

---

# **PySpin API Reference**

***Release 3.1***

**FLIR Integrated Imaging Solutions, Inc**

**Apr 05, 2023**



## CONTENTS:

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Software Licensing Information</b>	<b>3</b>
<b>3</b>	<b>Event Classes</b>	<b>5</b>
3.1	PySpin.DeviceArrivalEventHandler . . . . .	5
3.2	PySpin.DeviceEventHandler . . . . .	6
3.3	PySpin.DeviceRemovalEventHandler . . . . .	6
3.4	PySpin.EventHandler . . . . .	6
3.5	PySpin.ImageEventHandler . . . . .	7
3.6	PySpin.ImageListEventHandler . . . . .	7
3.7	PySpin.InterfaceArrivalEventHandler . . . . .	7
3.8	PySpin.InterfaceEventHandler . . . . .	7
3.9	PySpin.InterfaceRemovalEventHandler . . . . .	8
3.10	PySpin.LoggingEventHandler . . . . .	8
3.11	PySpin.LoggingEventDataPtr . . . . .	8
3.12	PySpin.SystemEventHandler . . . . .	9
<b>4</b>	<b>PySpin Classes</b>	<b>11</b>
4.1	PySpin.CBasePtr . . . . .	11
4.2	PySpin.Camera . . . . .	12
4.3	PySpin.CameraBase . . . . .	34
4.4	PySpin.CameraList . . . . .	38
4.5	PySpin.CameraPtr . . . . .	40
4.6	PySpin.ChannelStatistics . . . . .	41
4.7	PySpin.ChunkData . . . . .	41
4.8	PySpin.Image . . . . .	45
4.9	PySpin.ImageList . . . . .	54
4.10	PySpin.ImageProcessor . . . . .	55
4.11	PySpin.ImagePtr . . . . .	56
4.12	PySpin.IInterface . . . . .	56
4.13	PySpin.InterfaceList . . . . .	57
4.14	PySpin.InterfacePtr . . . . .	58
4.15	PySpin.SpinnakerException . . . . .	58
4.16	PySpin.SpinVideo . . . . .	59
4.17	PySpin.System . . . . .	60
4.18	PySpin.SystemPtr . . . . .	64
<b>5</b>	<b>QuickSpin classes</b>	<b>65</b>
5.1	PySpin.TransportLayerDevice . . . . .	65

5.2	PySpin.TransportLayerInterface . . . . .	67
5.3	PySpin.TransportLayerStream . . . . .	68
<b>6</b>	<b>PySpin Module</b>	<b>71</b>
	<b>Python Module Index</b>	<b>73</b>
	<b>Index</b>	<b>75</b>

## **INTRODUCTION**

PySpin is a wrapper for FLIR Integrated Imaging Solutions' Spinnaker library.

FLIR Integrated Imaging Solutions' website is located at <https://www.flir.com/iis/machine-vision>.

The PySpin Python extension provides a common software interface to control and acquire images from FLIR USB 3.0, GigE, and USB 2.0 cameras using the same API.



## SOFTWARE LICENSING INFORMATION

Component	License
PySpin	Copyright (c) 2001-2023 FLIR Systems, Inc. All Rights Reserved. This software is the confidential and proprietary information of FLIR Integrated Imaging Solutions, Inc. ("Confidential Information"). You shall not disclose such Confidential Information and shall use it only in accordance with the terms of the license agreement you entered into with FLIR Integrated Imaging Solutions, Inc. (FLIR). FLIR MAKES NO REPRESENTATIONS OR WARRANTIES ABOUT THE SUITABILITY OF THE SOFTWARE, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. FLIR SHALL NOT BE LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR ITS DERIVATIVES.
GenICam	GenICam License <a href="http://www.emva.org/wp-content/uploads/GenICam_License_20140921.pdf">http://www.emva.org/wp-content/uploads/GenICam_License_20140921.pdf</a>
AdapterList	The Code Project Open License (CPOL) <a href="http://www.codeproject.com/info/cpol10.aspx">http://www.codeproject.com/info/cpol10.aspx</a>
Boost	Boost Software License <a href="http://www.boost.org/users/license.html">http://www.boost.org/users/license.html</a>
FFMPEG	LGPLv2.1 License <a href="https://www.ffmpeg.org/legal.html">https://www.ffmpeg.org/legal.html</a>
FreeImage	FreeImage public license <a href="http://freeimage.sourceforge.net/freeimage-license.txt">http://freeimage.sourceforge.net/freeimage-license.txt</a>
Libusb	LGPLv2. License <a href="http://www.gnu.org/licenses/old-licenses/lgpl-2.1.txt">http://www.gnu.org/licenses/old-licenses/lgpl-2.1.txt</a>
Libraw394	LGPLv2.0 License <a href="http://www.gnu.org/licenses/old-licenses/lgpl-2.0.txt">http://www.gnu.org/licenses/old-licenses/lgpl-2.0.txt</a>
log4Net	Apache license 2.0 <a href="https://logging.apache.org/log4net/license.html">https://logging.apache.org/log4net/license.html</a>
log4Cpp	LGPL License <a href="http://log4cpp.sourceforge.net/#license">http://log4cpp.sourceforge.net/#license</a>
Work with Bitmaps Faster in C#	The Code Project Open License (CPOL) 1.02 <a href="http://www.codeproject.com/info/cpol10.aspx">http://www.codeproject.com/info/cpol10.aspx</a>
GUI ListView Improvements	WP:CC_BY-SA License <a href="https://goo.gl/a9I9yA">https://goo.gl/a9I9yA</a>





## EVENT CLASSES

- *PySpin.DeviceArrivalEventHandler*
- *PySpin.DeviceEventHandler*
- *PySpin.DeviceRemovalEventHandler*
- *PySpin.EventHandler*
- *PySpin.ImageEventHandler*
- *PySpin.ImageListEventHandler*
- *PySpin.InterfaceArrivalEventHandler*
- *PySpin.InterfaceEventHandler*
- *PySpin.InterfaceRemovalEventHandler*
- *PySpin.LoggingEventHandler*
- *PySpin.LoggingEventDataPtr*
- *PySpin.SystemEventHandler*

### 3.1 PySpin.DeviceArrivalEventHandler

**class** `PySpin.DeviceArrivalEventHandler`

Proxy of C++ Spinnaker::DeviceArrivalEventHandler class.

**OnDeviceArrival** (*self*, *pCamera*)

**Parameters**

**pCamera** (*Spinnaker::CameraPtr*) –

**property thisown**

The membership flag

## 3.2 PySpin.DeviceEventHandler

**class** PySpin.DeviceEventHandler

Proxy of C++ Spinnaker::DeviceEventHandler class.

**GetDeviceEventId**(*self*) → uint64\_t

**GetDeviceEventName**(*self*) → gcstring

**OnDeviceEvent**(*self*, *eventName*)

**Parameters**

**eventName** (*Spinnaker::GenICam::gcstring*) –

**property thisown**

The membership flag

## 3.3 PySpin.DeviceRemovalEventHandler

**class** PySpin.DeviceRemovalEventHandler

Proxy of C++ Spinnaker::DeviceRemovalEventHandler class.

**OnDeviceRemoval**(*self*, *pCamera*)

**Parameters**

**pCamera** (*Spinnaker::CameraPtr*) –

**property thisown**

The membership flag

## 3.4 PySpin.EventHandler

**class** PySpin.EventHandler(\*args, \*\*kwargs)

Proxy of C++ Spinnaker::EventHandler class.

**GetEventPayloadData**(*self*) → PyObject \*

**GetEventPayloadDataSize**(*self*) → size\_t const

**GetEventType**(*self*) → Spinnaker::EventType

**SetEventType**(*self*, *eventType*)

**Parameters**

**eventType** (*enum Spinnaker::EventType*) –

**property thisown**

The membership flag

### 3.5 PySpin.ImageEventHandler

**class** PySpin.ImageEventHandler

Proxy of C++ Spinnaker::ImageEventHandler class.

**OnImageEvent**(*self*, *image*)

**Parameters**

**image** (*Spinnaker::ImagePtr*) –

**property thisown**

The membership flag

### 3.6 PySpin.ImageListEventHandler

**class** PySpin.ImageListEventHandler

Proxy of C++ Spinnaker::ImageListEventHandler class.

**OnImageListEvent**(*self*, *imageList*)

**Parameters**

**imageList** (*Spinnaker::ImageList*) –

**property thisown**

The membership flag

### 3.7 PySpin.InterfaceArrivalEventHandler

**class** PySpin.InterfaceArrivalEventHandler

Proxy of C++ Spinnaker::InterfaceArrivalEventHandler class.

**OnInterfaceArrival**(*self*, *pInterface*)

**Parameters**

**pInterface** (*Spinnaker::InterfacePtr*) –

**property thisown**

The membership flag

### 3.8 PySpin.InterfaceEventHandler

**class** PySpin.InterfaceEventHandler

Proxy of C++ Spinnaker::InterfaceEventHandler class.

**OnDeviceArrival**(*self*, *pCamera*)

**Parameters**

**pCamera** (*Spinnaker::CameraPtr*) –

**OnDeviceRemoval**(*self*, *pCamera*)

**Parameters**

**pCamera** (*Spinnaker::CameraPtr*) –

**property thisown**

The membership flag

## 3.9 PySpin.InterfaceRemovalEventHandler

**class** **PySpin.InterfaceRemovalEventHandler**

Proxy of C++ Spinnaker::InterfaceRemovalEventHandler class.

**OnInterfaceRemoval**(*self*, *pInterface*)

**Parameters**

**pInterface** (*Spinnaker::InterfacePtr*) –

**property thisown**

The membership flag

## 3.10 PySpin.LoggingEventHandler

**class** **PySpin.LoggingEventHandler**

Proxy of C++ Spinnaker::LoggingEventHandler class.

**OnLogEvent**(*self*, *eventPtr*)

**Parameters**

**eventPtr** (*Spinnaker::LoggingEventDataPtr*) –

**property thisown**

The membership flag

## 3.11 PySpin.LoggingEventDataPtr

**class** **PySpin.LoggingEventDataPtr**(\*args)

A reference tracked pointer to the LoggingEvent object.

C++ includes: LoggingEventDataPtr.h

**property thisown**

The membership flag

## 3.12 PySpin.SystemEventHandler

**class** PySpin.SystemEventHandler

Proxy of C++ Spinnaker::SystemEventHandler class.

**OnInterfaceArrival**(*self*, *pInterface*)

Parameters

**pInterface** (*Spinnaker::InterfacePtr*) –

**OnInterfaceRemoval**(*self*, *pInterface*)

Parameters

**pInterface** (*Spinnaker::InterfacePtr*) –

**property** **thisown**

The membership flag



## PYSPIN CLASSES

- *PySpin.CBasePtr*
- *PySpin.Camera*
- *PySpin.CameraBase*
- *PySpin.CameraList*
- *PySpin.CameraPtr*
- *PySpin.ChannelStatistics*
- *PySpin.ChunkData*
- *PySpin.Image*
- *PySpin.ImageList*
- *PySpin.ImageProcessor*
- *PySpin.ImagePtr*
- *PySpin.IInterface*
- *PySpin.InterfaceList*
- *PySpin.InterfacePtr*
- *PySpin.SpinnakerException*
- *PySpin.SpinVideo*
- *PySpin.System*
- *PySpin.SystemPtr*

## 4.1 PySpin.CBasePtr

**class** `PySpin.CBasePtr(*args)`

Encapsulates a GenApi pointer dealing with the `dynamic_cast` automatically.

C++ includes: `Pointer.h`

**GetAccessMode**(*self*) → `Spinnaker::GenApi::EAccessMode`

**IsValid**(*self*) → `bool`

`bool Spinnaker::GenApi::CPointer< T, B >::IsValid() const throw ()` true if the pointer is valid

**property thisown**

The membership flag

## 4.2 PySpin.Camera

**class** PySpin.Camera(\*args, \*\*kwargs)

The camera object class.

C++ includes: Camera.h

**property AasRoiEnable**

**property AasRoiHeight**

**property AasRoiOffsetX**

**property AasRoiOffsetY**

**property AasRoiWidth**

**property AcquisitionAbort**

**property AcquisitionArm**

**property AcquisitionBurstFrameCount**

**property AcquisitionFrameCount**

**property AcquisitionFrameRate**

**property AcquisitionFrameRateEnable**

**property AcquisitionLineRate**

**property AcquisitionMode**

**property AcquisitionResultingFrameRate**

**property AcquisitionStart**

**property AcquisitionStatus**

**property AcquisitionStatusSelector**

**property AcquisitionStop**

**property ActionDeviceKey**

**property ActionGroupKey**

**property ActionGroupMask**

**property ActionQueueSize**

**property ActionSelector**

**property ActionUnconditionalMode**



property AdaptiveCompressionEnable

property AdcBitDepth

property AutoAlgorithmSelector

property AutoExposureControlLoopDamping

property AutoExposureControlPriority

property AutoExposureEVCompensation

property AutoExposureExposureTimeLowerLimit

property AutoExposureExposureTimeUpperLimit

property AutoExposureGainLowerLimit

property AutoExposureGainUpperLimit

property AutoExposureGreyValueLowerLimit

property AutoExposureGreyValueUpperLimit

property AutoExposureLightingMode

property AutoExposureMeteringMode

property AutoExposureTargetGreyValue

property AutoExposureTargetGreyValueAuto

property BalanceRatio

property BalanceRatioSelector

property BalanceWhiteAuto

property BalanceWhiteAutoDamping

property BalanceWhiteAutoLowerLimit

property BalanceWhiteAutoProfile

property BalanceWhiteAutoUpperLimit

property BinningHorizontal

property BinningHorizontalMode

property BinningSelector

property BinningVertical

property BinningVerticalMode

property BlackLevel

property BlackLevelAuto

property BlackLevelAutoBalance

property BlackLevelClampingEnable  
property BlackLevelRaw  
property BlackLevelSelector  
property ChunkBlackLevel  
property ChunkBlackLevelSelector  
property ChunkCRC  
property ChunkCompressionMode  
property ChunkCompressionRatio  
property ChunkCounterSelector  
property ChunkCounterValue  
property ChunkEnable  
property ChunkEncoderSelector  
property ChunkEncoderStatus  
property ChunkEncoderValue  
property ChunkExposureEndLineStatusAll  
property ChunkExposureTime  
property ChunkExposureTimeSelector  
property ChunkFrameID  
property ChunkGain  
property ChunkGainSelector  
property ChunkHeight  
property ChunkImage  
property ChunkImageComponent  
property ChunkInferenceBoundingBoxResult  
property ChunkInferenceConfidence  
property ChunkInferenceFrameId  
property ChunkInferenceResult  
property ChunkLinePitch  
property ChunkLineStatusAll  
property ChunkModeActive  
property ChunkOffsetX

