount). It was representing the number of message channels supported by this device.
- #define SFNC_GEV_STREAMCHANNELCOUNT "GevStreamChannelCount"

  This feature is deprecated (See DeviceStreamChannelCount). It was representing the number of stream channels supported by this device.
- #define **SFNC_GEV_SUPPORTEDOPTIONALCOMMANDSUSERDEFINEDNAME** "GevSupported↩ OptionalCommandsUserDefinedName"
- #define **SFNC_GEV_SUPPORTEDOPTIONALCOMMANDSSERIALNUMBER** "GevSupported↩ OptionalCommandsSerialNumber"
- #define **SFNC_GEV_SUPPORTEDOPTIONALCOMMANDSEVENTDATA** "GevSupportedOptional↩ CommandsEVENTDATA"
- #define **SFNC_GEV_SUPPORTEDOPTIONALCOMMANDSEVENT** "GevSupportedOptional↩ CommandsEVENT"
- #define **SFNC_GEV_SUPPORTEDOPTIONALCOMMANDSPACKETRESEND** "GevSupported↩ OptionalCommandsPACKETRESEND"
- #define **SFNC_GEV_SUPPORTEDOPTIONALCOMMANDSWRITEMEM** "GevSupportedOptional↩ CommandsWRITEMEM"
- #define **SFNC_GEV_SUPPORTEDOPTIONALCOMMANDSCONCATENATION** "GevSupported↩ OptionalCommandsConcatenation"
- #define SFNC_GEV_HEARTBEATTIMEOUT "GevHeartbeatTimeout"

  This feature is deprecated (See DeviceLinkHeartbeatTimeout). It was controling the current heartbeat timeout in milliseconds.
- #define SFNC_GEV_TIMESTAMPTICKFREQUENCY "GevTimestampTickFrequency"

  This feature is deprecated (See the increment of the TimestampLatchValue feature). It was used to indicate the number of timestamp ticks in 1 second (frequency in Hz). If IEEE 1588 is used, this feature must return 1,000,000,000 (1 GHz).
- #define SFNC_GEV_TIMESTAMPCONTROLLATCH "GevTimestampControlLatch"

  This feature is deprecated (See TimestampLatch). It was used to latche the current timestamp counter into GevTimestampValue.
- #define SFNC_GEV_TIMESTAMPCONTROLRESET "GevTimestampControlReset"

  This feature is deprecated (See TimestampReset). It was used to reset the timestamp counter to 0. This feature is not available or as no effect when IEEE 1588 is used.

- #define SFNC_GEV_TIMESTAMPVALUE "GevTimestampValue"

  This feature is deprecated (See TimestampLatchValue). It was used to return the latched 64-bit value of the timestamp counter.
- #define SFNC_GEV_DISCOVERYACKDELAY "GevDiscoveryAckDelay"

  Indicates the maximum randomized delay the device will wait to acknowledge a discovery command.
- #define SFNC_GEV_GVCPEXTENDEDSTATUSCODES "GevGVCPExtendedStatusCodes"

  Enables the generation of extended status codes.
- #define SFNC_GEV_GVCPPENDINGACK "GevGVCPPendingAck"

  Enables the generation of PENDING_ACK.
- #define SFNC_GEV_GVCPHEARTBEATDISABLE "GevGVCPHeartbeatDisable"

  This feature is deprecated (See DeviceHeartbeatMode).It was used to disable the GVCP heartbeat.
- #define SFNC_GEV_GVCPPENDINGTIMEOUT "GevGVCPPendingTimeout"

  This feature is deprecated (See DeviceLinkCommandTimeout). It was used to indicate the longest GVCP command execution time before a device returns a PENDING_ACK.
- #define SFNC_GEV_PRIMARYAPPLICATIONSWITCHOVERKEY "GevPrimaryApplicationSwitchover↩ Key"

  Controls the key to use to authenticate primary application switchover requests.
- #define SFNC_GEV_CCP "GevCCP"

  Controls the device access privilege of an application.
- #define SFNC_GEV_PRIMARYAPPLICATIONSOCKET "GevPrimaryApplicationSocket"

  Returns the UDP source port of the primary application.
- #define SFNC_GEV_PRIMARYAPPLICATIONIPADDRESS "GevPrimaryApplicationIPAddress"

  Returns the address of the primary application.
- #define SFNC_GEV_MCPHOSTPORT "GevMCPHostPort"

  Controls the port to which the device must send messages. Setting this value to 0 closes the message channel.
- #define SFNC_GEV_MCDA "GevMCDA"

  Controls the destination IP address for the message channel.
- #define SFNC_GEV_MCTT "GevMCTT"

  Provides the transmission timeout value in milliseconds.
- #define SFNC_GEV_MCRC "GevMCRC"

  Controls the number of retransmissions allowed when a message channel message times out.
- #define SFNC_GEV_MCSP "GevMCSP"

  This feature indicates the sourceport for the message channel.
- #define SFNC_GEV_STREAMCHANNELSELECTOR "GevStreamChannelSelector"

  Selects the stream channel to control.
- #define SFNC_GEV_SCCFGUNCONDITIONALSTREAMING "GevSCCFGUnconditionalStreaming"

  Enables the alternate IP destination for stream packets resent due to a packet resend request. When True, the source IP address provided in the packet resend command packet is used. When False, the value set in the GevSCDA[GevStreamChannelSelector]featureis used.
- #define SFNC_GEV_SCCFGEXTENDEDCHUNKDATA "GevSCCFGExtendedChunkData"

  Enables cameras to use the extended chunk data payload type for this stream channel.
- #define SFNC_GEV_SCPDIRECTION "GevSCPDirection"

  This feature is deprecated (See DeviceStreamChannelType). It was used to report the direction of the stream channel.
- #define SFNC_GEV_SCPINTERFACEINDEX "GevSCPInterfaceIndex"

  Index of the logical link to use.
- #define SFNC_GEV_SCPHOSTPORT "GevSCPHostPort"

  Controls the port of the selected channel to which a GVSP transmitter must send data stream or the port from which a GVSP receiver may receive data stream. Setting this value to 0 closes the stream channel.
- #define SFNC_GEV_SCPSFIRETESTPACKET "GevSCPSFireTestPacket"

  Sends a test packet. When this feature is set, the device will fire one test packet.

- #define SFNC_GEV_SCPSDONOTFRAGMENT "GevSCPSDoNotFragment"

  The state of this feature is copied into the "do not fragment" bit of IP header of eachstream packet. It can be used by the application to prevent IP fragmentation of packets on the stream channel.
- #define SFNC_GEV_SCPSBIGENDIAN "GevSCPSBigEndian"

  This feature is deprecated (See DeviceStreamChannelEndianness). It was used to control the endianness of multi-byte pixel data for this stream.
- #define SFNC_GEV_SCPSPACKETSIZE "GevSCPSPacketSize"

  Version 2.3Standard FeaturesNaming Convention2016-5-26Page477of 519CategoryGigEVisionLevel↩ RecommendedInterfaceIIntegerAccessRead/(Write)UnitBVisibilityExpertValues>0This GigE Vision specific feature corresponds to DeviceStreamChannelPacketSizeand should be kept in sync with it. It specifiesthe stream packet size, in bytes, to send on the selected channel for a GVSP transmitter or specifies the maximum packet size supported by a GVSP receiver.This does not include data leader and data trailer and the last data packet which might be of smaller size (since packet size is not necessarily a multiple of block size for stream channel).
- #define SFNC_GEV_SCPD "GevSCPD"

  Controls the delay (in GEV timestamp counter unit) to insert between each packet for this stream channel. This can be used as a crude flow-control mechanism if the application or the network infrastructure cannot keep up with the packets coming from the device.
- #define SFNC_GEV_SCDA "GevSCDA"

  Controls the destination IP address of the selected stream channel to which a GVSP transmitter must send data stream or the destination IP address from which a GVSP receiver may receive data stream.
- #define SFNC_GEV_SCSP "GevSCSP"

  Indicates the source port of the stream channel.
- #define SFNC_GEV_MANIFESTENTRYSELECTOR "GevManifestEntrySelector"

  Deprecated.
- #define SFNC_GEV_MANIFESTXMLMAJORVERSION "GevManifestXMLMajorVersion"

  Deprecated.
- #define SFNC_GEV_MANIFESTXMLMINORVERSION "GevManifestXMLMinorVersion"

  Deprecated.
- #define SFNC_GEV_MANIFESTXMLSUBMINORVERSION "GevManifestXMLSubMinorVersion"

  Deprecated.
- #define SFNC_GEV_MANIFESTSCHEMAMAJORVERSION "GevManifestSchemaMajorVersion"

  Deprecated.
- #define SFNC_GEV_MANIFESTSCHEMAMINORVERSION "GevManifestSchemaMinorVersion"

  Deprecated.
- #define SFNC_GEV_MANIFESTPRIMARYURL "GevManifestPrimaryURL"

  Deprecated.
- #define SFNC_GEV_MANIFESTSECONDARYURL "GevManifestSecondaryURL"

  Deprecated.
- #define SFNC_CL_CONFIGURATION "ClConfiguration"

  Deprecated.
- #define SFNC_CL_TIMESLOTSCOUNT "ClTimeSlotsCount"

  Deprecated.
- #define SFNC_DEVICETAPGEOMETRY "DeviceTapGeometry"

  This device tap geometry feature describes the geometrical properties characterizing the taps of a camera as presented at the output of the device.
- #define SFNC_USERSETCONTROL "UserSetControl"

  Category that contains the User Set control features.
- #define SFNC_USERSETSELECTOR "UserSetSelector"

  Selects the feature User Set to load, save or configure.
- #define SFNC_USERSETLOAD "UserSetLoad"

  Loads the User Set specified by UserSetSelectorto the device and makes it active.
- #define SFNC_USERSETSAVE "UserSetSave"

Save the User Set specified by UserSetSelectorto the non-volatile memory of the device.

- #define SFNC_USERSETDEFAULTSELECTOR "UserSetDefaultSelector"

  This feature is deprecated (See UserSetDefault). Selects the feature User Set to load and make active when the device is reset.
- #define SFNC_CHUNKDATACONTROL "ChunkDataControl"

  Category that contains the Chunk Data control features.
- #define SFNC_CHUNKMODEACTIVE "ChunkModeActive"

  Activates the inclusion of Chunk data in the payload of the image.
- #define SFNC_CHUNKSELECTOR "ChunkSelector"

  Selects which Chunk to enable or control.
- #define SFNC_CHUNKENABLE "ChunkEnable"

  Enables the inclusion of the selected Chunk data in the payload of the image.
- #define SFNC_CHUNKIMAGE "ChunkImage"

  Returns the entire image data included in the payload.
- #define SFNC_CHUNKOFFSETX "ChunkOffsetX"

  Returns the OffsetXof the image included in the payload.
- #define SFNC_CHUNKOFFSETY "ChunkOffsetY"

  Returns the OffsetYof the image included in the payload.
- #define SFNC_CHUNKWIDTH "ChunkWidth"

  Returns the Widthof the image included in the payload.
- #define SFNC_CHUNKHEIGHT "ChunkHeight"

  Returns the Heightof the image included in the payload.
- #define SFNC_CHUNKPIXELFORMAT "ChunkPixelFormat"

  Returns the PixelFormatof the image included in the payload.
- #define SFNC_CHUNKPIXELDYNAMICRANGEMIN "ChunkPixelDynamicRangeMin"

  Returns the minimum value of dynamic range of the image included in the payload.
- #define SFNC_CHUNKPIXELDYNAMICRANGEMAX "ChunkPixelDynamicRangeMax"

  Returns themaximum value of dynamic range of the image included in the payload.
- #define **SFNC_CHUNKDYNAMICRANGEMIN** "ChunkDynamicRangeMin"
- #define **SFNC_CHUNKDYNAMICRANGEMAX** "ChunkDynamicRangeMax"
- #define SFNC_CHUNKTIMESTAMP "ChunkTimestamp"

  Returns the Timestamp of the image included in the payload at the time of the FrameStart internal event.
- #define SFNC_CHUNKLINESTATUSALL "ChunkLineStatusAll"

  Returns the status of all the I/O lines at the time of the FrameStart internal event.
- #define SFNC_CHUNKCOUNTERSELECTOR "ChunkCounterSelector"

  Selects which counter to retrieve data from.
- #define SFNC_CHUNKCOUNTERVALUE "ChunkCounterValue"

  Returns the value of the selected Chunk counter at the time of the FrameStart event.
- #define **SFNC_CHUNKCOUNTER** "ChunkCounter"
- #define SFNC_CHUNKTIMERSELECTOR "ChunkTimerSelector"

  Selects which Timer to retrieve data from.
- #define SFNC_CHUNKTIMERVALUE "ChunkTimerValue"

  Returns the value of the selected Timer at the time of the FrameStart internal event.
- #define **SFNC_CHUNKTIMER** "ChunkTimer"
- #define SFNC_CHUNKEXPOSURETIME "ChunkExposureTime"

  Returns the exposure time used to capture the image.
- #define SFNC_CHUNKGAINSELECTOR "ChunkGainSelector"

  Selects which Gain to return.
- #define SFNC_CHUNKGAIN "ChunkGain"

  Returns the gain used to capture the image.
- #define SFNC_CHUNKBLACKLEVELSELECTOR "ChunkBlackLevelSelector"

Selects which Black Level to return.

- #define **SFNC_CHUNKBLACKLEVEL** "ChunkBlackLevel"
- #define SFNC_CHUNKLINEPITCH "ChunkLinePitch"

  Returns the LinePitchof the image included in the payload.
- #define SFNC_CHUNKFRAMEID "ChunkFrameID"

  Returns the unique Identifier of the frame (or image) included in the payload.
- #define **SFNC_CHUNKBINNINGVERTICALID** "ChunkBinningVertical"
- #define **SFNC_CHUNKBINNINGHORIZONTALID** "ChunkBinningHorizontal"
- #define SFNC_FILEACCESSCONTROL "FileAccessControl"

  Category that contains the File Access control features.
- #define SFNC_FILESELECTOR "FileSelector"

  Selects the target file in the device.
- #define SFNC_FILEOPERATIONSELECTOR "FileOperationSelector"

  Selects the target operation for the selected file in the device. This Operation is executed when the FileOperationExecutefeature is called.
- #define SFNC_FILEOPERATIONEXECUTE "FileOperationExecute"

  Executes the operationselected by FileOperationSelector on the selected file.
- #define SFNC_FILEOPENMODE "FileOpenMode"

  Selects the access mode in which a file is opened in the device.
- #define SFNC_FILEACCESSBUFFER "FileAccessBuffer"

  Defines the intermediate access buffer that allows the exchange of data between the device file storage and the application.
- #define SFNC_FILEACCESSOFFSET "FileAccessOffset"

  Controls the Offset of the mapping between the device file storage and the FileAccessBuffer.
- #define SFNC_FILEACCESSLENGTH "FileAccessLength"

  Controls the Length of the mapping between the device file storage and the FileAccessBuffer.
- #define SFNC_FILEOPERATIONSTATUS "FileOperationStatus"

  Represents the file operation execution status.
- #define SFNC_FILEOPERATIONRESULT "FileOperationResult"

  Represents the file operation result. For Read or Write operations, the number of successfully read/written bytes is returned.
- #define SFNC_FILESIZE "FileSize"

  Represents the size of the selected file in bytes.
- #define SFNC_COLORTRANSFORMATIONCONTROL "ColorTransformationControl"

  Category that contains the Color Transformation control features.
- #define SFNC_COLORTRANSFORMATIONSELECTOR "ColorTransformationSelector"

  Selects which Color Transformation module is controlled by the various Color Transformation features.
- #define SFNC_COLORTRANSFORMATIONENABLE "ColorTransformationEnable"

  Activates the selected Color Transformation module.
- #define SFNC_COLORTRANSFORMATIONVALUESELECTOR "ColorTransformationValueSelector"

  Selects the Gain factor or Offset of the Transformation matrix to access in the selected Color Transformation module.
- #define SFNC_COLORTRANSFORMATIONVALUE "ColorTransformationValue"

  Represents the value of the selected Gain factor or Offset inside the Transformation matrix.
- #define SFNC_ACTIONCONTROL "ActionControl"

  Category that contains the Action control features.
- #define SFNC_ACTIONDEVICEKEY "ActionDeviceKey"

  Provides the device key that allows the device to check the validity of action commands. The device internal assertion of an action signal is only authorized if the ActionDeviceKeyand the action device key value in the protocol message are equal.
- #define SFNC_ACTIONSELECTOR "ActionSelector"

  Selects to which Action Signal further Action settings apply.

- #define SFNC_ACTIONGROUPMASK "ActionGroupMask"

  Provides the mask that the device will use to validate the action on reception of the action protocol message.
- #define SFNC_ACTIONGROUPKEY "ActionGroupKey"

  Provides the key that the device will use to validate the action on reception of the action protocol message.
- #define GENTL_SFNC_TLPORT "TLPort"

  The GenICam port through which the System module is accessed.
- #define GENTL_SFNC_TLVENDORNAME "TLVendorName"

  Name of the GenTL Producer vendor.
- #define GENTL_SFNC_TLMODELNAME "TLModelName"

  Name of the GenTL Producer to distinguish different kinds of GenTL Producer implementations from one vendor.
- #define GENTL_SFNC_TLID "TLID"

  Unique identifier of the GenTL Producer like a GUID.
- #define GENTL_SFNC_TLVERSION "TLVersion"

  Vendor specific version string of the GenTL Producer.
- #define GENTL_SFNC_TLPATH "TLPath"

  Full path to the GenTL Producer including filename and extension.
- #define GENTL_SFNC_TLTYPE "TLType"

  Transport layer type of the GenTL Producer implementation.
- #define GENTL_SFNC_GENTLVERSIONMAJOR "GenTLVersionMajor"

  Major version number of the GenTL specification the GenTL Producer implementation complies with.
- #define GENTL_SFNC_GENTLVERSIONMINOR "GenTLVersionMinor"

  Minor version number of the GenTL specification the GenTL Producer implementation complies with.
- #define GENTL_SFNC_GENTLINTERFACEUPDATELIST "InterfaceUpdateList"

  Updates the internal list of the interfaces. This feature should be readable if the execution cannot performed immediately. The command then returns and the status can be polled. This function interacts with the TLUpdateInterfaceList function of the GenTL Producer. It is up to the GenTL Consumer to handle access in case both methods are used.
- #define GENTL_SFNC_GENTLINTERFACESELECTOR "InterfaceSelector"

  Selector for the different GenTL Producer interfaces. This interface list only changes on execution of "InterfaceUpdateList". The selector is 0-based in order to match the index of the C interface.
- #define GENTL_SFNC_GENTLINTERFACEID "InterfaceID"