property ChunkOffsetY

property ChunkPartSelector

property ChunkPixelDynamicRangeMax

property ChunkPixelDynamicRangeMin

property ChunkPixelFormat

property ChunkRegionID

property ChunkScan3dAxisMax

property ChunkScan3dAxisMin

property ChunkScan3dCoordinateOffset

property ChunkScan3dCoordinateReferenceSelector

property ChunkScan3dCoordinateReferenceValue

property ChunkScan3dCoordinateScale

property ChunkScan3dCoordinateSelector

property ChunkScan3dCoordinateSystem

property ChunkScan3dCoordinateSystemReference

property ChunkScan3dCoordinateTransformSelector

property ChunkScan3dDistanceUnit

property ChunkScan3dInvalidDataFlag

property ChunkScan3dInvalidDataValue

property ChunkScan3dOutputMode

property ChunkScan3dTransformValue

property ChunkScanLineSelector

property ChunkSelector

property ChunkSequencerSetActive

property ChunkSerialData

property ChunkSerialDataLength

property ChunkSerialReceiveOverflow

property ChunkSourceID

property ChunkStreamChannelID

property ChunkTimerSelector

property ChunkTimerValue

property ChunkTimestamp

property ChunkTimestampLatchValue

property ChunkTransferBlockID

property ChunkTransferQueueCurrentBlockCount

property ChunkTransferStreamID

property ChunkWidth

property ClConfiguration

property ClTimeSlotsCount

property ColorTransformationEnable

property ColorTransformationSelector

property ColorTransformationValue

property ColorTransformationValueSelector

property CompressionRatio

property CompressionSaturationPriority

property CounterDelay

property CounterDuration

property CounterEventActivation

property CounterEventSource

property CounterReset

property CounterResetActivation

property CounterResetSource

property CounterSelector

property CounterStatus

property CounterTriggerActivation

property CounterTriggerSource

property CounterValue

property CounterValueAtReset

property CxpConnectionSelector

property CxpConnectionTestErrorCount

property CxpConnectionTestMode

property CxpConnectionTestPacketCount

property CxpLinkConfiguration  
property CxpLinkConfigurationPreferred  
property CxpLinkConfigurationStatus  
property CxpPoCxpAuto  
property CxpPoCxpStatus  
property CxpPoCxpTripReset  
property CxpPoCxpTurnOff  
property DecimationHorizontal  
property DecimationHorizontalMode  
property DecimationSelector  
property DecimationVertical  
property DecimationVerticalMode  
property DefectCorrectStaticEnable  
property DefectCorrectionMode  
property DefectTableApply  
property DefectTableCoordinateX  
property DefectTableCoordinateY  
property DefectTableFactoryRestore  
property DefectTableIndex  
property DefectTablePixelCount  
property DefectTableSave  
property Deinterlacing  
property DeviceCharacterSet  
property DeviceClockFrequency  
property DeviceClockSelector  
property DeviceConnectionSelector  
property DeviceConnectionSpeed  
property DeviceConnectionStatus  
property DeviceEventChannelCount  
property DeviceFamilyName  
property DeviceFeaturePersistenceEnd

property DeviceFeaturePersistenceStart

property DeviceFirmwareVersion

property DeviceGenCPVersionMajor

property DeviceGenCPVersionMinor

property DeviceID

property DeviceIndicatorMode

property DeviceLinkBandwidthReserve

property DeviceLinkCommandTimeout

property DeviceLinkConnectionCount

property DeviceLinkCurrentThroughput

property DeviceLinkHeartbeatMode

property DeviceLinkHeartbeatTimeout

property DeviceLinkSelector

property DeviceLinkSpeed

property DeviceLinkThroughputLimit

property DeviceLinkThroughputLimitMode

property DeviceManifestEntrySelector

property DeviceManifestPrimaryURL

property DeviceManifestSchemaMajorVersion

property DeviceManifestSchemaMinorVersion

property DeviceManifestSecondaryURL

property DeviceManifestXMLMajorVersion

property DeviceManifestXMLMinorVersion

property DeviceManifestXMLSubMinorVersion

property DeviceManufacturerInfo

property DeviceMaxThroughput

property DeviceModelName

property DevicePowerSupplySelector

property DeviceRegistersCheck

property DeviceRegistersEndianness

property DeviceRegistersStreamingEnd

property DeviceRegistersStreamingStart  
property DeviceRegistersValid  
property DeviceReset  
property DeviceSFNCVersionMajor  
property DeviceSFNCVersionMinor  
property DeviceSFNCVersionSubMinor  
property DeviceScanType  
property DeviceSerialNumber  
property DeviceSerialPortBaudRate  
property DeviceSerialPortSelector  
property DeviceStreamChannelCount  
property DeviceStreamChannelEndianness  
property DeviceStreamChannelLink  
property DeviceStreamChannelPacketSize  
property DeviceStreamChannelSelector  
property DeviceStreamChannelType  
property DeviceTLType  
property DeviceTLVersionMajor  
property DeviceTLVersionMinor  
property DeviceTLVersionSubMinor  
property DeviceTapGeometry  
property DeviceTemperature  
property DeviceTemperatureSelector  
property DeviceType  
property DeviceUptime  
property DeviceUserID  
property DeviceVendorName  
property DeviceVersion  
property EncoderDivider  
property EncoderMode  
property EncoderOutputMode

property EncoderReset  
property EncoderResetActivation  
property EncoderResetSource  
property EncoderSelector  
property EncoderSourceA  
property EncoderSourceB  
property EncoderStatus  
property EncoderTimeout  
property EncoderValue  
property EncoderValueAtReset  
property EnumerationCount  
property EventAcquisitionEnd  
property EventAcquisitionEndFrameID  
property EventAcquisitionEndTimestamp  
property EventAcquisitionError  
property EventAcquisitionErrorFrameID  
property EventAcquisitionErrorTimestamp  
property EventAcquisitionStart  
property EventAcquisitionStartFrameID  
property EventAcquisitionStartTimestamp  
property EventAcquisitionTransferEnd  
property EventAcquisitionTransferEndFrameID  
property EventAcquisitionTransferEndTimestamp  
property EventAcquisitionTransferStart  
property EventAcquisitionTransferStartFrameID  
property EventAcquisitionTransferStartTimestamp  
property EventAcquisitionTrigger  
property EventAcquisitionTriggerFrameID  
property EventAcquisitionTriggerTimestamp  
property EventActionLate  
property EventActionLateFrameID



property EventActionLateTimestamp  
property EventCounter0End  
property EventCounter0EndFrameID  
property EventCounter0EndTimestamp  
property EventCounter0Start  
property EventCounter0StartFrameID  
property EventCounter0StartTimestamp  
property EventCounter1End  
property EventCounter1EndFrameID  
property EventCounter1EndTimestamp  
property EventCounter1Start  
property EventCounter1StartFrameID  
property EventCounter1StartTimestamp  
property EventEncoder0Restarted  
property EventEncoder0RestartedFrameID  
property EventEncoder0RestartedTimestamp  
property EventEncoder0Stopped  
property EventEncoder0StoppedFrameID  
property EventEncoder0StoppedTimestamp  
property EventEncoder1Restarted  
property EventEncoder1RestartedFrameID  
property EventEncoder1RestartedTimestamp  
property EventEncoder1Stopped  
property EventEncoder1StoppedFrameID  
property EventEncoder1StoppedTimestamp  
property EventError  
property EventErrorCode  
property EventErrorFrameID  
property EventErrorTimestamp  
property EventExposureEnd  
property EventExposureEndFrameID

property EventExposureEndTimestamp  
property EventExposureStart  
property EventExposureStartFrameID  
property EventExposureStartTimestamp  
property EventFrameBurstEnd  
property EventFrameBurstEndFrameID  
property EventFrameBurstEndTimestamp  
property EventFrameBurstStart  
property EventFrameBurstStartFrameID  
property EventFrameBurstStartTimestamp  
property EventFrameEnd  
property EventFrameEndFrameID  
property EventFrameEndTimestamp  
property EventFrameStart  
property EventFrameStartFrameID  
property EventFrameStartTimestamp  
property EventFrameTransferEnd  
property EventFrameTransferEndFrameID  
property EventFrameTransferEndTimestamp  
property EventFrameTransferStart  
property EventFrameTransferStartFrameID  
property EventFrameTransferStartTimestamp  
property EventFrameTrigger  
property EventFrameTriggerFrameID  
property EventFrameTriggerTimestamp  
property EventLine0AnyEdge  
property EventLine0AnyEdgeFrameID  
property EventLine0AnyEdgeTimestamp  
property EventLine0FallingEdge  
property EventLine0FallingEdgeFrameID  
property EventLine0FallingEdgeTimestamp

property EventLine0RisingEdge  
property EventLine0RisingEdgeFrameID  
property EventLine0RisingEdgeTimestamp  
property EventLine1AnyEdge  
property EventLine1AnyEdgeFrameID  
property EventLine1AnyEdgeTimestamp  
property EventLine1FallingEdge  
property EventLine1FallingEdgeFrameID  
property EventLine1FallingEdgeTimestamp  
property EventLine1RisingEdge  
property EventLine1RisingEdgeFrameID  
property EventLine1RisingEdgeTimestamp  
property EventLinkSpeedChange  
property EventLinkSpeedChangeFrameID  
property EventLinkSpeedChangeTimestamp  
property EventLinkTrigger0  
property EventLinkTrigger0FrameID  
property EventLinkTrigger0Timestamp  
property EventLinkTrigger1  
property EventLinkTrigger1FrameID  
property EventLinkTrigger1Timestamp  
property EventNotification  
property EventSelector  
property EventSequencerSetChange  
property EventSequencerSetChangeFrameID  
property EventSequencerSetChangeTimestamp  
property EventSerialData  
property EventSerialDataLength  
property EventSerialPortReceive  
property EventSerialPortReceiveTimestamp  
property EventSerialReceiveOverflow

property EventStream@TransferBlockEnd  
property EventStream@TransferBlockEndFrameID  
property EventStream@TransferBlockEndTimestamp  
property EventStream@TransferBlockStart  
property EventStream@TransferBlockStartFrameID  
property EventStream@TransferBlockStartTimestamp  
property EventStream@TransferBlockTrigger  
property EventStream@TransferBlockTriggerFrameID  
property EventStream@TransferBlockTriggerTimestamp  
property EventStream@TransferBurstEnd  
property EventStream@TransferBurstEndFrameID  
property EventStream@TransferBurstEndTimestamp  
property EventStream@TransferBurstStart  
property EventStream@TransferBurstStartFrameID  
property EventStream@TransferBurstStartTimestamp  
property EventStream@TransferEnd  
property EventStream@TransferEndFrameID  
property EventStream@TransferEndTimestamp  
property EventStream@TransferOverflow  
property EventStream@TransferOverflowFrameID  
property EventStream@TransferOverflowTimestamp  
property EventStream@TransferPause  
property EventStream@TransferPauseFrameID  
property EventStream@TransferPauseTimestamp  
property EventStream@TransferResume  
property EventStream@TransferResumeFrameID  
property EventStream@TransferResumeTimestamp  
property EventStream@TransferStart  
property EventStream@TransferStartFrameID  
property EventStream@TransferStartTimestamp  
property EventTest

property EventTestTimestamp  
property EventTimer0End  
property EventTimer0EndFrameID  
property EventTimer0EndTimestamp  
property EventTimer0Start  
property EventTimer0StartFrameID  
property EventTimer0StartTimestamp  
property EventTimer1End  
property EventTimer1EndFrameID  
property EventTimer1EndTimestamp  
property EventTimer1Start  
property EventTimer1StartFrameID  
property EventTimer1StartTimestamp  
property ExposureActiveMode  
property ExposureAuto  
property ExposureMode  
property ExposureTime  
property ExposureTimeMode  
property ExposureTimeSelector  
property FactoryReset  
property FileAccessBuffer  
property FileAccessLength  
property FileAccessOffset  
property FileOpenMode  
property FileOperationExecute  
property FileOperationResult  
property FileOperationSelector  
property FileOperationStatus  
property FileSelector  
property FileSize  
property Gain

property GainAuto

property GainAutoBalance

property GainSelector

property Gamma

property GammaEnable

property GevActiveLinkCount

property GevCCP

property GevCurrentDefaultGateway

property GevCurrentIPAddress

property GevCurrentIPConfigurationDHCP

property GevCurrentIPConfigurationLLA

property GevCurrentIPConfigurationPersistentIP

property GevCurrentPhysicalLinkConfiguration

property GevCurrentSubnetMask

property GevDiscoveryAckDelay

property GevFirstURL

property GevGVCPExtendedStatusCodes

property GevGVCPExtendedStatusCodesSelector

property GevGVCPHeartbeatDisable

property GevGVCPPendingAck

property GevGVCPPendingTimeout

property GevGVSPExtendedIDMode

property GevHeartbeatTimeout

property GevIEEE1588

property GevIEEE1588ClockAccuracy

property GevIEEE1588Mode

property GevIEEE1588Status

property GevIPConfigurationStatus

property GevInterfaceSelector

property GevMACAddress

property GevMCDA

property `GevMCPHostPort`  
property `GevMCRC`  
property `GevMCSP`  
property `GevMCTT`  
property `GevNumberOfInterfaces`  
property `GevPAUSEFrameReception`  
property `GevPAUSEFrameTransmission`  
property `GevPersistentDefaultGateway`  
property `GevPersistentIPAddress`  
property `GevPersistentSubnetMask`  
property `GevPhysicalLinkConfiguration`  
property `GevPrimaryApplicationIPAddress`  
property `GevPrimaryApplicationSocket`  
property `GevPrimaryApplicationSwitchoverKey`  
property `GevSCCFGAllInTransmission`  
property `GevSCCFGExtendedChunkData`  
property `GevSCCFGPacketResendDestination`  
property `GevSCCFGUnconditionalStreaming`  
property `GevSCDA`  
property `GevSCPD`  
property `GevSCPDDirection`  
property `GevSCPHostPort`  
property `GevSCPInterfaceIndex`  
property `GevSCPSBigEndian`  
property `GevSCPSDoNotFragment`  
property `GevSCPSFireTestPacket`  
property `GevSCPSPacketSize`  
property `GevSCSP`  
property `GevSCZoneConfigurationLock`  
property `GevSCZoneCount`  
property `GevSCZoneDirectionAll`

property `GevSecondURL`

property `GevStreamChannelSelector`

property `GevSupportedOption`

property `GevSupportedOptionSelector`

property `GevTimestampTickFrequency`

property `GuiXmlManifestAddress`

property `Height`

property `HeightMax`

property `ImageComponentEnable`

property `ImageComponentSelector`

property `ImageCompressionBitrate`

property `ImageCompressionJPEGFormatOption`

property `ImageCompressionMode`

property `ImageCompressionQuality`

property `ImageCompressionRateOption`

`Init(self)`  
    void Spinnaker::Camera::Init()

property `IspEnable`

property `LUTEnable`

property `LUTIndex`

property `LUTSelector`

property `LUTValue`

property `LUTValueAll`

property `LineFilterWidth`

property `LineFormat`

property `LineInputFilterSelector`

property `LineInverter`

property `LineMode`

property `LinePitch`

property `LineSelector`

property `LineSource`



property LineStatus  
property LineStatusAll  
property LinkErrorCount  
property LinkUptime  
property LogicBlockLUTInputActivation  
property LogicBlockLUTInputSelector  
property LogicBlockLUTInputSource  
property LogicBlockLUTOutputValue  
property LogicBlockLUTOutputValueAll  
property LogicBlockLUTRowIndex  
property LogicBlockLUTSelector  
property LogicBlockSelector  
property MaxDeviceResetTime  
property OffsetX  
property OffsetY  
property PacketResendRequestCount  
property PayloadSize  
property PixelColorFilter  
property PixelDynamicRangeMax  
property PixelDynamicRangeMin  
property PixelFormat  
property PixelFormatInfoID  
property PixelFormatInfoSelector  
property PixelSize  
property PowerSupplyCurrent  
property PowerSupplyVoltage  
property RegionDestination  
property RegionMode  
property RegionSelector  
property ReverseX  
property ReverseY

property RgbTransformLightSource  
property Saturation  
property SaturationEnable  
property Scan3dAxisMax  
property Scan3dAxisMin  
property Scan3dCoordinateOffset  
property Scan3dCoordinateReferenceSelector  
property Scan3dCoordinateReferenceValue  
property Scan3dCoordinateScale  
property Scan3dCoordinateSelector  
property Scan3dCoordinateSystem  
property Scan3dCoordinateSystemReference  
property Scan3dCoordinateTransformSelector  
property Scan3dDistanceUnit  
property Scan3dInvalidDataFlag  
property Scan3dInvalidDataValue  
property Scan3dOutputMode  
property Scan3dTransformValue  
property SensorDescription  
property SensorDigitizationTaps  
property SensorHeight  
property SensorShutterMode  
property SensorTaps  
property SensorWidth  
property SequencerConfigurationMode  
property SequencerConfigurationValid  
property SequencerFeatureEnable  
property SequencerMode  
property SequencerPathSelector  
property SequencerSetActive  
property SequencerSetLoad

property SequencerSetNext  
property SequencerSetSave  
property SequencerSetSelector  
property SequencerSetStart  
property SequencerSetValid  
property SequencerTriggerActivation  
property SequencerTriggerSource  
property SerialPortBaudRate  
property SerialPortDataBits  
property SerialPortParity  
property SerialPortSelector  
property SerialPortSource  
property SerialPortStopBits  
property SerialReceiveFramingErrorCount  
property SerialReceiveParityErrorCount  
property SerialReceiveQueueClear  
property SerialReceiveQueueCurrentCharacterCount  
property SerialReceiveQueueMaxCharacterCount  
property SerialTransmitQueueCurrentCharacterCount  
property SerialTransmitQueueMaxCharacterCount  
property Sharpening  
property SharpeningAuto  
property SharpeningEnable  
property SharpeningThreshold  
property SoftwareSignalPulse  
property SoftwareSignalSelector  
property SourceCount  
property SourceSelector  
property TLParamsLocked  
property Test0001  
property TestEventGenerate

property TestPattern  
property TestPatternGeneratorSelector  
property TestPendingAck  
property TimerDelay  
property TimerDuration  
property TimerReset  
property TimerSelector  
property TimerStatus  
property TimerTriggerActivation  
property TimerTriggerSource  
property TimerValue  
property Timestamp  
property TimestampLatch  
property TimestampLatchValue  
property TimestampReset  
property TransferAbort  
property TransferBlockCount  
property TransferBurstCount  
property TransferComponentSelector  
property TransferControlMode  
property TransferOperationMode  
property TransferPause  
property TransferQueueCurrentBlockCount  
property TransferQueueMaxBlockCount  
property TransferQueueMode  
property TransferQueueOverflowCount  
property TransferResume  
property TransferSelector  
property TransferStart  
property TransferStatus  
property TransferStatusSelector

property TransferStop  
property TransferStreamChannel  
property TransferTriggerActivation  
property TransferTriggerMode  
property TransferTriggerSelector  
property TransferTriggerSource  
property TriggerActivation  
property TriggerDelay  
property TriggerDivider  
property TriggerEventTest  
property TriggerMode  
property TriggerMultiplier  
property TriggerOverlap  
property TriggerSelector  
property TriggerSoftware  
property TriggerSource  
property UserOutputSelector  
property UserOutputValue  
property UserOutputValueAll  
property UserOutputValueAllMask  
property UserSetDefault  
property UserSetFeatureEnable  
property UserSetLoad  
property UserSetSave  
property UserSetSelector  
property V3\_3Enable  
property WhiteClip  
property WhiteClipSelector  
property Width  
property WidthMax  
property aPAUSEMACtrlFramesReceived

**property** `aPAUSEMACtrlFramesTransmitted`

**property** `thisown`

The membership flag

## 4.3 PySpin.CameraBase

**class** `PySpin.CameraBase(*args, **kwargs)`

The base class for the camera object.