  GenTL Producer wide unique identifier of the selected interface.
- #define GENTL_SFNC_GEVVERSIONMAJOR "GevVersionMajor"

  This feature is deprecated (See InterfaceTLVersionMajor). Major version number of the GigE Vision specification the GenTL Producer implementation complies with.
- #define GENTL_SFNC_GEVVERSIONMINOR "GevVersionMinor"

  This feature is deprecated (See InterfaceTLVersionMinor). Minor version number of the GigE Vision specification the GenTL Producer implementation complies with.
- #define GENTL_SFNC_GEVINTERFACEMACADDRESS "GevInterfaceMACAddress"

  48-bit MAC address of the selected interface. Note that for a GenTL Producer implementation supporting GigE Vision this feature is mandatory.
- #define GENTL_SFNC_GEVINTERFACEDEFAULTIPADDRESS "GevInterfaceDefaultIPAddress"

  IP address of the first subnet of the selected interface. Note that for a GenTL Producer implementation supporting GigE Vision this feature is mandatory.
- #define GENTL_SFNC_GEVINTERFACEDEFAULTSUBNETMASK "GevInterfaceDefaultSubnetMask"

  Subnet mask of the first subnet of the selected interface. Note that for a GenTL Producer implementation supporting GigE Vision this feature is mandatory.
- #define GENTL_SFNC_GEVINTERFACEDEFAULTGATEWAY "GevInterfaceDefaultGateway"

  Gateway of the selected interface.
- #define GENTL_SFNC_INTERFACEPORT "InterfacePort"

The GenICam port through which the Interface module is accessed.

- #define GENTL_SFNC_INTERFACEID "InterfaceID"

  GenTL Producer wide unique identifier of the selected interface.
- #define GENTL_SFNC_INTERFACETYPE "InterfaceType"

  Transport layer type of the interface.
- #define GENTL_SFNC_DEVICEUPDATELIST "DeviceUpdateList"

  Updates the internal device list. This feature should be readable if the execution cannot be performed immediately. The command then returns and the status can be polled. This feature interacts with the IFUpdateDeviceList function of the GenTL Producer. It is up to the GenTL Consumer to handle access in case both methods are used.
- #define GENTL_SFNC_DEVICESELECTOR "DeviceSelector"

  Selector for the different devices on this interface. This value only changes on execution of "Device↵ UpdateList". The selector is 0-based in order to match the index of the C interface.
- #define GENTL_SFNC_DEVICEID "DeviceID"

  Interface wide unique identifier of the selected device. This value only changes on execution of the DeviceUpdateList command.
- #define GENTL_SFNC_DEVICEVENDORNAME "DeviceVendorName"

  Name of the device vendor. This value only changes on execution of the DeviceUpdateList command.
- #define GENTL_SFNC_DEVICEMODELNAME "DeviceModelName"

  Name of the device model. This value only changes on execution of the DeviceUpdateList command.
- #define GENTL_SFNC_DEVICEACCESSSTATUS "DeviceAccessStatus"

  Gives the device's access status at the moment of the last execution of the DeviceUpdateList command. This value only changes on execution of the DeviceUpdateList command.
- #define GENTL_SFNC_GEVINTERFACEGATEWAYSELECTOR "GevInterfaceGatewaySelector"

  Selector for the different gateway entries for this interface. The selector is 0-based. Note that for a GenTL Producer implementation supporting GigE Vision this feature is mandatory.
- #define GENTL_SFNC_GEVINTERFACEGATEWAY "GevInterfaceGateway"

  IP address of the selected gateway entry of this interface. Note that for a GenTL Producer implementation supporting GigE Vision this feature is mandatory.
- #define GENTL_SFNC_GEVINTERFACEMACADDRESS "GevInterfaceMACAddress"

  48-bit MAC address of the selected interface. Note that for a GenTL Producer implementation supporting GigE Vision this feature is mandatory.
- #define GENTL_SFNC_GEVINTERFACESUBNETSELECTOR "GevInterfaceSubnetSelector"

  Selector for the subnet of this interface. Note that for a GenTL Producer implementation supporting GigE Vision this feature is mandatory.
- #define GENTL_SFNC_GEVINTERFACESUBNETIPADDRESS "GevInterfaceSubnetIPAddress"

  IP address of the selected subnet of this interface. Note that for a GenTL Producer implementation supporting GigE Vision this feature is mandatory.
- #define GENTL_SFNC_GEVINTERFACESUBNETMASK "GevInterfaceSubnetMask"

  Subnet mask of the selected subnet of this interface. Note that for a GenTL Producer implementation supporting GigE Vision this feature is mandatory.
- #define GENTL_SFNC_DEVICEPORT "DevicePort"

  The GenICam port through which the Device module is accessed.Note that DevicePort is a port node (not a feature node) and is generally not accessed by the end user directly.
- #define GENTL_SFNC_DEVICEID "DeviceID"

  Interface wide unique identifier of the selected device. This value only changes on execution of the DeviceUpdateList command.
- #define GENTL_SFNC_DEVICEVENDORNAME "DeviceVendorName"

  Name of the device vendor. This value only changes on execution of the DeviceUpdateList command.
- #define GENTL_SFNC_DEVICEMODELNAME "DeviceModelName"

  Name of the device model. This value only changes on execution of the DeviceUpdateList command.
- #define GENTL_SFNC_DEVICETYPE "DeviceType"

  Transport layer type of the device.

- #define GENTL_SFNC_STREAMSELECTOR "StreamSelector"

  Selector for the different stream channels.
- #define GENTL_SFNC_STREAMID "StreamID"

  Device unique ID for the stream.
- #define GENTL_SFNC_GEVDEVICEIPADDRESS "GevDeviceIPAddress"

  Current IP address of the GVCP interface of the selected remote device.
- #define GENTL_SFNC_GEVDEVICESUBNETMASK "GevDeviceSubnetMask"

  Current subnet mask of the GVCP interface of the selected remote device.
- #define GENTL_SFNC_GEVDEVICEMACADDRESS "GevDeviceMACAddress"

  48-bit MAC address of the GVCP interface of the selected remote device.
- #define GENTL_SFNC_GEVDEVICEGATEWAY "GevDeviceGateway"

  Current gateway IP address of the GVCP interface of the selected remote device.
- #define GENTL_SFNC_DEVICEENDIANESSMECHANISM "DeviceEndianessMechanism"

  Identifies the endianess handling mode.
- #define GENTL_SFNC_STREAMPORT "StreamPort"

  The GenICam port through which the Data Stream module is accessed.
- #define GENTL_SFNC_STREAMID "StreamID"

  Device unique ID for the stream.
- #define GENTL_SFNC_STREAMANNOUNCEDBUFFERCOUNT "StreamAnnouncedBufferCount"

  Number of announced (known) buffers on this stream. This value is volatile. It may change if additional buffers are announced and/or buffers are revoked by the GenTL Consumer.
- #define **GENTL_SFNC_STREAMACQUISITIONMODESELECTOR** "StreamAcquisitionMode↩
Selector"
- #define GENTL_SFNC_STREAMANNOUNCEBUFFERMINIMUM "StreamAnnounceBufferMinimum"

  Minimal number of buffers to announce to enable selected buffer handling mode.
- #define GENTL_SFNC_STREAMTYPE "StreamType"

  Transport layer type of the Data Stream.
- #define GENTL_SFNC_BUFFERPORT "BufferPort"

  The GenICam port through which the Buffer module is accessed.
- #define GENTL_SFNC_BUFFERDATA "BufferData"

  Entire buffer data.
- #define GENTL_SFNC_BUFFERUSERDATA "BufferUserData"

  Pointer to user data casted to an integer number referencing GenTL Consumer specific data. It is reflecting the pointer provided by the user data pointer at buffer announcement.This allows the GenTL Consumer to attach information to a buffer.
- #define **GENTL_SFNC_BUFFER_CUSTOM_HOSTTIMESTAMP** "HostTimestamp"
- #define **SFNC_CHUNKSELECTORVALUE_IMAGE** "Image"
- #define **SFNC_CHUNKSELECTORVALUE_OFFSETX** "OffsetX"
- #define **SFNC_CHUNKSELECTORVALUE_OFFSETY** "OffsetY"
- #define **SFNC_CHUNKSELECTORVALUE_WIDTH** "Width"
- #define **SFNC_CHUNKSELECTORVALUE_HEIGHT** "Height"
- #define **SFNC_CHUNKSELECTORVALUE_PIXELFORMAT** "PixelFormat"
- #define **SFNC_CHUNKSELECTORVALUE_DYNAMICRANGEMAX** "DynamicRangeMax"
- #define **SFNC_CHUNKSELECTORVALUE_DYNAMICRANGEMIN** "DynamicRangeMin"
- #define **SFNC_CHUNKSELECTORVALUE_PIXELDYNAMICRANGEMAX** "PixelDynamicRange↩
Max"
- #define **SFNC_CHUNKSELECTORVALUE_PIXELDYNAMICRANGEMIN** "PixelDynamicRangeMin"
- #define **SFNC_CHUNKSELECTORVALUE_TIMESTAMP** "Timestamp"
- #define **SFNC_CHUNKSELECTORVALUE_LINESTATUSALL** "LineStatusAll"
- #define **SFNC_CHUNKSELECTORVALUE_COUNTERVALUE** "CounterValue"
- #define **SFNC_CHUNKSELECTORVALUE_TIMERVALUE** "TimerValue"
- #define **SFNC_CHUNKSELECTORVALUE_EXPOSURETIME** "ExposureTime"
- #define **SFNC_CHUNKSELECTORVALUE_GAIN** "Gain"