C++ includes: CameraBase.h

**BeginAcquisition**(*self*)

`void Spinnaker::CameraBase::BeginAcquisition()`

Starts the image acquisition engine. The camera must be initialized via a call to `Init()` before starting an acquisition.

See: `Init()`

**DeInit**(*self*)

`void Spinnaker::CameraBase::DeInit()`

Disconnect camera port and free GenICam node map and GUI XML. Do not call more functions that access the remote device such as `WritePort/ReadPort` after calling `DeInit()`; Events should also be unregistered before calling camera `DeInit()`. Otherwise an exception will be thrown in the `DeInit()` call and require the user to unregister events before the camera can be re-initialized again.

See: `Init()`

See: `UnregisterEvent(Event & evtToUnregister)`

**DiscoverMaxPacketSize**(*self*) → unsigned int

`unsigned int Spinnaker::CameraBase::DiscoverMaxPacketSize()`

Returns the largest packet size that can be safely used on the interface that device is connected to

The maximum packet size returned.

**EndAcquisition**(*self*)

`void Spinnaker::CameraBase::EndAcquisition()`

Stops the image acquisition engine. If `EndAcquisition()` is called without a prior call to `BeginAcquisition()` an error message “Camera is not started” will be thrown. All Images that were acquired using `GetNextImage()` need to be released first using `image->Release()` before calling `EndAcquisition()`. All buffers in the input pool and output queue will be discarded when `EndAcquisition()` is called.

See: `Init()`

See: `BeginAcquisition()`

See: `GetNextImage( grabTimeout )`

See: `Image::Release()`

**ForceIP**(*self*)

**GetAccessMode**(*self*) → Spinnaker::GenApi::EAccessMode

GenApi::EAccessMode Spinnaker::CameraBase::GetAccessMode() const

Returns the access mode that the software has on the Camera. The camera does not need to be initialized before calling this function.

See: Init()

An enumeration value indicating the access mode

**GetBufferOwnership**(*self*) → Spinnaker::BufferOwnership

**GetGuiXml**(*self*) → gcstring

GenICam::gcstring Spinnaker::CameraBase::GetGuiXml() const

Returns the GUI XML that can be passed into the Spinnaker GUI framework

GenICam::gcstring that represents the uncompressed GUI XML file

**GetNextImage**(*self*, *grabTimeout*=EVENT\_TIMEOUT\_INFINITE, *streamIndex*=0) → *ImagePtr*

#### Parameters

- **grabTimeout** (a 64bit value that represents a timeout in milliseconds) –
- **streamIndex** (uint64\_t) –
- **ImagePtr** –
- **Spinnaker::CameraBase::GetNextImage(uint64\_t –**
- **grabTimeout=EVENT\_TIMEOUT\_INFINITE –**
- **streamID=0)** (uint64\_t) –
- **This** (Gets the next image that was received by the transport layer.) –
- **cameras** (function will block indefinitely until an image arrives. Most) –
- **camera** (support one stream so the default streamID is 0 but if a) –
- **select** (supports multiple streams the user can input the streamID to) –
- **images** (from which stream to grab) –
- **See** (EndAcquisition()) –
- **See** –
- **See** –
- **Parameters** –
- ----- –
- **grabTimeout** –
- **streamID** (The stream to grab the image.) –
- **object** (pointer to an Image) –

**GetNextImageSync**(*self*, *grabTimeout*=EVENT\_TIMEOUT\_INFINITE) → *ImageList*

**Parameters**

**grabTimeout** (*uint64\_t*) –

**GetNodeMap**(*self*) → *INodeMap*

GenApi::INodeMap& Spinnaker::CameraBase::GetNodeMap() const

Gets a reference to the node map that is generated from a GenICam XML file. The camera must be initialized by a call to Init() first before a node map reference can be successfully acquired.

See: Init()

A reference to the INodeMap.

**GetNumDataStreams**(*self*) → unsigned int

unsigned int Spinnaker::CameraBase::GetNumDataStreams()

Returns the number of streams that a device supports.

The number of data streams

**GetNumImagesInUse**(*self*) → unsigned int

unsigned int Spinnaker::CameraBase::GetNumImagesInUse()

Returns the number of images that are currently in use. Each of the images that are currently in use must be cleaned up with a call to image->Release() before calling system->ReleaseInstance().

The number of images that needs to be cleaned up.

**GetTLDeviceNodeMap**(*self*) → *INodeMap*

GenApi::INodeMap& Spinnaker::CameraBase::GetTLDeviceNodeMap() const

Gets a reference to the node map that is generated from a GenICam XML file for the GenTL Device module. The camera does not need to be initialized before acquiring this node map.

A reference to the INodeMap.

**GetTLStreamNodeMap**(*self*, *streamIndex*=0) → *INodeMap*

**Parameters**

- **streamIndex** (*uint64\_t*) –

- **const** (GenApi::INodeMap& Spinnaker::CameraBase::GetTLStreamNodeMap()) –

- **XML** (Gets a reference to the node map that is generated from a GenICam) –

- **be** (file for the GenTL Stream module. The camera does not need to) –

- **map**. (initialized before acquiring this node) –

- **INodeMap**. (A reference to the) –

**GetUniqueID**(*self*) → *gcstring*

GenICam::gcstring Spinnaker::CameraBase::GetUniqueID()

This returns a unique id string that identifies the camera. This is the camera serial number.

string that uniquely identifies the camera (serial number)

**GetUserBufferCount**(*self*) → *uint64\_t*



**GetUserBufferSize**(*self*) → uint64\_t

**GetUserBufferTotalSize**(*self*) → uint64\_t

**Init**(*self*)

void Spinnaker::CameraBase::Init()

Connect to camera, retrieve XML and generate node map. This function needs to be called before any camera related API calls such as BeginAcquisition(), EndAcquisition(), GetNodeMap(), GetNextImage().

See: BeginAcquisition()

See: EndAcquisition()

See: GetNodeMap()

See: GetNextImage()

**IsInitialized**(*self*) → bool

bool Spinnaker::CameraBase::IsInitialized()

Checks if camera is initialized. This function needs to return true in order to retrieve a valid NodeMap from the GetNodeMap() call.

See: GetNodeMap()

If camera is initialized or not

**IsStreaming**(*self*) → bool

bool Spinnaker::CameraBase::IsStreaming() const

Returns true if the camera is currently streaming or false if it is not.

See: Init()

returns true if camera is streaming and false otherwise.

**IsValid**(*self*) → bool

bool Spinnaker::CameraBase::IsValid()

Checks a flag to determine if camera is still valid for use.

If camera is valid or not

Note that CameraPtr and CameraBase both define an IsValid() function. In order to determine the validity of the camera using a CameraPtr, user must first call get() to retrieve the CameraBase object.

**RegisterEventHandler**(*self*, *evtHandlerToRegister*)

#### Parameters

- **evtHandlerToRegister** (*Spinnaker::ImageEventHandler &*) –
- **RegisterEventHandler**(*self* –
- **evtHandlerToRegister** –
- **eventName**) –
- **evtHandlerToRegister** –
- **eventName** (*Spinnaker::GenICam::gcstring const &*) –
- **RegisterEventHandler**(*self* –
- **evtHandlerToRegister** –

- **streamIndex**) –
- **evtHandlerToRegister** –
- **streamIndex** (*uint64\_t*) –

**SetBufferOwnership**(*self, mode*)

**Parameters**

**mode** (*enum Spinnaker::BufferOwnership const*) –

**SetUserBuffers**(*self, pMemBuffers, totalSize*)

**Parameters**

- **pMemBuffers** (*void \*const*) –
- **totalSize** (*uint64\_t*) –
- **SetUserBuffers**(*self* –
- **ppMemBuffers** (*void \*\*const*) –
- **bufferCount** (*uint64\_t const*) –
- **bufferSize**) –
- **ppMemBuffers** –
- **bufferCount** –
- **bufferSize** (*uint64\_t const*) –

**UnregisterEventHandler**(*self, evtHandlerToUnregister*)

**Parameters**

**evtHandlerToUnregister** (*Spinnaker::EventHandler &*) –

**property thisown**

The membership flag

## 4.4 PySpin.CameraList

**class** **PySpin.CameraList**(\*args)

Used to hold a list of camera objects.

C++ includes: CameraList.h

**Add**(*self, camera*)

**Parameters**

**camera** (*Spinnaker::CameraPtr*) –

**Append**(*self, list*)

**Parameters**

- **list** (*Spinnaker::CameraList const &*) –
- **void** –
- **&otherList**) (*Spinnaker::CameraList::Append*([CameraList](#)) –
- **list.** (*Appends a camera list to the current*) –

- **Parameters** –
- ----- –
- **otherList** (*The other list to append to this list*) –

**Clear**(*self*)

void Spinnaker::CameraList::Clear()

Clears the list of cameras and destroys their corresponding reference counted objects. This is necessary in order to clean up the parent interface. It is important that the camera list is destroyed or is cleared before calling system->ReleaseInstance() or else the call to system->ReleaseInstance() will result in an error message thrown that a reference to the camera is still held.

See: System:ReleaseInstance()

**GetByDeviceID**(*self, deviceID*) → *CameraPtr*

**Parameters**

**deviceID** (*std::string*) –

**GetByIndex**(*self, index*) → *CameraPtr*

**Parameters**

- **index** (*The index at which to retrieve the camera object*) –
- **CameraPtr** –
- **const** (*Spinnaker::CameraList::GetByIndex(int index)*) –
- **"index"**. (*Returns a pointer to a camera object at the*) –
- **Parameters** –
- ----- –
- **index** –
- **object.** (*A pointer to an camera*) –

**GetBySerial**(*self, serialNumber*) → *CameraPtr*

**Parameters**

- **serialNumber** (*The serial number of the camera object to retrieve*) –
- **CameraPtr** –
- **const** (*Spinnaker::CameraList::GetBySerial(std::string serialNumber)*) –
- **number.** (*Returns a pointer to a camera object with the specified serial*) –
- **Parameters** –
- ----- –
- **serialNumber** –
- **object.** (*A pointer to an camera*) –

**GetSize**(*self*) → unsigned int

int Spinnaker::CameraList::GetSize() const

Returns the size of the camera list. The size is the number of Camera objects stored in the list.

An integer that represents the list size.

**Remove**(*self*, *camera*)

**Parameters**

**camera** (*Spinnaker::CameraPtr*) –

**RemoveByDeviceID**(*self*, *deviceID*)

**Parameters**

**deviceID** (*std::string*) –

**RemoveByIndex**(*self*, *index*)

**Parameters**

- **index** (*The index at which to remove the Camera object*) –
- **void** –
- **index** (*Spinnaker::CameraList::RemoveByIndex(int)*) –
- **reference** (*Removes a camera at "index" and destroys its corresponding*) –
- **object.** (*counted*) –
- **Parameters** –
- ----- –
- **index** –

**RemoveBySerial**(*self*, *serialNumber*)

**Parameters**

- **serialNumber** (*The serial number of the Camera object to remove*) –
- **void** –
- **serialNumber** (*Spinnaker::CameraList::RemoveBySerial(std::string)*) –
- **its** (*Removes a camera using its serial number and destroys*) –
- **object.** (*corresponding reference counted*) –
- **Parameters** –
- ----- –
- **serialNumber** –

**property thisown**

The membership flag

## 4.5 PySpin.CameraPtr

**class** **PySpin.CameraPtr**(\*args)

A reference tracked pointer to a camera object.

C++ includes: CameraPtr.h

**property thisown**

The membership flag

## 4.6 PySpin.ChannelStatistics

**class** PySpin.ChannelStatistics(*image, channel*)

Class used to store statistics (as properties) for one channel of an image. Properties:

- **channel**: The image channel that the statistics are based on (as an int).
- **range\_min**: The smallest possible pixel value.
- **range\_max**: The largest possible pixel value.
- **pixel\_value\_min**: The smallest pixel value in the current channel.
- **pixel\_value\_max**: The largest pixel value in the current channel.
- **num\_pixel\_values**: The total number of pixel values in the current channel.
- **pixel\_value\_mean**: The average pixel value in the current channel.
- **histogram**: NumPy array representing the histogram of the current channel.

**property channel**

**property histogram**

**property num\_pixel\_values**

**property pixel\_value\_max**

**property pixel\_value\_mean**

**property pixel\_value\_min**

**property range\_max**

**property range\_min**

**property thisown**

The membership flag

## 4.7 PySpin.ChunkData

**class** PySpin.ChunkData(\*args)

The chunk data which contains additional information about an image.