- #define **SFNC_CHUNKSELECTORVALUE_BLACKLEVEL** "BlackLevel"
- #define **SFNC_CHUNKSELECTORVALUE_LINEPITCH** "LinePitch"
- #define **SFNC_CHUNKSELECTORVALUE_FRAMEID** "FrameID"
- #define **SFNC_DEVICE_TEMERATURESELECTORVALUE_SENSOR** "Sensor"
- #define **SFNC_DEVICE_TEMERATURESELECTORVALUE_MAINBOARD** "Mainboard"
- #define **SFNC_DEVICE_CLOCKSELECTORVALUE_SENSOR** "Sensor"
- #define **SFNC_DEVICE_CLOCKSELECTORVALUE_SENSORDIGITIZATION** "SensorDigitization"
- #define **SFNC_DEVICE_CLOCKSELECTORVALUE_CAMERALINK** "CameraLink"
- #define **SFNC_DEVICE_SERIALPORTSELECTORVALUE_CAMERALINK** "CameraLink"
- #define **SFNC_TESTIMAGESELECTORVALUE_OFF** "Off"
- #define **SFNC_TESTIMAGESELECTORVALUE_BLACK** "Black"
- #define **SFNC_TESTIMAGESELECTORVALUE_WHITE** "White"
- #define **SFNC_TESTIMAGESELECTORVALUE_GREYHORIZONTALRAMP** "GreyHorizontalRamp"
- #define **SFNC_TESTIMAGESELECTORVALUE_GREYVERTICALRAMP** "GreyVerticalRamp"
- #define **SFNC_TESTIMAGESELECTORVALUE_GREYHORIZONTALRAMPMOVING** "Grey↩
HorizontalRampMoving"
- #define **SFNC_TESTIMAGESELECTORVALUE_GREYVERTICALRAMPMOVING** "GreyVertical↩
RampMoving"
- #define **SFNC_TESTIMAGESELECTORVALUE_HORIZONTALLINEMOVING** "HorzontalLine↩
Moving"
- #define **SFNC_TESTIMAGESELECTORVALUE_VERTICALLINEMOVING** "VerticalLineMoving"
- #define **SFNC_TESTIMAGESELECTORVALUE_COLORBAR** "ColorBar"
- #define **SFNC_TESTIMAGESELECTORVALUE_FRAMECOUNTER** "FrameCounter"
- #define **SFNC_ACQUISITION_STATUSSELECTORVALUE_ACQUISITIONTRIGGERWAIT** "Acquisition↩
TriggerWait"
- #define **SFNC_ACQUISITION_STATUSSELECTORVALUE_ACQUISITIONACTIVE** "Acquisition↩
Active"
- #define **SFNC_ACQUISITION_STATUSSELECTORVALUE_ACQUISITIONTRANSFER** "Acquisition↩
Transfer"
- #define **SFNC_ACQUISITION_STATUSSELECTORVALUE_FRAMETRIGGERWAIT** "FrameTrigger↩
Wait"
- #define **SFNC_ACQUISITION_STATUSSELECTORVALUE_FRAMEACTIVE** "FrameActive"
- #define **SFNC_ACQUISITION_STATUSSELECTORVALUE_FRAMETRANSFER** "FrameTransfer"
- #define **SFNC_ACQUISITION_STATUSSELECTORVALUE_EXPOSUREACTIVE** "ExposureActive"
- #define **SFNC_TRIGGERSELECTORVALUE_ACQUISITIONSTART** "AcquisitionStart"
- #define **SFNC_TRIGGERSELECTORVALUE_ACQUISITIONEND** "AcquisitionEnd"
- #define **SFNC_TRIGGERSELECTORVALUE_ACQUISITIONACTIVE** "AcquisitionActive"
- #define **SFNC_TRIGGERSELECTORVALUE_FRAMESTART** "FrameStart"
- #define **SFNC_TRIGGERSELECTORVALUE_FRAMEEND** "FrameEnd"
- #define **SFNC_TRIGGERSELECTORVALUE_FRAMEACTIVE** "FrameActive"
- #define **SFNC_TRIGGERSELECTORVALUE_FRAMEBURSTSTART** "FrameBurstStart"
- #define **SFNC_TRIGGERSELECTORVALUE_FRAMEBURSTEND** "FrameBurstEnd"
- #define **SFNC_TRIGGERSELECTORVALUE_FRAMEBURSTACTIVE** "FrameBurstActive"
- #define **SFNC_TRIGGERSELECTORVALUE_LINESTART** "LineStart"
- #define **SFNC_TRIGGERSELECTORVALUE_EXPOSURESTART** "ExposureStart"
- #define **SFNC_TRIGGERSELECTORVALUE_EXPOSUREEND** "ExposureEnd"
- #define **SFNC_TRIGGERSELECTORVALUE_EXPOSUREACTIVE** "ExposureActive"
- #define **SFNC_LINESELECTORVALUE_LINE0** "Line0"
- #define **SFNC_LINESELECTORVALUE_LINE1** "Line1"
- #define **SFNC_LINESELECTORVALUE_LINE2** "Line2"
- #define **SFNC_LINESELECTORVALUE_LINE3** "Line3"
- #define **SFNC_LINESELECTORVALUE_LINE4** "Line4"
- #define **SFNC_LINESELECTORVALUE_LINE5** "Line5"
- #define **SFNC_LINESELECTORVALUE_LINE6** "Line6"
- #define **SFNC_LINESELECTORVALUE_LINE7** "Line7"

- #define **SFNC_LINESELECTORVALUE_CC1** "CC1"
- #define **SFNC_LINESELECTORVALUE_CC2** "CC2"
- #define **SFNC_LINESELECTORVALUE_CC3** "CC3"
- #define **SFNC_LINESELECTORVALUE_CC4** "CC4"
- #define **SFNC_USEROUTPUTSELECTORVALUE_USEROUTPUT0** "UserOutput0"
- #define **SFNC_USEROUTPUTSELECTORVALUE_USEROUTPUT1** "UserOutput1"
- #define **SFNC_USEROUTPUTSELECTORVALUE_USEROUTPUT2** "UserOutput2"
- #define **SFNC_USEROUTPUTSELECTORVALUE_USEROUTPUT3** "UserOutput3"
- #define **SFNC_COUNTERSELECTORVALUE_COUNTER1** "Counter1"
- #define **SFNC_COUNTERSELECTORVALUE_COUNTER2** "Counter2"
- #define **SFNC_COUNTERSELECTORVALUE_COUNTER3** "Counter3"
- #define **SFNC_COUNTERSELECTORVALUE_COUNTER4** "Counter4"
- #define **SFNC_COUNTERSELECTORVALUE_COUNTER5** "Counter5"
- #define **SFNC_COUNTERSELECTORVALUE_COUNTER6** "Counter6"
- #define **SFNC_COUNTERSELECTORVALUE_COUNTER7** "Counter7"
- #define **SFNC_COUNTERSELECTORVALUE_COUNTER8** "Counter8"
- #define **SFNC_TIMERSELECTORVALUE_TIMER1** "Timer1"
- #define **SFNC_TIMERSELECTORVALUE_TIMER2** "Timer2"
- #define **SFNC_TIMERSELECTORVALUE_TIMER3** "Timer3"
- #define **SFNC_TIMERSELECTORVALUE_TIMER4** "Timer4"
- #define **SFNC_TIMERSELECTORVALUE_TIMER5** "Timer5"
- #define **SFNC_TIMERSELECTORVALUE_TIMER6** "Timer6"
- #define **SFNC_TIMERSELECTORVALUE_TIMER7** "Timer7"
- #define **SFNC_TIMERSELECTORVALUE_TIMER8** "Timer8"
- #define **SFNC_EVENTSELECTORVALUE_ACQUISITIONTRIGGER** "AcquisitionTrigger"
- #define **SFNC_EVENTSELECTORVALUE_ACQUISITIONSTART** "AcquisitionStart"
- #define **SFNC_EVENTSELECTORVALUE_ACQUISITIONEND** "AcquisitionEnd"
- #define **SFNC_EVENTSELECTORVALUE_ACQUISITIONTRANSFERSTART** "AcquisitionTransfer←
Start"
- #define **SFNC_EVENTSELECTORVALUE_ACQUISITIONTRANSFEREND** "AcquisitionTransferEnd"
- #define **SFNC_EVENTSELECTORVALUE_ACQUISITIONERROR** "AcquisitionError"
- #define **SFNC_EVENTSELECTORVALUE_FRAMETRIGGER** "FrameTrigger"
- #define **SFNC_EVENTSELECTORVALUE_FRAMESTART** "FrameStart"
- #define **SFNC_EVENTSELECTORVALUE_FRAMEEND** "FrameEnd"
- #define **SFNC_EVENTSELECTORVALUE_FRAMEBURSTSTART** "FrameBurstStart"
- #define **SFNC_EVENTSELECTORVALUE_FRAMEBURSTEND** "FrameBurstEnd"
- #define **SFNC_EVENTSELECTORVALUE_FRAMETRANSFERSTART** "FrameTransferStart"
- #define **SFNC_EVENTSELECTORVALUE_FRAMETRANSFEREND** "FrameTransferEnd"
- #define **SFNC_EVENTSELECTORVALUE_EXPOSURESTART** "ExposureStart"
- #define **SFNC_EVENTSELECTORVALUE_EXPOSUREEND** "ExposureEnd"
- #define **SFNC_EVENTSELECTORVALUE_COUNTER1START** "Counter1Start"
- #define **SFNC_EVENTSELECTORVALUE_COUNTER2START** "Counter2Start"
- #define **SFNC_EVENTSELECTORVALUE_COUNTER3START** "Counter3Start"
- #define **SFNC_EVENTSELECTORVALUE_COUNTER4START** "Counter4Start"
- #define **SFNC_EVENTSELECTORVALUE_COUNTER5START** "Counter5Start"
- #define **SFNC_EVENTSELECTORVALUE_COUNTER6START** "Counter6Start"
- #define **SFNC_EVENTSELECTORVALUE_COUNTER7START** "Counter7Start"
- #define **SFNC_EVENTSELECTORVALUE_COUNTER8START** "Counter8Start"
- #define **SFNC_EVENTSELECTORVALUE_COUNTER1END** "Counter1End"
- #define **SFNC_EVENTSELECTORVALUE_COUNTER2END** "Counter2End"
- #define **SFNC_EVENTSELECTORVALUE_COUNTER3END** "Counter3End"
- #define **SFNC_EVENTSELECTORVALUE_COUNTER4END** "Counter4End"
- #define **SFNC_EVENTSELECTORVALUE_COUNTER5END** "Counter5End"
- #define **SFNC_EVENTSELECTORVALUE_COUNTER6END** "Counter6End"

- #define **SFNC_EVENTSELECTORVALUE_COUNTER7END** "Counter7End"
- #define **SFNC_EVENTSELECTORVALUE_COUNTER8END** "Counter8End"
- #define **SFNC_EVENTSELECTORVALUE_TIMER1START** "Timer1Start"
- #define **SFNC_EVENTSELECTORVALUE_TIMER2START** "Timer2Start"
- #define **SFNC_EVENTSELECTORVALUE_TIMER3START** "Timer3Start"
- #define **SFNC_EVENTSELECTORVALUE_TIMER4START** "Timer4Start"
- #define **SFNC_EVENTSELECTORVALUE_TIMER5START** "Timer5Start"
- #define **SFNC_EVENTSELECTORVALUE_TIMER6START** "Timer6Start"
- #define **SFNC_EVENTSELECTORVALUE_TIMER7START** "Timer7Start"
- #define **SFNC_EVENTSELECTORVALUE_TIMER8START** "Timer8Start"
- #define **SFNC_EVENTSELECTORVALUE_TIMER1END** "Timer1End"
- #define **SFNC_EVENTSELECTORVALUE_TIMER2END** "Timer2End"
- #define **SFNC_EVENTSELECTORVALUE_TIMER3END** "Timer3End"
- #define **SFNC_EVENTSELECTORVALUE_TIMER4END** "Timer4End"
- #define **SFNC_EVENTSELECTORVALUE_TIMER5END** "Timer5End"
- #define **SFNC_EVENTSELECTORVALUE_TIMER6END** "Timer6End"
- #define **SFNC_EVENTSELECTORVALUE_TIMER7END** "Timer7End"
- #define **SFNC_EVENTSELECTORVALUE_TIMER8END** "Timer8End"
- #define **SFNC_EVENTSELECTORVALUE_LINE0RISINGEDGE** "Line0RisingEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE1RISINGEDGE** "Line1RisingEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE2RISINGEDGE** "Line2RisingEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE3RISINGEDGE** "Line3RisingEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE4RISINGEDGE** "Line4RisingEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE5RISINGEDGE** "Line5RisingEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE6RISINGEDGE** "Line6RisingEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE7RISINGEDGE** "Line7RisingEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE0FALLINGEDGE** "Line0FallingEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE1FALLINGEDGE** "Line1FallingEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE2FALLINGEDGE** "Line2FallingEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE3FALLINGEDGE** "Line3FallingEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE4FALLINGEDGE** "Line4FallingEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE5FALLINGEDGE** "Line5FallingEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE6FALLINGEDGE** "Line6FallingEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE7FALLINGEDGE** "Line7FallingEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE0ANYEDGE** "Line0AnyEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE1ANYEDGE** "Line1AnyEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE2ANYEDGE** "Line2AnyEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE3ANYEDGE** "Line3AnyEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE4ANYEDGE** "Line4AnyEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE5ANYEDGE** "Line5AnyEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE6ANYEDGE** "Line6AnyEdge"
- #define **SFNC_EVENTSELECTORVALUE_LINE7ANYEDGE** "Line7AnyEdge"
- #define **SFNC_EVENTSELECTORVALUE_ERROR** "Error"
- #define **SFNC_EVENTSELECTORVALUE_ERRORS** "Errors"
- #define **SFNC_GAINSELECTORVALUE_ALL** "All"
- #define **SFNC_GAINSELECTORVALUE_RED** "Red"
- #define **SFNC_GAINSELECTORVALUE_GREEN** "Green"
- #define **SFNC_GAINSELECTORVALUE_BLUE** "Blue"
- #define **SFNC_GAINSELECTORVALUE_Y** "Y"
- #define **SFNC_GAINSELECTORVALUE_U** "U"
- #define **SFNC_GAINSELECTORVALUE_V** "V"
- #define **SFNC_GAINSELECTORVALUE_TAP1** "Tap1"
- #define **SFNC_GAINSELECTORVALUE_TAP2** "Tap2"
- #define **SFNC_GAINSELECTORVALUE_TAP3** "Tap3"

- #define **SFNC_GAINSELECTORVALUE_TAP4** "Tap4"
- #define **SFNC_GAINSELECTORVALUE_TAP5** "Tap5"
- #define **SFNC_GAINSELECTORVALUE_TAP6** "Tap6"
- #define **SFNC_GAINSELECTORVALUE_TAP7** "Tap7"
- #define **SFNC_GAINSELECTORVALUE_TAP8** "Tap8"
- #define **SFNC_GAINSELECTORVALUE_ANALOGALL** "AnalogAll"
- #define **SFNC_GAINSELECTORVALUE_ANALOGRED** "AnalogRed"
- #define **SFNC_GAINSELECTORVALUE_ANALOGGREEN** "AnalogGreen"
- #define **SFNC_GAINSELECTORVALUE_ANALOGBLUE** "AnalogBlue"
- #define **SFNC_GAINSELECTORVALUE_ANALOGY** "AnalogY"
- #define **SFNC_GAINSELECTORVALUE_ANALOGU** "AnalogU"
- #define **SFNC_GAINSELECTORVALUE_ANALOGV** "AnalogV"
- #define **SFNC_GAINSELECTORVALUE_ANALOGTAP1** "AnalogTap1"
- #define **SFNC_GAINSELECTORVALUE_ANALOGTAP2** "AnalogTap2"
- #define **SFNC_GAINSELECTORVALUE_ANALOGTAP3** "AnalogTap3"
- #define **SFNC_GAINSELECTORVALUE_ANALOGTAP4** "AnalogTap4"
- #define **SFNC_GAINSELECTORVALUE_ANALOGTAP5** "AnalogTap5"
- #define **SFNC_GAINSELECTORVALUE_ANALOGTAP6** "AnalogTap6"
- #define **SFNC_GAINSELECTORVALUE_ANALOGTAP7** "AnalogTap7"
- #define **SFNC_GAINSELECTORVALUE_ANALOGTAP8** "AnalogTap8"
- #define **SFNC_GAINSELECTORVALUE_DIGITALALL** "DigitalAll"
- #define **SFNC_GAINSELECTORVALUE_DIGITALRED** "DigitalRed"
- #define **SFNC_GAINSELECTORVALUE_DIGITALGREEN** "DigitalGreen"
- #define **SFNC_GAINSELECTORVALUE_DIGITALBLUE** "DigitalBlue"
- #define **SFNC_GAINSELECTORVALUE_DIGITALY** "DigitalY"
- #define **SFNC_GAINSELECTORVALUE_DIGITALU** "DigitalU"
- #define **SFNC_GAINSELECTORVALUE_DIGITALV** "DigitalV"
- #define **SFNC_GAINSELECTORVALUE_DIGITALTAP1** "DigitalTap1"
- #define **SFNC_GAINSELECTORVALUE_DIGITALTAP2** "DigitalTap2"
- #define **SFNC_GAINSELECTORVALUE_DIGITALTAP3** "DigitalTap3"
- #define **SFNC_GAINSELECTORVALUE_DIGITALTAP4** "DigitalTap4"
- #define **SFNC_GAINSELECTORVALUE_DIGITALTAP5** "DigitalTap5"
- #define **SFNC_GAINSELECTORVALUE_DIGITALTAP6** "DigitalTap6"
- #define **SFNC_GAINSELECTORVALUE_DIGITALTAP7** "DigitalTap7"
- #define **SFNC_GAINSELECTORVALUE_DIGITALTAP8** "DigitalTap8"
- #define **SFNC_BLACKLEVELSELECTORVALUE_ALL** "All"
- #define **SFNC_BLACKLEVELSELECTORVALUE_RED** "Red"
- #define **SFNC_BLACKLEVELSELECTORVALUE_GREEN** "Green"
- #define **SFNC_BLACKLEVELSELECTORVALUE_BLUE** "Blue"
- #define **SFNC_BLACKLEVELSELECTORVALUE_Y** "Y"
- #define **SFNC_BLACKLEVELSELECTORVALUE_U** "U"
- #define **SFNC_BLACKLEVELSELECTORVALUE_V** "V"
- #define **SFNC_BLACKLEVELSELECTORVALUE_TAP1** "Tap1"
- #define **SFNC_BLACKLEVELSELECTORVALUE_TAP2** "Tap2"
- #define **SFNC_BLACKLEVELSELECTORVALUE_TAP3** "Tap3"
- #define **SFNC_BLACKLEVELSELECTORVALUE_TAP4** "Tap4"
- #define **SFNC_BLACKLEVELSELECTORVALUE_TAP5** "Tap5"
- #define **SFNC_BLACKLEVELSELECTORVALUE_TAP6** "Tap6"
- #define **SFNC_BLACKLEVELSELECTORVALUE_TAP7** "Tap7"
- #define **SFNC_BLACKLEVELSELECTORVALUE_TAP8** "Tap8"
- #define **SFNC_WHITECLIPSELECTORVALUE_ALL** "All"
- #define **SFNC_WHITECLIPSELECTORVALUE_RED** "Red"
- #define **SFNC_WHITECLIPSELECTORVALUE_GREEN** "Green"
- #define **SFNC_WHITECLIPSELECTORVALUE_BLUE** "Blue"