C++ includes: ChunkData.h

**GetBlackLevel**(*self*) → float64\_t

float64\_t Spinnaker::ChunkData::GetBlackLevel() const

Description: Returns the black level used to capture the image included in the payload. Visibility: Expert

**GetCRC**(*self*) → int64\_t

**GetCompressionMode**(*self*) → int64\_t

**GetCompressionRatio**(*self*) → float64\_t

**GetCounterValue**(*self*) → int64\_t

int64\_t Spinnaker::ChunkData::GetCounterValue() const

Description: Returns the value of the selected Chunk counter at the time of the FrameStart event. Visibility: Expert

**GetEncoderValue**(*self*) → int64\_t

int64\_t Spinnaker::ChunkData::GetEncoderValue() const

Description: Returns the counter's value of the selected Encoder at the time of the FrameStart in area scan mode or the counter's value at the time of the LineStart selected by ChunkScanLineSelector in LineScan mode. Visibility: Expert

**GetExposureEndLineStatusAll**(*self*) → int64\_t

**GetExposureTime**(*self*) → float64\_t

float64\_t Spinnaker::ChunkData::GetExposureTime() const

Description: Returns the exposure time used to capture the image. Visibility: Expert

**GetFrameID**(*self*) → int64\_t

int64\_t Spinnaker::ChunkData::GetFrameID() const

Description: Returns the unique Identifier of the frame (or image) included in the payload. Visibility: Expert

**GetGain**(*self*) → float64\_t

float64\_t Spinnaker::ChunkData::GetGain() const

Description: Returns the gain used to capture the image. Visibility: Expert

**GetHeight**(*self*) → int64\_t

int64\_t Spinnaker::ChunkData::GetHeight() const

Description: Returns the Height of the image included in the payload. Visibility: Expert

**GetImage**(*self*) → int64\_t

**GetInferenceBoundingBoxResult**(*self*) → InferenceBoundingBoxResult

**GetInferenceConfidence**(*self*) → float64\_t

**GetInferenceFrameId**(*self*) → int64\_t

**GetInferenceResult**(*self*) → int64\_t

**GetLinePitch**(*self*) → int64\_t

int64\_t Spinnaker::ChunkData::GetLinePitch() const

Description: Returns the LinePitch of the image included in the payload. Visibility: Expert

**GetLineStatusAll**(*self*) → int64\_t

int64\_t Spinnaker::ChunkData::GetLineStatusAll() const

Description: Returns the status of all the I/O lines at the time of the FrameStart internal event. Visibility: Expert

**GetOffsetX**(*self*) → int64\_t

int64\_t Spinnaker::ChunkData::GetOffsetX() const

Description: Returns the OffsetX of the image included in the payload. Visibility: Expert

**GetOffsetY**(*self*) → int64\_t

int64\_t Spinnaker::ChunkData::GetOffsetY() const

Description: Returns the OffsetY of the image included in the payload. Visibility: Expert

**GetPartSelector**(*self*) → int64\_t

int64\_t Spinnaker::ChunkData::GetPartSelector() const

Description: Selects the part to access in chunk data in a multipart transmission. Visibility: Expert

**GetPixelDynamicRangeMax**(*self*) → int64\_t

int64\_t Spinnaker::ChunkData::GetPixelDynamicRangeMax() const

Description: Returns the maximum value of dynamic range of the image included in the payload. Visibility: Expert

**GetPixelDynamicRangeMin**(*self*) → int64\_t

int64\_t Spinnaker::ChunkData::GetPixelDynamicRangeMin() const

Description: Returns the minimum value of dynamic range of the image included in the payload. Visibility: Expert

**GetScan3dAxisMax**(*self*) → float64\_t

float64\_t Spinnaker::ChunkData::GetScan3dAxisMax() const

Description: Returns the Maximum Axis value for the selected coordinate axis of the image included in the payload. Visibility: Expert

**GetScan3dAxisMin**(*self*) → float64\_t

float64\_t Spinnaker::ChunkData::GetScan3dAxisMin() const

Description: Returns the Minimum Axis value for the selected coordinate axis of the image included in the payload. Visibility: Expert

**GetScan3dCoordinateOffset**(*self*) → float64\_t

float64\_t Spinnaker::ChunkData::GetScan3dCoordinateOffset() const

Description: Returns the Offset for the selected coordinate axis of the image included in the payload. Visibility: Expert

**GetScan3dCoordinateReferenceValue**(*self*) → float64\_t

float64\_t Spinnaker::ChunkData::GetScan3dCoordinateReferenceValue() const

Description: Reads the value of a position or pose coordinate for the anchor or transformed coordinate systems relative to the reference point. Visibility: Expert

**GetScan3dCoordinateScale**(*self*) → float64\_t

float64\_t Spinnaker::ChunkData::GetScan3dCoordinateScale() const

Description: Returns the Scale for the selected coordinate axis of the image included in the payload. Visibility: Expert

**GetScan3dInvalidDataValue**(*self*) → float64\_t

float64\_t Spinnaker::ChunkData::GetScan3dInvalidDataValue() const

Description: Returns the Invalid Data Value used for the image included in the payload. Visibility: Expert

**GetScan3dTransformValue**(*self*) → float64\_t

float64\_t Spinnaker::ChunkData::GetScan3dTransformValue() const

Description: Returns the transform value. Visibility: Expert

**GetScanLineSelector**(*self*) → int64\_t

int64\_t Spinnaker::ChunkData::GetScanLineSelector() const

Description: Index for vector representation of one chunk value per line in an image. Visibility: Expert

**GetSequencerSetActive**(*self*) → int64\_t

int64\_t Spinnaker::ChunkData::GetSequencerSetActive() const

Description: Return the index of the active set of the running sequencer included in the payload. Visibility: Expert

**GetSerialDataLength**(*self*) → int64\_t

**GetStreamChannelID**(*self*) → int64\_t

int64\_t Spinnaker::ChunkData::GetStreamChannelID() const

Description: Returns identifier of the stream channel used to carry the block. Visibility: Expert

**GetTimerValue**(*self*) → float64\_t

float64\_t Spinnaker::ChunkData::GetTimerValue() const

Description: Returns the value of the selected Timer at the time of the FrameStart internal event. Visibility: Expert

**GetTimestamp**(*self*) → int64\_t

int64\_t Spinnaker::ChunkData::GetTimestamp() const

Description: Returns the Timestamp of the image included in the payload at the time of the FrameStart internal event. Visibility: Expert

**GetTimestampLatchValue**(*self*) → int64\_t

int64\_t Spinnaker::ChunkData::GetTimestampLatchValue() const

Description: Returns the last Timestamp latched with the TimestampLatch command. Visibility: Expert

**GetTransferBlockID**(*self*) → int64\_t

int64\_t Spinnaker::ChunkData::GetTransferBlockID() const

Description: Returns the unique identifier of the transfer block used to transport the payload. Visibility: Expert

**GetTransferQueueCurrentBlockCount**(*self*) → int64\_t

int64\_t Spinnaker::ChunkData::GetTransferQueueCurrentBlockCount() const

Description: Returns the current number of blocks in the transfer queue. Visibility: Expert

**GetWidth**(*self*) → int64\_t

int64\_t Spinnaker::ChunkData::GetWidth() const

Description: Returns the Width of the image included in the payload. Visibility: Expert

**SetChunks**(*self*, *pNodeMap*)

#### Parameters

- **pNodeMap** (*Spinnaker::GenApi::INodeMap &*) –
- **void** –
- **&pNodeMap** (*Spinnaker::ChunkData::SetChunks(GenApi::INodeMap)*) –

**property thisown**

The membership flag



## 4.8 PySpin.Image

**class** PySpin.**Image**(\*args, \*\*kwargs)

The image object class.

C++ includes: Image.h

**CheckCRC**(*self*) → bool

bool Spinnaker::Image::CheckCRC() const

Checks if the computed checksum matches with chunk data's ImageCRC

Returns true if computed checksum matches with the chunk data's CRC and false otherwise.

**static Create**() → *ImagePtr*

**static Create**(*image*) → *ImagePtr*

### Parameters

- **image** (*Spinnaker::ImagePtr const*) –
- **Create**(*width* –
- **height** (*size\_t*) –
- **offsetX** (*size\_t*) –
- **offsetY** (*size\_t*) –
- **pixelFormat** (*enum Spinnaker::PixelFormatEnums*) –
- **ImagePtr** (*copied from another*) –
- **width** (*or using*) –
- **height** –
- **offsetX** –
- **offsetY** –
- **pixelFormat** –
- **pData** (*void \**) –
- **Create**(*width* –
- **height** –
- **offsetX** –
- **offsetY** –
- **pixelFormat** –
- **pData** –
- **dataPayloadType** (*enum Spinnaker::TLPayloadType*) –
- **ImagePtr** –
- **width** –
- **height** –
- **offsetX** –
- **offsetY** –

- **pixelFormat** –
- **pData** –
- **dataPayloadType** –
- **dataSize** (*size\_t*) –
- **object** (*Creates a new Image*) –
- **constructor** (*either using a default*) –
- **ImagePtr** –
- **width** –
- **height** –

:param : :param offset\_x: :param offset\_y: :param pixel format: :param and a NumPy array containing 8-bit unsigned ints representing the image data: :param (replaces the void\* pData argument).:

**DeepCopy**(*self, pSrcImage*)

#### Parameters

- **pSrcImage** (*The Image to copy the data from.*) –
- **void** –
- **pSrcImage** (*Spinnaker::Image::DeepCopy(const ImagePtr)*) –
- **operation** (*Performs a deep copy of the Image. After this*) –
- **image** (*the*) –
- **not** (*contents and member variables will be the same. The Images will*) –
- **released.** (*share a buffer. The Image's current buffer will not be*) –
- **Parameters** –
- ----- –
- **pSrcImage** –

**GetBitsPerPixel**(*self*) → *size\_t*

*size\_t* Spinnaker::Image::GetBitsPerPixel() const

Gets the number of bits used per pixel in the image. This information is retrieved from the Transport Layer Image format headers. It is retrieved on a per image basis.

The number of bits used per pixel.

**GetBufferSize**(*self*) → *size\_t*

*size\_t* Spinnaker::Image::GetBufferSize() const

Gets the size of the buffer associated with the image in bytes.

The size of the buffer, in bytes.

**GetChunkData**(*self*) → *ChunkData*

const *ChunkData*& Spinnaker::Image::GetChunkData() const

Returns a pointer to a chunk data interface. No ownership is transferred, the chunk data interface reference is valid until Image::Release() is called on this image.

ChunkData interface that provides access to image chunks.

**GetChunkLayoutId**(*self*) → uint64\_t

uint64\_t Spinnaker::Image::GetChunkLayoutId() const

Returns the id of the chunk data layout.

uint64\_t value representing the id of the chunk data layout.

**GetColorProcessing**(*self*) → Spinnaker::ColorProcessingAlgorithm

ColorProcessingAlgorithm Spinnaker::Image::GetColorProcessing() const

Gets the algorithm used to produce the image.

See: Convert()

The color processing algorithm used to produce the image.

**GetDataAbsoluteMax**(*self*) → float

**GetDataAbsoluteMin**(*self*) → float

**GetFrameID**(*self*) → uint64\_t

uint64\_t Spinnaker::Image::GetFrameID() const

Gets the frame ID for this image.

The frame ID.

**GetHeight**(*self*) → size\_t

size\_t Spinnaker::Image::GetHeight() const

Gets the height of the image in pixels. This information is retrieved from the Transport Layer Image format headers. It is retrieved on a per image basis.

The height in pixels.

**GetID**(*self*) → uint64\_t

uint64\_t Spinnaker::Image::GetID() const

Gets a unique ID for this image. Each image in a stream will have a unique ID to help identify it.

The 64 bit unique id for this image.

**GetImageSize**(*self*) → size\_t

size\_t Spinnaker::Image::GetImageSize() const

Returns the size of the image

The image size in bytes.

**GetImageStatus**(*self*) → Spinnaker::ImageStatus

ImageStatus Spinnaker::Image::GetImageStatus() const

Returns data integrity status of the image returned from GetNextImage()

Returns whether image has any data integrity issues.

**static GetImageStatusDescription**(*status*) → char const \*

**Parameters**

**status** (enum Spinnaker::ImageStatus) –

**GetNumChannels**(*self*) → size\_t

**GetPayloadType**(*self*) → size\_t

size\_t Spinnaker::Image::GetPayloadType() const

Gets the payload type that was transmitted. This is a device types specific value that identifies how the image was transmitted. This information is retrieved from the Transport Layer Image format headers. It is retrieved on a per image basis.

Device types specific payload type.

**GetPixelFormat**(*self*) → Spinnaker::PixelFormatEnums

Spinnaker::PixelFormatEnums Spinnaker::Image::GetPixelFormat() const

Returns an enum value that represents the pixel format of this image. The enum can be used with the easy access GenICam features available through the Camera.h header file. This easy access enum can also be used in the Convert() function.

See: Convert()

enum value representing the PixelFormat.

**GetPixelFormatIntType**(*self*) → Spinnaker::PixelFormatIntType

**GetPixelFormatName**(*self*) → gcstring

GenICam::gcstring Spinnaker::Image::GetPixelFormatName() const

Returns a string value that represents this image's pixel format. The string is a valid SFNC name that maps to the underlying TL specific pixel format. This is the most generic way to identify the pixel format of the image.

string value representing the PixelFormat.

**GetPrivateData**(*self*) → void \*

void\* Spinnaker::Image::GetPrivateData() const

Gets a pointer to the user passed data associated with the image. This function is considered unsafe. The pointer returned could be invalidated if the buffer is released. The pointer may also be invalidated if the Image object is passed to Image::Release().

TODO: no way to set private data for image yet.

A pointer to the user passed data pointer.

**GetStreamIndex**(*self*) → uint64\_t

**GetStride**(*self*) → size\_t

size\_t Spinnaker::Image::GetStride() const

Gets the stride of the image in bytes. The stride of an image is how many bytes are in each row. This information is retrieved from the Transport Layer Image format headers. It is retrieved on a per image basis.

The stride in bytes.

**GetTLPayloadType**(*self*) → Spinnaker::TLPayloadType

PayloadTypeInfoIDs Spinnaker::Image::GetTLPayloadType() const

Gets the GenTL specific payload type that was transmitted. This is a Transport Layer specific value that identifies how the image was transmitted. This information is retrieved from the Transport Layer Image format headers. It is retrieved on a per image basis.

Transport Layer specific payload type.

**GetTLPixelFormat**(*self*) → uint64\_t

uint64\_t Spinnaker::Image::GetTLPixelFormat() const

Gets the pixel format of the image. This is a Transport Layer specific pixel format that identifies how the pixels in the image should be interpreted. To understand how to interpret this value it is necessary to know what the transport layer namespace is. This can be retrieved through a call to GetTLPixelFormatNamespace(). This information is retrieved from the Transport Layer Image format headers. It is retrieved on a per image basis.

See: GetTLPixelFormatNamespace()

Transport Layer specific pixel format.

**GetTLPixelFormatNamespace**(*self*) → Spinnaker::TLPixelFormatNamespace

PixelFormatNamespaceID Spinnaker::Image::GetTLPixelFormatNamespace() const

Returns an enum value that represents the namespace in which this image's TL specific pixel format resides. This information is important to properly interpret the value returned by GetTLPixelFormat()

See: GetTLPixelFormat()

enum value representing the PixelFormatNamespace.

**GetTimeStamp**(*self*) → uint64\_t

uint64\_t Spinnaker::Image::GetTimeStamp() const

Gets the time stamp for the image in nanoseconds.

The time stamp of the image.

**GetValidPayloadSize**(*self*) → size\_t

size\_t Spinnaker::Image::GetValidPayloadSize() const

Returns the size of valid data in the image payload. This is the actual amount of data read from the device. A user created image has a payload size of zero. GetBufferSize() returns the total size of bytes allocated for the image.

See: GetBufferSize()

size\_t value representing valid payload.

**GetWidth**(*self*) → size\_t

size\_t Spinnaker::Image::GetWidth() const

Gets the width of the image in pixels. This information is retrieved from the Transport Layer image format headers. It is retrieved on a per image basis.

The width in pixels.

**GetXOffset**(*self*) → size\_t

size\_t Spinnaker::Image::GetXOffset() const

Gets the ROI x offset in pixels for this image. This information is retrieved from the Transport Layer Image format headers. It is retrieved on a per image basis.

The x offset in pixels.

**GetXPadding**(*self*) → size\_t

size\_t Spinnaker::Image::GetXPadding() const

Gets the x padding in bytes for this image. This is the number of bytes at the end of each line to facilitate alignment in buffers. This information is retrieved from the Transport Layer Image format headers. It is retrieved on a per image basis.

The x padding in bytes.

**GetYOffset**(*self*) → size\_t

size\_t Spinnaker::Image::GetYOffset() const

Gets the ROI y offset in pixels for this image. This information is retrieved from the Transport Layer Image format headers. It is retrieved on a per image basis.

The y offset in pixels.

**GetYPadding**(*self*) → size\_t

size\_t Spinnaker::Image::GetYPadding() const

Gets the y padding in bytes for this image. This is the number of bytes at the end of each image to facilitate alignment in buffers. This information is retrieved from the Transport Layer Image format headers. It is retrieved on a per image basis.

The y padding in bytes.

**HasCRC**(*self*) → bool

bool Spinnaker::Image::HasCRC() const

Checks if the image contains ImageCRC checksum from chunk data

Returns true if image contains ImageCRC checksum from chunk data and false otherwise.