- #define **SFNC_WHITECLIPSELECTORVALUE_Y** "Y"
- #define **SFNC_WHITECLIPSELECTORVALUE_U** "U"
- #define **SFNC_WHITECLIPSELECTORVALUE_V** "V"
- #define **SFNC_WHITECLIPSELECTORVALUE_TAP1** "Tap1"
- #define **SFNC_WHITECLIPSELECTORVALUE_TAP2** "Tap2"
- #define **SFNC_WHITECLIPSELECTORVALUE_TAP3** "Tap3"
- #define **SFNC_WHITECLIPSELECTORVALUE_TAP4** "Tap4"
- #define **SFNC_WHITECLIPSELECTORVALUE_TAP5** "Tap5"
- #define **SFNC_WHITECLIPSELECTORVALUE_TAP6** "Tap6"
- #define **SFNC_WHITECLIPSELECTORVALUE_TAP7** "Tap7"
- #define **SFNC_WHITECLIPSELECTORVALUE_TAP8** "Tap8"
- #define **SFNC_BALANCERATIOSELECTORVALUE_ALL** "All"
- #define **SFNC_BALANCERATIOSELECTORVALUE_RED** "Red"
- #define **SFNC_BALANCERATIOSELECTORVALUE_GREEN** "Green"
- #define **SFNC_BALANCERATIOSELECTORVALUE_BLUE** "Blue"
- #define **SFNC_BALANCERATIOSELECTORVALUE_Y** "Y"
- #define **SFNC_BALANCERATIOSELECTORVALUE_U** "U"
- #define **SFNC_BALANCERATIOSELECTORVALUE_V** "V"
- #define **SFNC_BALANCERATIOSELECTORVALUE_TAP1** "Tap1"
- #define **SFNC_BALANCERATIOSELECTORVALUE_TAP2** "Tap2"
- #define **SFNC_BALANCERATIOSELECTORVALUE_TAP3** "Tap3"
- #define **SFNC_BALANCERATIOSELECTORVALUE_TAP4** "Tap4"
- #define **SFNC_BALANCERATIOSELECTORVALUE_TAP5** "Tap5"
- #define **SFNC_BALANCERATIOSELECTORVALUE_TAP6** "Tap6"
- #define **SFNC_BALANCERATIOSELECTORVALUE_TAP7** "Tap7"
- #define **SFNC_BALANCERATIOSELECTORVALUE_TAP8** "Tap8"
- #define **SFNC_LUTSELECTORVALUE_LUMINANCE** "Luminance"
- #define **SFNC_LUTSELECTORVALUE_RED** "Red"
- #define **SFNC_LUTSELECTORVALUE_GREEN** "Green"
- #define **SFNC_LUTSELECTORVALUE_BLUE** "Blue"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_IPCONFIGURATIONLLA** "IP↩ConfigurationLLA"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_IPCONFIGURATIONDHCP** "IP↩ConfigurationDHCP"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_IPCONFIGURATIONPERSISTENT↩IP** "IPConfigurationPersistentIP"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_STREAMCHANNELSOURCESOCK↩ET** "StreamChannelSourceSocket"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_MESSAGECHANNELSOURCESOC↩KET** "MessageChannelSourceSocket"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_COMMANDSCONCATENATI↩ON** "CommandsConcatenation"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_WRITEMEM** "WriteMem"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_PACKETRESEND** "PacketResend"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_EVENT** "Event"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_EVENTDATA** "EventData"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_PENDINGACK** "PendingAck"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_ACTION** "Action"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_PRIMARYAPPLICATIONSWITCH↩OVER** "PrimaryApplicationSwitchover"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_EXTENDEDSTATUSCODES** "Extended↩StatusCodes"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_DISCOVERYACKDELAY** "Discovery↩AckDelay"

- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_DISCOVERYACKDELAYWRITABLE** "DiscoveryAckDelayWritable"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_TESTDATA** "TestData"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_MANIFESTTABLE** "ManifestTable"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_CCPAPPLICATIONSOCKET** "CCPApplicationSocket"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_LINKSPEED** "LinkSpeed"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_HEARTBEATDISABLE** "HeartbeatDisable"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_SERIALNUMBER** "SerialNumber"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_USERDEFINEDNAME** "UserDefinedName"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_STREAMCHANNEL0BIGANDLITTLEENDIAN** "StreamChannel0BigAndLittleEndian"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_STREAMCHANNEL0IPREASSEMBLY** "StreamChannel0IPReassembly"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_STREAMCHANNEL0UNCONDITIONALSTREAMING** "StreamChannel0UnconditionalStreaming"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_STREAMCHANNEL0EXTENDEDCHUNKDATA** "StreamChannel0ExtendedChunkData"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_STREAMCHANNEL1BIGANDLITTLEENDIAN** "StreamChannel1BigAndLittleEndian"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_STREAMCHANNEL1IPREASSEMBLY** "StreamChannel1IPReassembly"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_STREAMCHANNEL1UNCONDITIONALSTREAMING** "StreamChannel1UnconditionalStreaming"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_STREAMCHANNEL1EXTENDEDCHUNKDATA** "StreamChannel1ExtendedChunkData"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_STREAMCHANNEL2BIGANDLITTLEENDIAN** "StreamChannel2BigAndLittleEndian"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_STREAMCHANNEL2IPREASSEMBLY** "StreamChannel2IPReassembly"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_STREAMCHANNEL2UNCONDITIONALSTREAMING** "StreamChannel2UnconditionalStreaming"
- #define **SFNC_GEV_SUPPORTEDOPTIONSELECTORVALUE_STREAMCHANNEL2EXTENDEDCHUNKDATA** "StreamChannel2ExtendedChunkData"
- #define **SFNC_USERSETSELECTORVALUE_DEFAULT** "Default"
- #define **SFNC_USERSETSELECTORVALUE_USERSET1** "UserSet1"
- #define **SFNC_USERSETSELECTORVALUE_USERSET2** "UserSet2"
- #define **SFNC_USERSETSELECTORVALUE_USERSET3** "UserSet3"
- #define **SFNC_USERSETSELECTORVALUE_USERSET4** "UserSet4"
- #define **SFNC_USERSETDEFAULTSELECTORVALUE_DEFAULT** "Default"
- #define **SFNC_USERSETDEFAULTSELECTORVALUE_USERSET1** "UserSet1"
- #define **SFNC_USERSETDEFAULTSELECTORVALUE_USERSET2** "UserSet2"
- #define **SFNC_USERSETDEFAULTSELECTORVALUE_USERSET3** "UserSet3"
- #define **SFNC_USERSETDEFAULTSELECTORVALUE_USERSET4** "UserSet4"
- #define **SFNC_CHUNKCOUNTERSELECTORVALUE_COUNTER1** "Counter1"
- #define **SFNC_CHUNKCOUNTERSELECTORVALUE_COUNTER2** "Counter2"
- #define **SFNC_CHUNKCOUNTERSELECTORVALUE_COUNTER3** "Counter3"
- #define **SFNC_CHUNKCOUNTERSELECTORVALUE_COUNTER4** "Counter4"
- #define **SFNC_CHUNKCOUNTERSELECTORVALUE_COUNTER5** "Counter5"
- #define **SFNC_CHUNKCOUNTERSELECTORVALUE_COUNTER6** "Counter6"
- #define **SFNC_CHUNKCOUNTERSELECTORVALUE_COUNTER7** "Counter7"
- #define **SFNC_CHUNKCOUNTERSELECTORVALUE_COUNTER8** "Counter8"
- #define **SFNC_CHUNKTIMERSELECTORVALUE_TIMER1** "Timer1"