**HasChunkData**(*self*) → bool

**IsCompressed**(*self*) → bool

**IsInUse**(*self*) → bool

bool Spinnaker::Image::IsInUse()

Returns true if the image is still in use by the stream

Returns true if the image is in use and false otherwise.

**IsIncomplete**(*self*) → bool

bool Spinnaker::Image::IsIncomplete() const

Returns a boolean value indicating if this image was incomplete. An image is marked as incomplete if the transport layer received less data than it requested.

Returns true if image is incomplete, false otherwise.

**static Load**(*pFilename*, *format*=SPINNAKER\_IMAGE\_FILE\_FORMAT\_FROM\_FILE\_EXT) → *ImagePtr*

#### Parameters

- **pFilename** (*char const \**) –
- **format** (*enum Spinnaker::ImageFileFormat*) –

**Release**(*self*)

void Spinnaker::Image::Release()

**ResetImage**(*self*, *width*, *height*, *offsetX*, *offsetY*, *pixelFormat*)

#### Parameters

- **width** (*The width of image in pixels to set.*) –
- **height** (*The height of image in pixels to set.*) –
- **offsetX** (*The x offset in pixels to set.*) –

- `offsetY` (*The y offset in pixels to set.*) –
- `pixelFormat` (*Pixel format to set.*) –
- `ResetImage`(self –
- `width` –
- `height` –
- `offsetX` –
- `offsetY` –
- `pixelFormat` –
- `pData`) –
- `width` –
- `height` –
- `offsetX` –
- `offsetY` –
- `pixelFormat` –
- `pData` (*Pointer to the image buffer.*) –
- `ResetImage`(self –
- `width` –
- `height` –
- `offsetX` –
- `offsetY` –
- `pixelFormat` –
- `pData` –
- `dataPayloadType` (*enum Spinnaker::TLPayloadType*) –
- `dataSize`) –
- `width` –
- `height` –
- `offsetX` –
- `offsetY` –
- `pixelFormat` –
- `pData` –
- `dataPayloadType` –
- `dataSize` (*size\_t*) –
- `void` –
- `width` –
- `height` –
- `size_t` –

- `offsetX` –
- `offsetY` –
- `pixelFormat` –
- `void` –
- `*pData`) –
- `object.` (*Sets new dimensions of the image*) –
- `Parameters` –
- -----
- `width` –
- `height` –
- `offsetX` –
- `offsetY` –
- `pixelFormat` –
- `pData` –

**Save**(*self*, *pFilename*, *format*=`SPINNAKER_IMAGE_FILE_FORMAT_FROM_FILE_EXT`)

#### Parameters

- `pFilename` (*Filename to save image with.*) –
- `format` (*enum Spinnaker::ImageFileFormat*) –
- `Save`(*self* –
- `pFilename` –
- `pOption`) –
- `pFilename` –
- `pOption` (*Options to use while saving image.*) –
- `Save`(*self* –
- `pFilename` –
- `pOption`) –
- `pFilename` –
- `pOption` –
- `Save`(*self* –
- `pFilename` –
- `pOption`) –
- `pFilename` –
- `pOption` –
- `Save`(*self* –
- `pFilename` –
- `pOption`) –



- `pFilename` –
- `pOption` –
- `Save(self` –
- `pFilename` –
- `pOption)` –
- `pFilename` –
- `pOption` –
- `Save(self` –
- `pFilename` –
- `pOption)` –
- `pFilename` –
- `pOption` –
- `Save(self` –
- `pFilename` –
- `pOption)` –
- `pFilename` –
- `pOption` –
- `Save(self` –
- `pFilename` –
- `option)` –
- `pFilename` –
- `option (Spinnaker::SIOption &)` –
- `void` –
- `*pFilename (Spinnaker::Image::Save(const char)` –
- `&pOption) (BMPOption)` –
- `specified.` (*Saves the image to the specified file name with the options*) –
- `Parameters` –
- -----
- `pFilename` –
- `pOption` –

property `thisown`

The membership flag

## 4.9 PySpin.ImageList

**class** PySpin.ImageList(\*args)

Proxy of C++ Spinnaker::ImageList class.

**Add**(self, image)

Parameters

**image** (Spinnaker::ImagePtr) –

**Append**(self, list)

Parameters

**list** (Spinnaker::ImageList const &) –

**Clear**(self)

**GetByIndex**(self, index) → ImagePtr

Parameters

**index** (unsigned int) –

**GetByPixelFormat**(self, pixelFormat) → ImagePtr

Parameters

**pixelFormat** (enum Spinnaker::PixelFormatEnums) –

**GetSize**(self) → unsigned int

**static Load**(filename) → ImageList

Parameters

**filename** (char const \*) –

**Release**(self)

**RemoveByIndex**(self, index)

Parameters

**index** (unsigned int) –

**RemoveByPixelFormat**(self, pixelFormat)

Parameters

**pixelFormat** (enum Spinnaker::PixelFormatEnums) –

**Save**(self, filename)

Parameters

**filename** (char const \*) –

**property thisown**

The membership flag

## 4.10 PySpin.ImageProcessor

**class** PySpin.ImageProcessor(\*args)

Proxy of C++ Spinnaker::ImageProcessor class.

**ApplyGamma**(self, srcImage, gamma, applyGammaInverse=False) → ImagePtr

### Parameters

- **srcImage** (Spinnaker::ImagePtr const &) –
- **gamma** (float) –
- **applyGammaInverse** (bool) –
- **ApplyGamma**(self –
- **srcImage** –
- **destImage** (Spinnaker::ImagePtr &) –
- **gamma** –
- **applyGammaInverse=False**) –
- **srcImage** –
- **destImage** –
- **gamma** –
- **applyGammaInverse** –

**Convert**(self, srcImage, destFormat) → ImagePtr

### Parameters

- **srcImage** (Spinnaker::ImagePtr const &) –
- **destFormat** (enum Spinnaker::PixelFormatEnums) –
- **Convert**(self –
- **srcImage** –
- **destImage** (Spinnaker::ImagePtr &) –
- **destFormat**) –
- **srcImage** –
- **destImage** –
- **destFormat** –
- **Convert**(self –
- **srcImageList** (Spinnaker::ImageList const &) –
- **ImagePtr** (destFormat) ->) –
- **srcImageList** –
- **destFormat** –
- **Convert**(self –
- **srcImageList** –

- **destImage** –
- **destFormat** –
- **srcImageList** –
- **destImage** –
- **destFormat** –

**GetColorProcessing**(*self*) → Spinnaker::ColorProcessingAlgorithm

**GetNumDecompressionThreads**(*self*) → unsigned int

**SetColorProcessing**(*self*, *colorAlgorithm*)

Parameters

**colorAlgorithm** (enum Spinnaker::ColorProcessingAlgorithm) –

**SetNumDecompressionThreads**(*self*, *numThreads*)

Parameters

**numThreads** (unsigned int) –

**property thisown**

The membership flag

## 4.11 PySpin.ImagePtr

**class** PySpin.**ImagePtr**(\*args)

A reference tracked pointer to an image object. When the ImagePtr goes out of scope, it will trigger an auto release of the image from the stream.

C++ includes: ImagePtr.h

**property thisown**

The membership flag

## 4.12 PySpin.IInterface

**class** PySpin.**IInterface**(\*args, \*\*kwargs)

Proxy of C++ Spinnaker::IInterface class.

**GetCameras**(*self*, *updateCameras=True*) → *CameraList*

Parameters

**updateCameras** (bool) –

**GetTLNodeMap**(*self*) → INodeMap

**IsInUse**(*self*) → bool

**IsValid**(*self*) → bool

**RegisterEventHandler**(*self*, *evtHandlerToRegister*)

**Parameters**

**evtHandlerToRegister** (*Spinnaker::EventHandler &*) –

**SendActionCommand**(*self*, *deviceKey*, *groupKey*, *groupMask*, *actionTime=0*, *pResultSize=None*, *results=0*)

**Parameters**

- **deviceKey** (*unsigned int*) –
- **groupKey** (*unsigned int*) –
- **groupMask** (*unsigned int*) –
- **actionTime** (*unsigned long long*) –
- **pResultSize** (*unsigned int \**) –
- **results** (*Spinnaker::ActionCommandResult []*) –

**property TLInterface**

**UnregisterEventHandler**(*self*, *evtHandlerToUnregister*)

**Parameters**

**evtHandlerToUnregister** (*Spinnaker::EventHandler &*) –

**UpdateCameras**(*self*) → bool

**property thisown**

The membership flag

## 4.13 PySpin.InterfaceList

**class** **PySpin.InterfaceList**(\*args)

A list of the available interfaces on the system.

C++ includes: InterfaceList.h

**Add**(*self*, *iface*)

**Parameters**

**iface** (*Spinnaker::InterfacePtr*) –

**Append**(*self*, *list*)

**Parameters**

**list** (*Spinnaker::InterfaceList const \**) –

**Clear**(*self*)

void *Spinnaker::InterfaceList::Clear*()

Clears the list of interfaces and destroys their corresponding objects. It is important to first make sure there are no referenced cameras still in use before calling *Clear*(). If a camera on any of the interfaces is still in use this function will throw an exception.

**GetByIndex**(*self*, *index*) → *InterfacePtr*

**Parameters**

- **index** (*The index at which to retrieve the Interface object*) –
- **const** (*InterfacePtr Spinnaker::InterfaceList::GetByIndex(int index)*) –
- **"index"**. (*Returns a pointer to an Interface object at the*) –
- **Parameters** –
- ----- –
- **index** –
- **object.** (*A pointer to an Interface*) –

**GetByInterfaceID**(*self*, *interfaceID*) → *InterfacePtr*

**Parameters**

**interfaceID** (*std::string*) –

**GetSize**(*self*) → unsigned int

int Spinnaker::InterfaceList::GetSize() const

Returns the size of the interface list. The size is the number of Interface objects stored in the list.

An integer that represents the list size.

**Remove**(*self*, *iface*)

**Parameters**

**iface** (*Spinnaker::InterfacePtr*) –

**property thisown**

The membership flag

## 4.14 PySpin.InterfacePtr

**class** PySpin.**InterfacePtr**(\*args)

A reference tracked pointer to the interface object.

C++ includes: InterfacePtr.h

**property thisown**

The membership flag

## 4.15 PySpin.SpinnakerException

**class** PySpin.**SpinnakerException**

Exception class for the PySpin module. This class has these attributes: message, errorcode, fullmessage

**errorcode** = 0

**fullmessage** = ''

**message** = ''

## 4.16 PySpin.SpinVideo

### class PySpin.SpinVideo

Provides the functionality for the user to record images to an AVI file.

C++ includes: SpinVideo.h

**Append**(*self*, *pImage*)

#### Parameters

- **pImage** (*The image to append.*) –
- **virtual** –
- **pImage** (*void Spinnaker::Video::SpinVideo::Append(ImagePtr) –*
- **file.** (*Append an image to the AVI/MP4) –*
- **Parameters** –
- **-----** –
- **pImage** –

**Close**(*self*)

virtual void Spinnaker::Video::SpinVideo::Close()

Close the AVI/MP4 file.

See: Open()

**Open**(*self*, *pFileName*, *pOption*)

#### Parameters

- **pFileName** (*The filename of the MP4 file.*) –
- **pOption** (*H264 options to apply to the MP4 file.*) –
- **Open**(*self* –
- **pFileName** –
- **pOption**) –
- **pFileName** –
- **pOption** –
- **Open**(*self* –
- **pFileName** –
- **pOption**) –
- **pFileName** –
- **pOption** –
- **void** (*virtual*) –
- **\*pFileName** (*Spinnaker::Video::SpinVideo::Open(const char) –*
- **Video::H264Option** –
- **&pOption**) –

- **The** (*Open an H264 MP4 file in preparation for writing Images to disk.*) –
- **automatically** (*(size of MP4 files is limited to 2GB. The filenames are)*) –
- **specified.** (*generated using the filename*) –
- **Parameters** –
- ----- –
- **pFileName** –
- **pOption** –
- **See** (*H264Option*) –
- **See** –

**SetMaximumFileSize**(*self, size*)

**Parameters**

**size** (*unsigned int*) –

**property thisown**

The membership flag

## 4.17 PySpin.System

**class** **PySpin.System**(\*args, \*\*kwargs)

The system object is used to retrieve the list of interfaces and cameras available.

C++ includes: System.h

**GetCameras**(*self, updateInterfaces=True, updateCameras=True*) → *CameraList*

**Parameters**

- **updateInterfaces** (*Determines whether or not updateInterfaceList() is*) –
- **updateCameras** (*Determines whether or not UpdateCameras() is called*) –
- **CameraList** –
- **updateInterfaces=true** (*Spinnaker::System::GetCameras(bool)*) –
- **bool** –
- **updateCameras=true**) –
- **call** (*Returns a list of cameras that are available on the system. This*) –
- **interfaces.** (*(returns both GigE Vision and Usb3 Vision cameras from all)*) –
- **It** (*The camera list object will reference count the cameras it returns.*) –
- **before** (*(is important that the camera list is destroyed or is cleared)*) –



- **system**->(calling `system-> ReleaseInstance()` or else the call to)-
- **a**(`ReleaseInstance()` will result in an error message thrown that)-
- **held.**(reference to the camera is still)-
- **See** (`CameraList::Clear()`)-
- **See** -
- **Parameters** -
- ----- -
- **updateInterfaces** -
- **system**(before getting cameras from available interfaces on the)-
- **updateCameras** -
- **system** -
- **cameras.**(An `CameraList` object that contains a list of all)-

**static** `GetInstance()` → *SystemPtr*

**GetInterfaces**(*self*, *updateInterface=True*) → *InterfaceList*

#### Parameters

- **updateInterface**(Determines whether or not `UpdateInterfaceList()` is)-
- **Spinnaker::System::GetInterfaces**(**bool** (*InterfaceList*))-
- **updateInterface=true**) -
- **call**(Returns a list of interfaces available on the system. This)-
- **interfaces.**(An `InterfaceList` object that contains a list of all)-
- **Parameters** -
- ----- -
- **updateInterface** -
- **interfaces**(called before getting available)-
- **interfaces.** -

**GetLibraryVersion**(*self*) → `LibraryVersion`

**GetLoggingEventPriorityLevel**(*self*) → `Spinnaker::SpinnakerLogLevel`

`SpinnakerLogLevel Spinnaker::System::GetLoggingEventPriorityLevel()`

Retrieves the current logging event priority level.

Spinnaker uses five levels of logging: Error - failures that are non- recoverable without user intervention.

Warning - failures that are recoverable without user intervention.

Notice - information about events such as camera arrival and removal, initialization and deinitialization, starting and stopping image acquisition, and feature modification.

Info - information about recurring events that are generated regularly such as information on individual images.

Debug - information that can be used to troubleshoot the system.

See: SpinnakerLogLevel

Level The threshold level

**GetTLNodeMap**(*self*) → INodeMap

**IsInUse**(*self*) → bool

bool Spinnaker::System::IsInUse()

Checks if the system is in use by any interface or camera objects.

Returns true if the system is in use and false otherwise.

**RegisterEventHandler**(*self*, *evtHandlerToRegister*, *updateInterface=False*)

**Parameters**

- **evtHandlerToRegister** (*Spinnaker::EventHandler &*) –
- **updateInterface** (*bool*) –

**RegisterLoggingEventHandler**(*self*, *handler*)

**Parameters**

- **handler** (*Spinnaker::LoggingEventHandler &*) –

**ReleaseInstance**(*self*)

void Spinnaker::System::ReleaseInstance()

This call releases the instance of the System Singleton for this process. After successfully releasing the System instance the pointer returned by GetInstance() will be invalid. Calling ReleaseInstance while a camera reference is still held will throw an error of type SPINNAKER\_ERR\_RESOURCE\_IN\_USE.

See: Error

See: GetInstance()

**SendActionCommand**(*self*, *deviceKey*, *groupKey*, *groupMask*, *actionTime=0*, *pResultSize=None*, *results=0*)

**Parameters**

- **deviceKey** (*Spinnaker::System::SendActionCommand(unsigned int)*) –
- **groupKey** (*int*) –
- **groupMask** (*unsigned int*) –
- **actionTime** (*unsigned long long*) –
- **pResultSize** (*unsigned int \**) –
- **results** (*Spinnaker::ActionCommandResult []*) –
- **void** –
- **deviceKey** –
- **unsigned** –
- **groupKey** –
- **groupMask** –
- **actionTime=0** (*unsigned long long*) –

:param : :param unsigned int \*pResultSize=0: :param ActionCommandResult results[]=NULL): :param Broadcast an Action Command to all devices on system: :param Parameters: :param ———: :param deviceKey: :type deviceKey: The Action Command's device key :param groupKey: :type groupKey: The Action Command's group key :param groupMask: :type groupMask: The Action Command's group mask :param actionTime: :type actionTime: (Optional) Time when to assert a future action. Zero :param means immediate action.: :param pResultSize: :type pResultSize: (Optional) The number of results in the results array. :param The value passed should be equal to the expected number of devices: :param that acknowledge the command. Returns the number of received results.: :param results: :type results: (Optional) An Array with \*pResultSize elements to hold the :param action command result status. The buffer is filled starting from index: :param 0. If received results are less than expected number of devices that: :param acknowledge the command: :param remaining results are not changed. If: :param received results are more than expected number of devices that: :param acknowledge the command: :param extra results are ignored and not appended to: :param array. This parameter is ignored if pResultSize is 0. Thus this: :param parameter can be NULL if pResultSize is 0 or NULL.:

**SetLoggingEventPriorityLevel**(*self, level*)

**Parameters**

- **level** (*enum Spinnaker::SpinnakerLogLevel*) –
- **void** –
- **Spinnaker::System::SetLoggingEventPriorityLevel**(*SpinnakerLogLevel level*) –
- **events** (*Sets a threshold priority level for logging event. Logging*) –
- **callbacks.** (*below such level will not trigger*) –
- **logging** (*Spinnaker uses five levels of*) –
- **intervention.** (*Warning - failures that are recoverable without user*) –
- **intervention.** –
- **removal** (*Notice - information about events such as camera arrival and*) –

:param : :param initialization and deinitialization: :param starting and stopping image: :param acquisition: :param and feature modification.: :param Info - information about recurring events that are generated regularly: :param such as information on individual images.: :param Debug - information that can be used to troubleshoot the system.: :param See: :type See: *SpinnakerLogLevel* :param Parameters: :param ———: :param level: :type level: The threshold level

**UnregisterAllLoggingEventHandlers**(*self*)

**UnregisterEventHandler**(*self, evtHandlerToUnregister*)

**Parameters**

- **evtHandlerToUnregister** (*Spinnaker::EventHandler &*) –

**UnregisterLoggingEventHandler**(*self, handler*)

**Parameters**

- **handler** (*Spinnaker::LoggingEventHandler &*) –

**UpdateCameras**(*self, updateInterfaces=True*) → bool

**Parameters**

- **updateInterfaces** (*bool*) –

- **bool** –
- **updateInterfaces=true**) (*Spinnaker::System::UpdateCameras(bool)* –
- **that** (*Updates the list of cameras on the system. Note*) –
- **each** (*System::GetCameras()* internally calls *UpdateCameras()* for) –
- **the** (*interface it enumerates. If the list changed between this call and*) –
- **true** (*last time UpdateCameras was called then the return value will be*) –

:param : :param otherwise it is false.: :param See: :type See: GetCameras() :param Parameters: :param  
———: :param updateInterfaces: :type updateInterfaces: Determines whether or not UpdateInter-  
faceList() is :param called before updating cameras for available interfaces on the system: :param True  
if cameras changed on interface and false otherwise.:

**UpdateInterfaceList**(*self*)

**property thisown**

The membership flag

## 4.18 PySpin.SystemPtr

**class** PySpin.**SystemPtr**(\*args)

A reference tracked pointer to a system object.

C++ includes: SystemPtr.h

**property thisown**

The membership flag

## QUICKSPIN CLASSES

- *PySpin.TransportLayerDevice*
- *PySpin.TransportLayerInterface*
- *PySpin.TransportLayerStream*

### 5.1 PySpin.TransportLayerDevice

**class** `PySpin.TransportLayerDevice`(*nodeMapTLDevice*)

Part of the QuickSpin API to provide access to camera information without having to first initialize the camera.

C++ includes: `TransportLayerDevice.h`

**property** `DeviceAccessStatus`

**property** `DeviceBootloaderVersion`

**property** `DeviceCurrentSpeed`

**property** `DeviceDisplayName`

**property** `DeviceDriverVersion`

**property** `DeviceEndiannessMechanism`

**property** `DeviceID`

**property** `DeviceInstanceId`

**property** `DeviceIsUpdater`

**property** `DeviceLinkSpeed`

**property** `DeviceLocation`

**property** `DeviceModelName`

**property** `DeviceMulticastMonitorMode`

**property** `DevicePortId`

**property** `DeviceReset`

property DeviceSerialNumber  
property DeviceType  
property DeviceU3VProtocol  
property DeviceUserID  
property DeviceVendorName  
property DeviceVersion  
property GUIXMLLocation  
property GUIXMLPath  
property GenICamXMLLocation  
property GenICamXMLPath  
property GevCCP  
property GevDeviceAutoForceIP  
property GevDeviceDiscoverMaximumPacketSize  
property GevDeviceForceGateway  
property GevDeviceForceIP  
property GevDeviceForceIPAddress  
property GevDeviceForceSubnetMask  
property GevDeviceGateway  
property GevDeviceIPAddress  
property GevDeviceIsWrongSubnet  
property GevDeviceMACAddress  
property GevDeviceMaximumPacketSize  
property GevDeviceMaximumRetryCount  
property GevDeviceModeIsBigEndian  
property GevDevicePort  
property GevDeviceReadAndWriteTimeout  
property GevDeviceSubnetMask  
property GevVersionMajor  
property GevVersionMinor  
property thisown

The membership flag

## 5.2 PySpin.TransportLayerInterface

**class** PySpin.TransportLayerInterface(*nodeMapTLDevice*)

Part of the QuickSpin API to provide access to camera information without having to first initialize the camera.

C++ includes: TransportLayerInterface.h

**property** ActionCommand

**property** DeviceAccessStatus

**property** DeviceCount

**property** DeviceID

**property** DeviceModelName

**property** DeviceSelector

**property** DeviceSerialNumber

**property** DeviceUnlock

**property** DeviceUpdateList

**property** DeviceVendorName

**property** FilterDriverStatus

**property** GevActionDeviceKey

**property** GevActionGroupKey

**property** GevActionGroupMask

**property** GevActionTime

**property** GevDeviceAutoForceIP

**property** GevDeviceForceGateway

**property** GevDeviceForceIP

**property** GevDeviceForceIPAddress

**property** GevDeviceForceSubnetMask

**property** GevDeviceGateway

**property** GevDeviceIPAddress

**property** GevDeviceMACAddress

**property** GevDeviceSubnetMask

**property** GevInterfaceGateway

**property** GevInterfaceGatewaySelector

**property** GevInterfaceMACAddress

```
property GevInterfaceMTU
property GevInterfaceReceiveLinkSpeed
property GevInterfaceSubnetIPAddress
property GevInterfaceSubnetMask
property GevInterfaceSubnetSelector
property GevInterfaceTransmitLinkSpeed
property HostAdapterDriverVersion
property HostAdapterName
property HostAdapterVendor
property IncompatibleDeviceCount
property IncompatibleDeviceID
property IncompatibleDeviceModelName
property IncompatibleDeviceSelector
property IncompatibleDeviceVendorName
property IncompatibleGevDeviceIPAddress
property IncompatibleGevDeviceMACAddress
property IncompatibleGevDeviceSubnetMask
property InterfaceDisplayName
property InterfaceID
property InterfaceType
property POEStatus
property thisown
```

The membership flag

## 5.3 PySpin.TransportLayerStream

**class** `PySpin.TransportLayerStream`(*nodeMapTLDevice*)

Part of the QuickSpin API to provide access to camera information without having to first initialize the camera.

C++ includes: `TransportLayerStream.h`

```
property StreamAnnounceBufferMinimum
property StreamAnnouncedBufferCount
property StreamBlockTransferSize
```



property StreamBufferAlignment  
property StreamBufferCountManual  
property StreamBufferCountMax  
property StreamBufferCountMode  
property StreamBufferCountResult  
property StreamBufferHandlingMode  
property StreamCRCCheckEnable  
property StreamChunkCountMaximum  
property StreamDeliveredFrameCount  
property StreamDroppedFrameCount  
property StreamID  
property StreamIncompleteFrameCount  
property StreamInputBufferCount  
property StreamIsGrabbing  
property StreamLostFrameCount  
property StreamMissedPacketCount  
property StreamMode  
property StreamOutputBufferCount  
property StreamPacketResendEnable  
property StreamPacketResendMaxRequests  
property StreamPacketResendReceivedPacketCount  
property StreamPacketResendRequestCount  
property StreamPacketResendRequestSuccessCount  
property StreamPacketResendRequestedPacketCount  
property StreamPacketResendTimeout  
property StreamReceivedFrameCount  
property StreamReceivedPacketCount  
property StreamStartedFrameCount  
property StreamType  
property thisown

The membership flag



**PYSPIN MODULE**



## PYTHON MODULE INDEX

### p

PySpin, [71](#)



## A

AasRoiEnable (*PySpin.Camera* property), 12  
 AasRoiHeight (*PySpin.Camera* property), 12  
 AasRoiOffsetX (*PySpin.Camera* property), 12  
 AasRoiOffsetY (*PySpin.Camera* property), 12  
 AasRoiWidth (*PySpin.Camera* property), 12  
 AcquisitionAbort (*PySpin.Camera* property), 12  
 AcquisitionArm (*PySpin.Camera* property), 12  
 AcquisitionBurstFrameCount (*PySpin.Camera* property), 12  
 AcquisitionFrameCount (*PySpin.Camera* property), 12  
 AcquisitionFrameRate (*PySpin.Camera* property), 12  
 AcquisitionFrameRateEnable (*PySpin.Camera* property), 12  
 AcquisitionLineRate (*PySpin.Camera* property), 12  
 AcquisitionMode (*PySpin.Camera* property), 12  
 AcquisitionResultingFrameRate (*PySpin.Camera* property), 12  
 AcquisitionStart (*PySpin.Camera* property), 12  
 AcquisitionStatus (*PySpin.Camera* property), 12  
 AcquisitionStatusSelector (*PySpin.Camera* property), 12  
 AcquisitionStop (*PySpin.Camera* property), 12  
 ActionCommand (*PySpin.TransportLayerInterface* property), 67  
 ActionDeviceKey (*PySpin.Camera* property), 12  
 ActionGroupKey (*PySpin.Camera* property), 12  
 ActionGroupMask (*PySpin.Camera* property), 12  
 ActionQueueSize (*PySpin.Camera* property), 12  
 ActionSelector (*PySpin.Camera* property), 12  
 ActionUnconditionalMode (*PySpin.Camera* property), 12  
 AdaptiveCompressionEnable (*PySpin.Camera* property), 12  
 AdcBitDepth (*PySpin.Camera* property), 13  
 Add() (*PySpin.CameraList* method), 38  
 Add() (*PySpin.ImageList* method), 54  
 Add() (*PySpin.InterfaceList* method), 57  
 aPAUSEMACtrlFramesReceived (*PySpin.Camera* property), 33  
 aPAUSEMACtrlFramesTransmitted (*PySpin.Camera*

property), 33  
 Append() (*PySpin.CameraList* method), 38  
 Append() (*PySpin.ImageList* method), 54  
 Append() (*PySpin.InterfaceList* method), 57  
 Append() (*PySpin.SpinVideo* method), 59  
 ApplyGamma() (*PySpin.ImageProcessor* method), 55  
 AutoAlgorithmSelector (*PySpin.Camera* property), 13  
 AutoExposureControlLoopDamping (*PySpin.Camera* property), 13  
 AutoExposureControlPriority (*PySpin.Camera* property), 13  
 AutoExposureEVCompensation (*PySpin.Camera* property), 13  
 AutoExposureExposureTimeLowerLimit (*PySpin.Camera* property), 13  
 AutoExposureExposureTimeUpperLimit (*PySpin.Camera* property), 13  
 AutoExposureGainLowerLimit (*PySpin.Camera* property), 13  
 AutoExposureGainUpperLimit (*PySpin.Camera* property), 13  
 AutoExposureGreyValueLowerLimit (*PySpin.Camera* property), 13  
 AutoExposureGreyValueUpperLimit (*PySpin.Camera* property), 13  
 AutoExposureLightingMode (*PySpin.Camera* property), 13  
 AutoExposureMeteringMode (*PySpin.Camera* property), 13  
 AutoExposureTargetGreyValue (*PySpin.Camera* property), 13  
 AutoExposureTargetGreyValueAuto (*PySpin.Camera* property), 13

## B

BalanceRatio (*PySpin.Camera* property), 13  
 BalanceRatioSelector (*PySpin.Camera* property), 13  
 BalanceWhiteAuto (*PySpin.Camera* property), 13  
 BalanceWhiteAutoDamping (*PySpin.Camera* property), 13

BalanceWhiteAutoLowerLimit (*PySpin.Camera property*), 13  
BalanceWhiteAutoProfile (*PySpin.Camera property*), 13  
BalanceWhiteAutoUpperLimit (*PySpin.Camera property*), 13  
BeginAcquisition() (*PySpin.CameraBase method*), 34  
BinningHorizontal (*PySpin.Camera property*), 13  
BinningHorizontalMode (*PySpin.Camera property*), 13  
BinningSelector (*PySpin.Camera property*), 13  
BinningVertical (*PySpin.Camera property*), 13  
BinningVerticalMode (*PySpin.Camera property*), 13  
BlackLevel (*PySpin.Camera property*), 13  
BlackLevelAuto (*PySpin.Camera property*), 13  
BlackLevelAutoBalance (*PySpin.Camera property*), 13  
BlackLevelClampingEnable (*PySpin.Camera property*), 13  
BlackLevelRaw (*PySpin.Camera property*), 14  
BlackLevelSelector (*PySpin.Camera property*), 14

## C

Camera (*class in PySpin*), 12  
CameraBase (*class in PySpin*), 34  
CameraList (*class in PySpin*), 38  
CameraPtr (*class in PySpin*), 40  
CBasePtr (*class in PySpin*), 11  
channel (*PySpin.ChannelStatistics property*), 41  
ChannelStatistics (*class in PySpin*), 41  
CheckCRC() (*PySpin.Image method*), 45  
ChunkBlackLevel (*PySpin.Camera property*), 14  
ChunkBlackLevelSelector (*PySpin.Camera property*), 14  
ChunkCompressionMode (*PySpin.Camera property*), 14  
ChunkCompressionRatio (*PySpin.Camera property*), 14  
ChunkCounterSelector (*PySpin.Camera property*), 14  
ChunkCounterValue (*PySpin.Camera property*), 14  
ChunkCRC (*PySpin.Camera property*), 14  
ChunkData (*class in PySpin*), 41  
ChunkEnable (*PySpin.Camera property*), 14  
ChunkEncoderSelector (*PySpin.Camera property*), 14  
ChunkEncoderStatus (*PySpin.Camera property*), 14  
ChunkEncoderValue (*PySpin.Camera property*), 14  
ChunkExposureEndLineStatusAll (*PySpin.Camera property*), 14  
ChunkExposureTime (*PySpin.Camera property*), 14  
ChunkExposureTimeSelector (*PySpin.Camera property*), 14  
ChunkFrameID (*PySpin.Camera property*), 14  
ChunkGain (*PySpin.Camera property*), 14  
ChunkGainSelector (*PySpin.Camera property*), 14