- #define **SFNC_CHUNKTIMERSELECTORVALUE_TIMER2** "Timer2"
- #define **SFNC_CHUNKTIMERSELECTORVALUE_TIMER3** "Timer3"
- #define **SFNC_CHUNKTIMERSELECTORVALUE_TIMER4** "Timer4"
- #define **SFNC_CHUNKTIMERSELECTORVALUE_TIMER5** "Timer5"
- #define **SFNC_CHUNKTIMERSELECTORVALUE_TIMER6** "Timer6"
- #define **SFNC_CHUNKTIMERSELECTORVALUE_TIMER7** "Timer7"
- #define **SFNC_CHUNKTIMERSELECTORVALUE_TIMER8** "Timer8"
- #define **SFNC_CHUNKGAINSELECTORVALUE_ALL** "All"
- #define **SFNC_CHUNKGAINSELECTORVALUE_RED** "Red"
- #define **SFNC_CHUNKGAINSELECTORVALUE_GREEN** "Green"
- #define **SFNC_CHUNKGAINSELECTORVALUE_BLUE** "Blue"
- #define **SFNC_CHUNKGAINSELECTORVALUE_Y** "Y"
- #define **SFNC_CHUNKGAINSELECTORVALUE_U** "U"
- #define **SFNC_CHUNKGAINSELECTORVALUE_V** "V"
- #define **SFNC_CHUNKGAINSELECTORVALUE_TAP1** "Tap1"
- #define **SFNC_CHUNKGAINSELECTORVALUE_TAP2** "Tap2"
- #define **SFNC_CHUNKGAINSELECTORVALUE_TAP3** "Tap3"
- #define **SFNC_CHUNKGAINSELECTORVALUE_TAP4** "Tap4"
- #define **SFNC_CHUNKGAINSELECTORVALUE_TAP5** "Tap5"
- #define **SFNC_CHUNKGAINSELECTORVALUE_TAP6** "Tap6"
- #define **SFNC_CHUNKGAINSELECTORVALUE_TAP7** "Tap7"
- #define **SFNC_CHUNKGAINSELECTORVALUE_TAP8** "Tap8"
- #define **SFNC_CHUNKGAINSELECTORVALUE_ANALOGALL** "AnalogAll"
- #define **SFNC_CHUNKGAINSELECTORVALUE_ANALOGRED** "AnalogRed"
- #define **SFNC_CHUNKGAINSELECTORVALUE_ANALOGGREEN** "AnalogGreen"
- #define **SFNC_CHUNKGAINSELECTORVALUE_ANALOGBLUE** "AnalogBlue"
- #define **SFNC_CHUNKGAINSELECTORVALUE_ANALOGY** "AnalogY"
- #define **SFNC_CHUNKGAINSELECTORVALUE_ANALOGU** "AnalogU"
- #define **SFNC_CHUNKGAINSELECTORVALUE_ANALOGV** "AnalogV"
- #define **SFNC_CHUNKGAINSELECTORVALUE_ANALOGTAP1** "AnalogTap1"
- #define **SFNC_CHUNKGAINSELECTORVALUE_ANALOGTAP2** "AnalogTap2"
- #define **SFNC_CHUNKGAINSELECTORVALUE_ANALOGTAP3** "AnalogTap3"
- #define **SFNC_CHUNKGAINSELECTORVALUE_ANALOGTAP4** "AnalogTap4"
- #define **SFNC_CHUNKGAINSELECTORVALUE_ANALOGTAP5** "AnalogTap5"
- #define **SFNC_CHUNKGAINSELECTORVALUE_ANALOGTAP6** "AnalogTap6"
- #define **SFNC_CHUNKGAINSELECTORVALUE_ANALOGTAP7** "AnalogTap7"
- #define **SFNC_CHUNKGAINSELECTORVALUE_ANALOGTAP8** "AnalogTap8"
- #define **SFNC_CHUNKGAINSELECTORVALUE_DIGITALALL** "DigitalAll"
- #define **SFNC_CHUNKGAINSELECTORVALUE_DIGITALRED** "DigitalRed"
- #define **SFNC_CHUNKGAINSELECTORVALUE_DIGITALGREEN** "DigitalGreen"
- #define **SFNC_CHUNKGAINSELECTORVALUE_DIGITALBLUE** "DigitalBlue"
- #define **SFNC_CHUNKGAINSELECTORVALUE_DIGITALY** "DigitalY"
- #define **SFNC_CHUNKGAINSELECTORVALUE_DIGITALU** "DigitalU"
- #define **SFNC_CHUNKGAINSELECTORVALUE_DIGITALV** "DigitalV"
- #define **SFNC_CHUNKGAINSELECTORVALUE_DIGITALTAP1** "DigitalTap1"
- #define **SFNC_CHUNKGAINSELECTORVALUE_DIGITALTAP2** "DigitalTap2"
- #define **SFNC_CHUNKGAINSELECTORVALUE_DIGITALTAP3** "DigitalTap3"
- #define **SFNC_CHUNKGAINSELECTORVALUE_DIGITALTAP4** "DigitalTap4"
- #define **SFNC_CHUNKGAINSELECTORVALUE_DIGITALTAP5** "DigitalTap5"
- #define **SFNC_CHUNKGAINSELECTORVALUE_DIGITALTAP6** "DigitalTap6"
- #define **SFNC_CHUNKGAINSELECTORVALUE_DIGITALTAP7** "DigitalTap7"
- #define **SFNC_CHUNKGAINSELECTORVALUE_DIGITALTAP8** "DigitalTap8"
- #define **SFNC_CHUNKBLACKLEVELSELECTORVALUE_ALL** "All"
- #define **SFNC_CHUNKBLACKLEVELSELECTORVALUE_RED** "Red"

- #define **SFNC_CHUNKBLACKLEVELSELECTORVALUE_GREEN** "Green"
- #define **SFNC_CHUNKBLACKLEVELSELECTORVALUE_BLUE** "Blue"
- #define **SFNC_CHUNKBLACKLEVELSELECTORVALUE_Y** "Y"
- #define **SFNC_CHUNKBLACKLEVELSELECTORVALUE_U** "U"
- #define **SFNC_CHUNKBLACKLEVELSELECTORVALUE_V** "V"
- #define **SFNC_CHUNKBLACKLEVELSELECTORVALUE_TAP1** "Tap1"
- #define **SFNC_CHUNKBLACKLEVELSELECTORVALUE_TAP2** "Tap2"
- #define **SFNC_CHUNKBLACKLEVELSELECTORVALUE_TAP3** "Tap3"
- #define **SFNC_CHUNKBLACKLEVELSELECTORVALUE_TAP4** "Tap4"
- #define **SFNC_CHUNKBLACKLEVELSELECTORVALUE_TAP5** "Tap5"
- #define **SFNC_CHUNKBLACKLEVELSELECTORVALUE_TAP6** "Tap6"
- #define **SFNC_CHUNKBLACKLEVELSELECTORVALUE_TAP7** "Tap7"
- #define **SFNC_CHUNKBLACKLEVELSELECTORVALUE_TAP8** "Tap8"
- #define **SFNC_FILESELECTORVALUE_USERSETDEFAULT** "UserSetDefault"
- #define **SFNC_FILESELECTORVALUE_USERSET1** "UserSet1"
- #define **SFNC_FILESELECTORVALUE_USERSET2** "UserSet2"
- #define **SFNC_FILESELECTORVALUE_USERSET3** "UserSet3"
- #define **SFNC_FILESELECTORVALUE_USERSET4** "UserSet4"
- #define **SFNC_FILESELECTORVALUE_LUTLUMINANCE** "LUTLuminance"
- #define **SFNC_FILESELECTORVALUE_LUTRED** "LUTRed"
- #define **SFNC_FILESELECTORVALUE_LUTGREEN** "LUTGreen"
- #define **SFNC_FILESELECTORVALUE_LUTBLUE** "LUTBlue"
- #define **SFNC_FILEOPERATIONSELECTORVALUE_OPEN** "Open"
- #define **SFNC_FILEOPERATIONSELECTORVALUE_CLOSE** "Close"
- #define **SFNC_FILEOPERATIONSELECTORVALUE_READ** "Read"
- #define **SFNC_FILEOPERATIONSELECTORVALUE_WRITE** "Write"
- #define **SFNC_FILEOPERATIONSELECTORVALUE_DELETE** "Delete"
- #define **SFNC_COLORTRANSFORMATIONSELECTORVALUE_RGBTORGB** "RGBtoRGB"
- #define **SFNC_COLORTRANSFORMATIONSELECTORVALUE_RGBTOYUV** "RGBtoYUV"
- #define **SFNC_COLORTRANSFORMATIONVALUESELECTORVALUE_Gain00** "Gain00"
- #define **SFNC_COLORTRANSFORMATIONVALUESELECTORVALUE_Gain01** "Gain01"
- #define **SFNC_COLORTRANSFORMATIONVALUESELECTORVALUE_Gain02** "Gain02"
- #define **SFNC_COLORTRANSFORMATIONVALUESELECTORVALUE_Gain10** "Gain10"
- #define **SFNC_COLORTRANSFORMATIONVALUESELECTORVALUE_Gain11** "Gain11"
- #define **SFNC_COLORTRANSFORMATIONVALUESELECTORVALUE_Gain12** "Gain12"
- #define **SFNC_COLORTRANSFORMATIONVALUESELECTORVALUE_Gain20** "Gain20"
- #define **SFNC_COLORTRANSFORMATIONVALUESELECTORVALUE_Gain21** "Gain21"
- #define **SFNC_COLORTRANSFORMATIONVALUESELECTORVALUE_Gain22** "Gain22"
- #define **SFNC_COLORTRANSFORMATIONVALUESELECTORVALUE_Offset0** "Offset0"
- #define **SFNC_COLORTRANSFORMATIONVALUESELECTORVALUE_Offset1** "Offset1"
- #define **SFNC_COLORTRANSFORMATIONVALUESELECTORVALUE_Offset2** "Offset2"

10.1.1 Detailed Description

Copyright 2014-2018 Baumer Optronic

10.1.2 Macro Definition Documentation

10.1.2.1 GENTL_SFNC_DEVICEID [1/2]

```
#define GENTL_SFNC_DEVICEID "DeviceID"
```

Interface wide unique identifier of the selected device. This value only changes on execution of the DeviceUpdateList command.