ChunkHeight (*PySpin.Camera property*), 14  
ChunkImage (*PySpin.Camera property*), 14  
ChunkImageComponent (*PySpin.Camera property*), 14  
ChunkInferenceBoundingBoxResult (*PySpin.Camera property*), 14  
ChunkInferenceConfidence (*PySpin.Camera property*), 14  
ChunkInferenceFrameId (*PySpin.Camera property*), 14  
ChunkInferenceResult (*PySpin.Camera property*), 14  
ChunkLinePitch (*PySpin.Camera property*), 14  
ChunkLineStatusAll (*PySpin.Camera property*), 14  
ChunkModeActive (*PySpin.Camera property*), 14  
ChunkOffsetX (*PySpin.Camera property*), 14  
ChunkOffsetY (*PySpin.Camera property*), 14  
ChunkPartSelector (*PySpin.Camera property*), 15  
ChunkPixelDynamicRangeMax (*PySpin.Camera property*), 15  
ChunkPixelDynamicRangeMin (*PySpin.Camera property*), 15  
ChunkPixelFormat (*PySpin.Camera property*), 15  
ChunkRegionID (*PySpin.Camera property*), 15  
ChunkScan3dAxisMax (*PySpin.Camera property*), 15  
ChunkScan3dAxisMin (*PySpin.Camera property*), 15  
ChunkScan3dCoordinateOffset (*PySpin.Camera property*), 15  
ChunkScan3dCoordinateReferenceSelector (*PySpin.Camera property*), 15  
ChunkScan3dCoordinateReferenceValue (*PySpin.Camera property*), 15  
ChunkScan3dCoordinateScale (*PySpin.Camera property*), 15  
ChunkScan3dCoordinateSelector (*PySpin.Camera property*), 15  
ChunkScan3dCoordinateSystem (*PySpin.Camera property*), 15  
ChunkScan3dCoordinateSystemReference (*PySpin.Camera property*), 15  
ChunkScan3dCoordinateTransformSelector (*PySpin.Camera property*), 15  
ChunkScan3dDistanceUnit (*PySpin.Camera property*), 15  
ChunkScan3dInvalidDataFlag (*PySpin.Camera property*), 15  
ChunkScan3dInvalidDataValue (*PySpin.Camera property*), 15  
ChunkScan3dOutputMode (*PySpin.Camera property*), 15  
ChunkScan3dTransformValue (*PySpin.Camera property*), 15  
ChunkScanLineSelector (*PySpin.Camera property*), 15  
ChunkSelector (*PySpin.Camera property*), 15  
ChunkSequencerSetActive (*PySpin.Camera property*), 15



- erty), 15
- ChunkSerialData (PySpin.Camera property), 15
- ChunkSerialDataLength (PySpin.Camera property), 15
- ChunkSerialReceiveOverflow (PySpin.Camera property), 15
- ChunkSourceID (PySpin.Camera property), 15
- ChunkStreamChannelID (PySpin.Camera property), 15
- ChunkTimerSelector (PySpin.Camera property), 15
- ChunkTimerValue (PySpin.Camera property), 15
- ChunkTimestamp (PySpin.Camera property), 15
- ChunkTimestampLatchValue (PySpin.Camera property), 16
- ChunkTransferBlockID (PySpin.Camera property), 16
- ChunkTransferQueueCurrentBlockCount (PySpin.Camera property), 16
- ChunkTransferStreamID (PySpin.Camera property), 16
- ChunkWidth (PySpin.Camera property), 16
- ClConfiguration (PySpin.Camera property), 16
- Clear() (PySpin.CameraList method), 39
- Clear() (PySpin.ImageList method), 54
- Clear() (PySpin.InterfaceList method), 57
- Close() (PySpin.SpinVideo method), 59
- ClTimeSlotsCount (PySpin.Camera property), 16
- ColorTransformationEnable (PySpin.Camera property), 16
- ColorTransformationSelector (PySpin.Camera property), 16
- ColorTransformationValue (PySpin.Camera property), 16
- ColorTransformationValueSelector (PySpin.Camera property), 16
- CompressionRatio (PySpin.Camera property), 16
- CompressionSaturationPriority (PySpin.Camera property), 16
- Convert() (PySpin.ImageProcessor method), 55
- CounterDelay (PySpin.Camera property), 16
- CounterDuration (PySpin.Camera property), 16
- CounterEventActivation (PySpin.Camera property), 16
- CounterEventSource (PySpin.Camera property), 16
- CounterReset (PySpin.Camera property), 16
- CounterResetActivation (PySpin.Camera property), 16
- CounterResetSource (PySpin.Camera property), 16
- CounterSelector (PySpin.Camera property), 16
- CounterStatus (PySpin.Camera property), 16
- CounterTriggerActivation (PySpin.Camera property), 16
- CounterTriggerSource (PySpin.Camera property), 16
- CounterValue (PySpin.Camera property), 16
- CounterValueAtReset (PySpin.Camera property), 16
- Create() (PySpin.Image static method), 45
- CxpConnectionSelector (PySpin.Camera property), 16
- CxpConnectionTestErrorCount (PySpin.Camera property), 16
- CxpConnectionTestMode (PySpin.Camera property), 16
- CxpConnectionTestPacketCount (PySpin.Camera property), 16
- CxpLinkConfiguration (PySpin.Camera property), 16
- CxpLinkConfigurationPreferred (PySpin.Camera property), 17
- CxpLinkConfigurationStatus (PySpin.Camera property), 17
- CxpPoCxpAuto (PySpin.Camera property), 17
- CxpPoCxpStatus (PySpin.Camera property), 17
- CxpPoCxpTripReset (PySpin.Camera property), 17
- CxpPoCxpTurnOff (PySpin.Camera property), 17
- ## D
- DecimationHorizontal (PySpin.Camera property), 17
- DecimationHorizontalMode (PySpin.Camera property), 17
- DecimationSelector (PySpin.Camera property), 17
- DecimationVertical (PySpin.Camera property), 17
- DecimationVerticalMode (PySpin.Camera property), 17
- DeepCopy() (PySpin.Image method), 46
- DefectCorrectionMode (PySpin.Camera property), 17
- DefectCorrectStaticEnable (PySpin.Camera property), 17
- DefectTableApply (PySpin.Camera property), 17
- DefectTableCoordinateX (PySpin.Camera property), 17
- DefectTableCoordinateY (PySpin.Camera property), 17
- DefectTableFactoryRestore (PySpin.Camera property), 17
- DefectTableIndex (PySpin.Camera property), 17
- DefectTablePixelCount (PySpin.Camera property), 17
- DefectTableSave (PySpin.Camera property), 17
- DeInit() (PySpin.CameraBase method), 34
- Deinterlacing (PySpin.Camera property), 17
- DeviceAccessStatus (PySpin.TransportLayerDevice property), 65
- DeviceAccessStatus (PySpin.TransportLayerInterface property), 67
- DeviceArrivalEventHandler (class in PySpin), 5
- DeviceBootloaderVersion (PySpin.TransportLayerDevice property), 65
- DeviceCharacterSet (PySpin.Camera property), 17
- DeviceClockFrequency (PySpin.Camera property), 17
- DeviceClockSelector (PySpin.Camera property), 17

`DeviceConnectionSelector` (*PySpin.Camera* property), 17

`DeviceConnectionSpeed` (*PySpin.Camera* property), 17

`DeviceConnectionStatus` (*PySpin.Camera* property), 17

`DeviceCount` (*PySpin.TransportLayerInterface* property), 67

`DeviceCurrentSpeed` (*PySpin.TransportLayerDevice* property), 65

`DeviceDisplayName` (*PySpin.TransportLayerDevice* property), 65

`DeviceDriverVersion` (*PySpin.TransportLayerDevice* property), 65

`DeviceEndiannessMechanism` (*PySpin.TransportLayerDevice* property), 65

`DeviceEventChannelCount` (*PySpin.Camera* property), 17

`DeviceEventHandler` (class in *PySpin*), 6

`DeviceFamilyName` (*PySpin.Camera* property), 17

`DeviceFeaturePersistenceEnd` (*PySpin.Camera* property), 17

`DeviceFeaturePersistenceStart` (*PySpin.Camera* property), 17

`DeviceFirmwareVersion` (*PySpin.Camera* property), 18

`DeviceGenCPVersionMajor` (*PySpin.Camera* property), 18

`DeviceGenCPVersionMinor` (*PySpin.Camera* property), 18

`DeviceID` (*PySpin.Camera* property), 18

`DeviceID` (*PySpin.TransportLayerDevice* property), 65

`DeviceID` (*PySpin.TransportLayerInterface* property), 67

`DeviceIndicatorMode` (*PySpin.Camera* property), 18

`DeviceInstanceId` (*PySpin.TransportLayerDevice* property), 65

`DeviceIsUpdater` (*PySpin.TransportLayerDevice* property), 65

`DeviceLinkBandwidthReserve` (*PySpin.Camera* property), 18

`DeviceLinkCommandTimeout` (*PySpin.Camera* property), 18

`DeviceLinkConnectionCount` (*PySpin.Camera* property), 18

`DeviceLinkCurrentThroughput` (*PySpin.Camera* property), 18

`DeviceLinkHeartbeatMode` (*PySpin.Camera* property), 18

`DeviceLinkHeartbeatTimeout` (*PySpin.Camera* property), 18

`DeviceLinkSelector` (*PySpin.Camera* property), 18

`DeviceLinkSpeed` (*PySpin.Camera* property), 18

`DeviceLinkSpeed` (*PySpin.TransportLayerDevice* property), 65

`DeviceLinkThroughputLimit` (*PySpin.Camera* property), 18

`DeviceLinkThroughputLimitMode` (*PySpin.Camera* property), 18

`DeviceLocation` (*PySpin.TransportLayerDevice* property), 65

`DeviceManifestEntrySelector` (*PySpin.Camera* property), 18

`DeviceManifestPrimaryURL` (*PySpin.Camera* property), 18

`DeviceManifestSchemaMajorVersion` (*PySpin.Camera* property), 18

`DeviceManifestSchemaMinorVersion` (*PySpin.Camera* property), 18

`DeviceManifestSecondaryURL` (*PySpin.Camera* property), 18

`DeviceManifestXMLMajorVersion` (*PySpin.Camera* property), 18

`DeviceManifestXMLMinorVersion` (*PySpin.Camera* property), 18

`DeviceManifestXMLSubMinorVersion` (*PySpin.Camera* property), 18

`DeviceManufacturerInfo` (*PySpin.Camera* property), 18

`DeviceMaxThroughput` (*PySpin.Camera* property), 18

`DeviceModelName` (*PySpin.Camera* property), 18

`DeviceModelName` (*PySpin.TransportLayerDevice* property), 65

`DeviceModelName` (*PySpin.TransportLayerInterface* property), 67

`DeviceMulticastMonitorMode` (*PySpin.TransportLayerDevice* property), 65

`DevicePortId` (*PySpin.TransportLayerDevice* property), 65

`DevicePowerSupplySelector` (*PySpin.Camera* property), 18

`DeviceRegistersCheck` (*PySpin.Camera* property), 18

`DeviceRegistersEndianness` (*PySpin.Camera* property), 18

`DeviceRegistersStreamingEnd` (*PySpin.Camera* property), 18

`DeviceRegistersStreamingStart` (*PySpin.Camera* property), 18

`DeviceRegistersValid` (*PySpin.Camera* property), 19

`DeviceRemovalEventHandler` (class in *PySpin*), 6

`DeviceReset` (*PySpin.Camera* property), 19

`DeviceReset` (*PySpin.TransportLayerDevice* property), 65

`DeviceScanType` (*PySpin.Camera* property), 19

`DeviceSelector` (*PySpin.TransportLayerInterface* property), 67

`DeviceSerialNumber` (*PySpin.Camera* property), 19

DeviceSerialNumber (PySpin.TransportLayerDevice property), 65	DeviceVersion (PySpin.TransportLayerDevice property), 66
DeviceSerialNumber (PySpin.TransportLayerInterface property), 67	DiscoverMaxPacketSize() (PySpin.CameraBase method), 34
DeviceSerialPortBaudRate (PySpin.Camera property), 19	<b>E</b>
DeviceSerialPortSelector (PySpin.Camera property), 19	EncoderDivider (PySpin.Camera property), 19
DeviceSFNCVersionMajor (PySpin.Camera property), 19	EncoderMode (PySpin.Camera property), 19
DeviceSFNCVersionMinor (PySpin.Camera property), 19	EncoderOutputMode (PySpin.Camera property), 19
DeviceSFNCVersionSubMinor (PySpin.Camera property), 19	EncoderReset (PySpin.Camera property), 19
DeviceStreamChannelCount (PySpin.Camera property), 19	EncoderResetActivation (PySpin.Camera property), 20
DeviceStreamChannelEndianness (PySpin.Camera property), 19	EncoderResetSource (PySpin.Camera property), 20
DeviceStreamChannelLink (PySpin.Camera property), 19	EncoderSelector (PySpin.Camera property), 20
DeviceStreamChannelPacketSize (PySpin.Camera property), 19	EncoderSourceA (PySpin.Camera property), 20
DeviceStreamChannelSelector (PySpin.Camera property), 19	EncoderSourceB (PySpin.Camera property), 20
DeviceStreamChannelType (PySpin.Camera property), 19	EncoderStatus (PySpin.Camera property), 20
DeviceTapGeometry (PySpin.Camera property), 19	EncoderTimeout (PySpin.Camera property), 20
DeviceTemperature (PySpin.Camera property), 19	EncoderValue (PySpin.Camera property), 20
DeviceTemperatureSelector (PySpin.Camera property), 19	EncoderValueAtReset (PySpin.Camera property), 20
DeviceTLType (PySpin.Camera property), 19	EndAcquisition() (PySpin.CameraBase method), 34
DeviceTLVersionMajor (PySpin.Camera property), 19	EnumerationCount (PySpin.Camera property), 20
DeviceTLVersionMinor (PySpin.Camera property), 19	errorcode (PySpin.SpinnakerException attribute), 58
DeviceTLVersionSubMinor (PySpin.Camera property), 19	EventAcquisitionEnd (PySpin.Camera property), 20
DeviceType (PySpin.Camera property), 19	EventAcquisitionEndFrameID (PySpin.Camera property), 20
DeviceType (PySpin.TransportLayerDevice property), 66	EventAcquisitionEndTimestamp (PySpin.Camera property), 20
DeviceU3VProtocol (PySpin.TransportLayerDevice property), 66	EventAcquisitionError (PySpin.Camera property), 20
DeviceUnlock (PySpin.TransportLayerInterface property), 67	EventAcquisitionErrorFrameID (PySpin.Camera property), 20
DeviceUpdateList (PySpin.TransportLayerInterface property), 67	EventAcquisitionErrorTimestamp (PySpin.Camera property), 20
DeviceUptime (PySpin.Camera property), 19	EventAcquisitionStart (PySpin.Camera property), 20
DeviceUserID (PySpin.Camera property), 19	EventAcquisitionStartFrameID (PySpin.Camera property), 20
DeviceUserID (PySpin.TransportLayerDevice property), 66	EventAcquisitionStartTimestamp (PySpin.Camera property), 20
DeviceVendorName (PySpin.Camera property), 19	EventAcquisitionTransferEnd (PySpin.Camera property), 20
DeviceVendorName (PySpin.TransportLayerDevice property), 66	EventAcquisitionTransferEndFrameID (PySpin.Camera property), 20
DeviceVendorName (PySpin.TransportLayerInterface property), 67	EventAcquisitionTransferEndTimestamp (PySpin.Camera property), 20
DeviceVersion (PySpin.Camera property), 19	EventAcquisitionTransferStart (PySpin.Camera property), 20
	EventAcquisitionTransferStartFrameID (PySpin.Camera property), 20
	EventAcquisitionTransferStartTimestamp (PySpin.Camera property), 20
	EventAcquisitionTrigger (PySpin.Camera property), 20