Interface wide unique identifier of the selected device.

Definition at line 1089 of file bgapi2_featurenames.h.

10.1.2.2 GENTL_SFNC_DEVICEID [2/2]

```
#define GENTL_SFNC_DEVICEID "DeviceID"
```

Interface wide unique identifier of the selected device. This value only changes on execution of the DeviceUpdateList command.

Interface wide unique identifier of the selected device.

Definition at line 1089 of file bgapi2_featurenames.h.

10.1.2.3 GENTL_SFNC_DEVICEMODELNAME [1/2]

```
#define GENTL_SFNC_DEVICEMODELNAME "DeviceModelName"
```

Name of the device model. This value only changes on execution of the DeviceUpdateList command.

Name of the device model.

Definition at line 1095 of file bgapi2_featurenames.h.

10.1.2.4 GENTL_SFNC_DEVICEMODELNAME [2/2]

```
#define GENTL_SFNC_DEVICEMODELNAME "DeviceModelName"
```

Name of the device model. This value only changes on execution of the DeviceUpdateList command.

Name of the device model.

Definition at line 1095 of file bgapi2_featurenames.h.

10.1.2.5 GENTL_SFNC_DEVICEVENDORNAME [1/2]

```
#define GENTL_SFNC_DEVICEVENDORNAME "DeviceVendorName"
```

Name of the device vendor. This value only changes on execution of the DeviceUpdateList command.

Name of the device vendor.

Definition at line 1092 of file bgapi2_featurenames.h.

10.1.2.6 GENTL_SFNC_DEVICEVENDORNAME [2/2]

```
#define GENTL_SFNC_DEVICEVENDORNAME "DeviceVendorName"
```

Name of the device vendor. This value only changes on execution of the DeviceUpdateList command.

Name of the device vendor.

Definition at line 1092 of file bgapi2_featurenames.h.

10.1.2.7 GENTL_SFNC_GEVINTERFACEMACADDRESS [1/2]

```
#define GENTL_SFNC_GEVINTERFACEMACADDRESS "GevInterfaceMACAddress"
```

48-bit MAC address of the selected interface. Note that for a GenTL Producer implementation supporting GigE Vision this feature is mandatory.

48-bit MAC address of this interface. Note that for a GenTL Producer implementation supporting GigE Vision this feature is mandatory.

Definition at line 1072 of file bgapi2_featurenames.h.

10.1.2.8 GENTL_SFNC_GEVINTERFACEMACADDRESS [2/2]

```
#define GENTL_SFNC_GEVINTERFACEMACADDRESS "GevInterfaceMACAddress"
```

48-bit MAC address of the selected interface. Note that for a GenTL Producer implementation supporting GigE Vision this feature is mandatory.

48-bit MAC address of this interface. Note that for a GenTL Producer implementation supporting GigE Vision this feature is mandatory.

Definition at line 1072 of file bgapi2_featurenames.h.

10.2 bgapi2_genicam.hpp File Reference

Classes

- class BGAPI2::SystemList

  This class is used to search and list system objects and may be instantiated only once.
- class BGAPI2::SystemList::iterator

  This class provides an iterator that can read or modify any object of the list.
- class BGAPI2::InterfaceList

  This class is used to search and list interface objects.
- class BGAPI2::InterfaceList::iterator

  This class provides a iterator that can read or modify any element in the list.
- class BGAPI2::DeviceList

  This class is used to discover and list device objects.
- class BGAPI2::DeviceList::iterator

  This class provides a iterator that can read or modify any element in the list.
- class BGAPI2::DataStreamList

  This class is used to discover and list data stream objects.
- class BGAPI2::DataStreamList::iterator

  This class provides an iterator that can read or modify any element in the list.
- class BGAPI2::BufferList

  This class is used for discovery and listing of buffer objects.
- class BGAPI2::BufferList::iterator

  This class provides a iterator that can read or modify any element in the list.
- class BGAPI2::Trace

  The class Trace offers the possibility to monitor the program flow and detect errors. This class belongs to the additional classes.
- class BGAPI2::Node

  The class Node represent one feature from the provided node list based on the underlying XML definition.
- class BGAPI2::NodeMap

  The class NodeMap represents a collection of Node objects based on the underlying XML definition file. This class supports two representation forms, a unstructured list and a tree structure.
- class BGAPI2::NodeMap::iterator

  This class provides a iterator that can read or modify any element in the list.
- class BGAPI2::INode

  The class INode act as base for of the main classes and provided the access to the node objects (features).
- class BGAPI2::Events::EventControl

  The class EventControl provided access to custom events as well as the event mode.
- class BGAPI2::Events::PnPEvent

  The class PnPEvent represented a plug'n play event and provides access to the event information.
- class BGAPI2::Events::InterfaceEventControl

  The class InterfaceEventControl provides access to interface specific events, e.g. plug'n play event.
- class BGAPI2::Events::DeviceEvent

  This class represents an device event which was received from the host. Use this class to get event information.
- class BGAPI2::Events::DeviceEventControl

  The class DeviceEventControl provides access to standard events transmitted from the device.
- class BGAPI2::Events::DataStreamEventControl

  The class DataStreamEventControl provides the new buffer event which is used for fetching images.

- class BGAPI2::Buffer

  This class realizes the data access to the memory. It contains information about the received data (e.g. image size, pixel format). This class belongs to the BGAPI2 main classes.
- class BGAPI2::DataStream

  This class represents a physical data stream from the device and it is responsible for the buffer handling. This class belongs to the BGAPI2 main classes.
- class BGAPI2::Device

  The class Device is used to retrieve information (e.g. model, manufacturer, access modes) of the device (camera) and also to control the device. This class belongs to the BGAPI2 main classes.
- class BGAPI2::Interface

  The class Interface represents a physical interface, e.g. GEV or a logical interface, such as USB and belongs to the BGAPI2 main classes.
- class BGAPI2::System

  The class System is the abstraction of a Producer and belongs to the BGAPI2 main classes.
- class BGAPI2::Image

  The class Image provides the ability of image transformation. This class belongs to the additional classes.
- class BGAPI2::ImageProcessor

  The task of the class ImageProcessor are the creation of image objects and the transformation of pixel formats.
- class BGAPI2::Polarizer

  Provides functionality to calculate several different formats out of the raw polarized camera data.
- class BGAPI2::Polarizer::formatlist

  This class provides a iterator that can read or modify any element in the list.
- class BGAPI2::Polarizer::formatlist::const_iterator

  This class provides a iterator that can read or modify any element in the list.
- class BGAPI2::Exceptions::IException

  This class is responable for the exception handling and represents the parent class of all exception classes.
- class BGAPI2::Exceptions::ErrorException

  General purpose exception.
- class BGAPI2::Exceptions::NotInitializedException

  The requested object is not initialized/opened.
- class BGAPI2::Exceptions::NotImplementedException

  The requested function/feature is not implemented.
- class BGAPI2::Exceptions::ResourceInUseException

  The requested object is already used.
- class BGAPI2::Exceptions::AccessDeniedException

  The requested operation is not allowed/possible, e.g. lose the connection to the device.
- class BGAPI2::Exceptions::InvalidHandleException

  (Given handle does not support the operation.)
- class BGAPI2::Exceptions::NoDataException

  An event contains no event data.
- class BGAPI2::Exceptions::InvalidParameterException

  One of the parameter given was not valid or out of range.
- class BGAPI2::Exceptions::AbortException

  An operation has been aborted before it could be completed.
- class BGAPI2::Exceptions::InvalidBufferException

  Invalid buffer is used. The used Buffer object is not valid.
- class BGAPI2::Exceptions::NotAvailableException

  The requested resource or information is not available at a given time in a current state.
- class BGAPI2::Exceptions::ObjectInvalidException

  The referenced object is not a valid object of BGAPI2.
- class BGAPI2::Exceptions::LowLevelException

  Exception thrown by deeper software layers like GenTL producer.

Namespaces

- BGAPI2

    The global namespace of Baumer GAPI SDK 2.
- BGAPI2::Events

    The namespace Events consists of classes which belongs to the event interface.
- BGAPI2::Exceptions

    The namespace Exceptions consists of classes which are responsible for exception handling.

Typedefs

- typedef void(BGAPI2CALL ∗ BGAPI2::Events::PnPEventHandler) (void ∗callBackOwner, PnPEvent ∗pBuffer)

    Function pointer for pnp event notification, which points to a user defined handler.
- typedef void(BGAPI2CALL ∗ BGAPI2::Events::DeviceEventHandler) (void ∗callBackOwner, DeviceEvent ∗pDeviceEvent)

    Function pointer for device event notification, which points to a user defined handler.
- typedef void(BGAPI2CALL ∗ BGAPI2::Events::NewBufferEventHandler) (void ∗callBackOwner, Buffer ∗pBuffer)

    Function pointer for buffer notification, which points to a user defined handler.

Enumerations

- enum BGAPI2::Events::EventMode { BGAPI2::Events::EVENTMODE_UNREGISTERED = 0, BGAPI2↩ ::Events::EVENTMODE_POLLING = 1, BGAPI2::Events::EVENTMODE_EVENT_HANDLER = 2 }

    Enumeration, which defines kinds of event modes.
- enum BGAPI2::Events::PnPType { BGAPI2::Events::PNPTYPE_DEVICEREMOVED = 0, BGAPI2::↩ Events::PNPTYPE_DEVICEADDED = 1 }

    Enumeration, which defines kinds of PnP events.

Index