EventAcquisitionTriggerFrameID (*PySpin.Camera property*), 20

EventAcquisitionTriggerTimestamp (*PySpin.Camera property*), 20

EventActionLate (*PySpin.Camera property*), 20

EventActionLateFrameID (*PySpin.Camera property*), 20

EventActionLateTimestamp (*PySpin.Camera property*), 20

EventCounter0End (*PySpin.Camera property*), 21

EventCounter0EndFrameID (*PySpin.Camera property*), 21

EventCounter0EndTimestamp (*PySpin.Camera property*), 21

EventCounter0Start (*PySpin.Camera property*), 21

EventCounter0StartFrameID (*PySpin.Camera property*), 21

EventCounter0StartTimestamp (*PySpin.Camera property*), 21

EventCounter1End (*PySpin.Camera property*), 21

EventCounter1EndFrameID (*PySpin.Camera property*), 21

EventCounter1EndTimestamp (*PySpin.Camera property*), 21

EventCounter1Start (*PySpin.Camera property*), 21

EventCounter1StartFrameID (*PySpin.Camera property*), 21

EventCounter1StartTimestamp (*PySpin.Camera property*), 21

EventEncoder0Restarted (*PySpin.Camera property*), 21

EventEncoder0RestartedFrameID (*PySpin.Camera property*), 21

EventEncoder0RestartedTimestamp (*PySpin.Camera property*), 21

EventEncoder0Stopped (*PySpin.Camera property*), 21

EventEncoder0StoppedFrameID (*PySpin.Camera property*), 21

EventEncoder0StoppedTimestamp (*PySpin.Camera property*), 21

EventEncoder1Restarted (*PySpin.Camera property*), 21

EventEncoder1RestartedFrameID (*PySpin.Camera property*), 21

EventEncoder1RestartedTimestamp (*PySpin.Camera property*), 21

EventEncoder1Stopped (*PySpin.Camera property*), 21

EventEncoder1StoppedFrameID (*PySpin.Camera property*), 21

EventEncoder1StoppedTimestamp (*PySpin.Camera property*), 21

EventError (*PySpin.Camera property*), 21

EventErrorCode (*PySpin.Camera property*), 21

EventErrorFrameID (*PySpin.Camera property*), 21

EventErrorTimestamp (*PySpin.Camera property*), 21

EventExposureEnd (*PySpin.Camera property*), 21

EventExposureEndFrameID (*PySpin.Camera property*), 21

EventExposureEndTimestamp (*PySpin.Camera property*), 21

EventExposureStart (*PySpin.Camera property*), 22

EventExposureStartFrameID (*PySpin.Camera property*), 22

EventExposureStartTimestamp (*PySpin.Camera property*), 22

EventFrameBurstEnd (*PySpin.Camera property*), 22

EventFrameBurstEndFrameID (*PySpin.Camera property*), 22

EventFrameBurstEndTimestamp (*PySpin.Camera property*), 22

EventFrameBurstStart (*PySpin.Camera property*), 22

EventFrameBurstStartFrameID (*PySpin.Camera property*), 22

EventFrameBurstStartTimestamp (*PySpin.Camera property*), 22

EventFrameEnd (*PySpin.Camera property*), 22

EventFrameEndFrameID (*PySpin.Camera property*), 22

EventFrameEndTimestamp (*PySpin.Camera property*), 22

EventFrameStart (*PySpin.Camera property*), 22

EventFrameStartFrameID (*PySpin.Camera property*), 22

EventFrameStartTimestamp (*PySpin.Camera property*), 22

EventFrameTransferEnd (*PySpin.Camera property*), 22

EventFrameTransferEndFrameID (*PySpin.Camera property*), 22

EventFrameTransferEndTimestamp (*PySpin.Camera property*), 22

EventFrameTransferStart (*PySpin.Camera property*), 22

EventFrameTransferStartFrameID (*PySpin.Camera property*), 22

EventFrameTransferStartTimestamp (*PySpin.Camera property*), 22

EventFrameTrigger (*PySpin.Camera property*), 22

EventFrameTriggerFrameID (*PySpin.Camera property*), 22

EventFrameTriggerTimestamp (*PySpin.Camera property*), 22

EventHandler (*class in PySpin*), 6

EventLine0AnyEdge (*PySpin.Camera property*), 22

EventLine0AnyEdgeFrameID (*PySpin.Camera property*), 22

EventLine0AnyEdgeTimestamp (*PySpin.Camera property*), 22

EventLine0FallingEdge (*PySpin.Camera property*),



22		
EventLine0FallingEdgeFrameID	(PySpin.Camera property), 22	EventSerialPortReceiveTimestamp (PySpin.Camera property), 23
EventLine0FallingEdgeTimestamp	(PySpin.Camera property), 22	EventSerialReceiveOverflow (PySpin.Camera property), 23
EventLine0RisingEdge	(PySpin.Camera property), 22	EventStream0TransferBlockEnd (PySpin.Camera property), 23
EventLine0RisingEdgeFrameID	(PySpin.Camera property), 23	EventStream0TransferBlockEndFrameID (PySpin.Camera property), 24
EventLine0RisingEdgeTimestamp	(PySpin.Camera property), 23	EventStream0TransferBlockEndTimestamp (PySpin.Camera property), 24
EventLine1AnyEdge	(PySpin.Camera property), 23	EventStream0TransferBlockStart (PySpin.Camera property), 24
EventLine1AnyEdgeFrameID	(PySpin.Camera property), 23	EventStream0TransferBlockStartFrameID (PySpin.Camera property), 24
EventLine1AnyEdgeTimestamp	(PySpin.Camera property), 23	EventStream0TransferBlockStartTimestamp (PySpin.Camera property), 24
EventLine1FallingEdge	(PySpin.Camera property), 23	EventStream0TransferBlockTrigger (PySpin.Camera property), 24
EventLine1FallingEdgeFrameID	(PySpin.Camera property), 23	EventStream0TransferBlockTriggerFrameID (PySpin.Camera property), 24
EventLine1FallingEdgeTimestamp	(PySpin.Camera property), 23	EventStream0TransferBlockTriggerTimestamp (PySpin.Camera property), 24
EventLine1RisingEdge	(PySpin.Camera property), 23	EventStream0TransferBurstEnd (PySpin.Camera property), 24
EventLine1RisingEdgeFrameID	(PySpin.Camera property), 23	EventStream0TransferBurstEndFrameID (PySpin.Camera property), 24
EventLine1RisingEdgeTimestamp	(PySpin.Camera property), 23	EventStream0TransferBurstEndTimestamp (PySpin.Camera property), 24
EventLinkSpeedChange	(PySpin.Camera property), 23	EventStream0TransferBurstStart (PySpin.Camera property), 24
EventLinkSpeedChangeFrameID	(PySpin.Camera property), 23	EventStream0TransferBurstStartFrameID (PySpin.Camera property), 24
EventLinkSpeedChangeTimestamp	(PySpin.Camera property), 23	EventStream0TransferBurstStartTimestamp (PySpin.Camera property), 24
EventLinkTrigger0	(PySpin.Camera property), 23	EventStream0TransferEnd (PySpin.Camera property), 24
EventLinkTrigger0FrameID	(PySpin.Camera property), 23	EventStream0TransferEndFrameID (PySpin.Camera property), 24
EventLinkTrigger0Timestamp	(PySpin.Camera property), 23	EventStream0TransferEndTimestamp (PySpin.Camera property), 24
EventLinkTrigger1	(PySpin.Camera property), 23	EventStream0TransferOverflow (PySpin.Camera property), 24
EventLinkTrigger1FrameID	(PySpin.Camera property), 23	EventStream0TransferOverflowFrameID (PySpin.Camera property), 24
EventLinkTrigger1Timestamp	(PySpin.Camera property), 23	EventStream0TransferOverflowTimestamp (PySpin.Camera property), 24
EventNotification	(PySpin.Camera property), 23	EventStream0TransferPause (PySpin.Camera property), 24
EventSelector	(PySpin.Camera property), 23	EventStream0TransferPauseFrameID (PySpin.Camera property), 24
EventSequencerSetChange	(PySpin.Camera property), 23	EventStream0TransferPauseTimestamp (PySpin.Camera property), 24
EventSequencerSetChangeFrameID	(PySpin.Camera property), 23	EventStream0TransferResume (PySpin.Camera property), 24
EventSequencerSetChangeTimestamp	(PySpin.Camera property), 23	
EventSerialData	(PySpin.Camera property), 23	
EventSerialDataLength	(PySpin.Camera property), 23	
EventSerialPortReceive	(PySpin.Camera property), 23	

EventStream0TransferResumeFrameID  
(*PySpin.Camera* property), 24

EventStream0TransferResumeTimestamp  
(*PySpin.Camera* property), 24

EventStream0TransferStart (*PySpin.Camera* property), 24

EventStream0TransferStartFrameID  
(*PySpin.Camera* property), 24

EventStream0TransferStartTimestamp  
(*PySpin.Camera* property), 24

EventTest (*PySpin.Camera* property), 24

EventTestTimestamp (*PySpin.Camera* property), 24

EventTimer0End (*PySpin.Camera* property), 25

EventTimer0EndFrameID (*PySpin.Camera* property), 25

EventTimer0EndTimestamp (*PySpin.Camera* property), 25

EventTimer0Start (*PySpin.Camera* property), 25

EventTimer0StartFrameID (*PySpin.Camera* property), 25

EventTimer0StartTimestamp (*PySpin.Camera* property), 25

EventTimer1End (*PySpin.Camera* property), 25

EventTimer1EndFrameID (*PySpin.Camera* property), 25

EventTimer1EndTimestamp (*PySpin.Camera* property), 25

EventTimer1Start (*PySpin.Camera* property), 25

EventTimer1StartFrameID (*PySpin.Camera* property), 25

EventTimer1StartTimestamp (*PySpin.Camera* property), 25

ExposureActiveMode (*PySpin.Camera* property), 25

ExposureAuto (*PySpin.Camera* property), 25

ExposureMode (*PySpin.Camera* property), 25

ExposureTime (*PySpin.Camera* property), 25

ExposureTimeMode (*PySpin.Camera* property), 25

ExposureTimeSelector (*PySpin.Camera* property), 25

## F

FactoryReset (*PySpin.Camera* property), 25

FileAccessBuffer (*PySpin.Camera* property), 25

FileAccessLength (*PySpin.Camera* property), 25

FileAccessOffset (*PySpin.Camera* property), 25

FileOpenMode (*PySpin.Camera* property), 25

FileOperationExecute (*PySpin.Camera* property), 25

FileOperationResult (*PySpin.Camera* property), 25

FileOperationSelector (*PySpin.Camera* property), 25

FileOperationStatus (*PySpin.Camera* property), 25

FileSelector (*PySpin.Camera* property), 25

FileSize (*PySpin.Camera* property), 25

FilterDriverStatus (*PySpin.TransportLayerInterface* property), 67

ForceIPC (*PySpin.CameraBase* method), 34

fullmessage (*PySpin.SpinnakerException* attribute), 58

## G

Gain (*PySpin.Camera* property), 25

GainAuto (*PySpin.Camera* property), 25

GainAutoBalance (*PySpin.Camera* property), 26

GainSelector (*PySpin.Camera* property), 26

Gamma (*PySpin.Camera* property), 26

GammaEnable (*PySpin.Camera* property), 26

GenICamXMLLocation (*PySpin.TransportLayerDevice* property), 66

GenICamXMLPath (*PySpin.TransportLayerDevice* property), 66

GetAccessMode() (*PySpin.CameraBase* method), 34

GetAccessMode() (*PySpin.CBasePtr* method), 11

GetBitsPerPixel() (*PySpin.Image* method), 46

GetBlackLevel() (*PySpin.ChunkData* method), 41

GetBufferOwnership() (*PySpin.CameraBase* method), 35

GetBufferSize() (*PySpin.Image* method), 46

GetByDeviceID() (*PySpin.CameraList* method), 39

GetByIndex() (*PySpin.CameraList* method), 39

GetByIndex() (*PySpin.ImageList* method), 54

GetByIndex() (*PySpin.InterfaceList* method), 57

GetByInterfaceID() (*PySpin.InterfaceList* method), 58

GetByPixelFormat() (*PySpin.ImageList* method), 54

GetBySerial() (*PySpin.CameraList* method), 39

GetCameras() (*PySpin.IInterface* method), 56

GetCameras() (*PySpin.System* method), 60

GetChunkData() (*PySpin.Image* method), 46

GetChunkLayoutId() (*PySpin.Image* method), 46

GetColorProcessing() (*PySpin.Image* method), 47

GetColorProcessing() (*PySpin.ImageProcessor* method), 56

GetCompressionMode() (*PySpin.ChunkData* method), 41

GetCompressionRatio() (*PySpin.ChunkData* method), 41

GetCounterValue() (*PySpin.ChunkData* method), 41

GetCRC() (*PySpin.ChunkData* method), 41

GetDataAbsoluteMax() (*PySpin.Image* method), 47

GetDataAbsoluteMin() (*PySpin.Image* method), 47

GetDeviceEventId() (*PySpin.DeviceEventHandler* method), 6

GetDeviceEventName() (*PySpin.DeviceEventHandler* method), 6

GetEncoderValue() (*PySpin.ChunkData* method), 42

GetEventPayloadData() (*PySpin.EventHandler* method), 6

GetEventPayloadDataSize() (*PySpin.EventHandler* method), 6

GetEventType() (*PySpin.EventHandler* method), 6

GetExposureEndLineStatusAll() *(PySpin.ChunkData method)*, 42  
 GetExposureTime() *(PySpin.ChunkData method)*, 42  
 GetFrameID() *(PySpin.ChunkData method)*, 42  
 GetFrameID() *(PySpin.Image method)*, 47  
 GetGain() *(PySpin.ChunkData method)*, 42  
 GetGuiXml() *(PySpin.CameraBase method)*, 35  
 GetHeight() *(PySpin.ChunkData method)*, 42  
 GetHeight() *(PySpin.Image method)*, 47  
 GetID() *(PySpin.Image method)*, 47  
 GetImage() *(PySpin.ChunkData method)*, 42  
 GetImageSize() *(PySpin.Image method)*, 47  
 GetImageStatus() *(PySpin.Image method)*, 47  
 GetImageStatusDescription() *(PySpin.Image static method)*, 47  
 GetInferenceBoundingBoxResult() *(PySpin.ChunkData method)*, 42  
 GetInferenceConfidence() *(PySpin.ChunkData method)*, 42  
 GetInferenceFrameId() *(PySpin.ChunkData method)*, 42  
 GetInferenceResult() *(PySpin.ChunkData method)*, 42  
 GetInstance() *(PySpin.System static method)*, 61  
 GetInterfaces() *(PySpin.System method)*, 61  
 GetLibraryVersion() *(PySpin.System method)*, 61  
 GetLinePitch() *(PySpin.ChunkData method)*, 42  
 GetLineStatusAll() *(PySpin.ChunkData method)*, 42  
 GetLoggingEventPriorityLevel() *(PySpin.System method)*, 61  
 GetNextImage() *(PySpin.CameraBase method)*, 35  
 GetNextImageSync() *(PySpin.CameraBase method)*, 35  
 GetNodeMap() *(PySpin.CameraBase method)*, 36  
 GetNumChannels() *(PySpin.Image method)*, 47  
 GetNumDataStreams() *(PySpin.CameraBase method)*, 36  
 GetNumDecompressionThreads() *(PySpin.ImageProcessor method)*, 56  
 GetNumImagesInUse() *(PySpin.CameraBase method)*, 36  
 GetOffsetX() *(PySpin.ChunkData method)*, 42  
 GetOffsetY() *(PySpin.ChunkData method)*, 42  
 GetPartSelector() *(PySpin.ChunkData method)*, 43  
 GetPayloadType() *(PySpin.Image method)*, 47  
 GetPixelDynamicRangeMax() *(PySpin.ChunkData method)*, 43  
 GetPixelDynamicRangeMin() *(PySpin.ChunkData method)*, 43  
 GetPixelFormat() *(PySpin.Image method)*, 48  
 GetPixelFormatIntType() *(PySpin.Image method)*, 48  
 GetPixelFormatName() *(PySpin.Image method)*, 48  
 GetPrivateData() *(PySpin.Image method)*, 48  
 GetScan3dAxisMax() *(PySpin.ChunkData method)*, 43  
 GetScan3dAxisMin() *(PySpin.ChunkData method)*, 43  
 GetScan3dCoordinateOffset() *(PySpin.ChunkData method)*, 43  
 GetScan3dCoordinateReferenceValue() *(PySpin.ChunkData method)*, 43  
 GetScan3dCoordinateScale() *(PySpin.ChunkData method)*, 43  
 GetScan3dInvalidDataValue() *(PySpin.ChunkData method)*, 43  
 GetScan3dTransformValue() *(PySpin.ChunkData method)*, 43  
 GetScanLineSelector() *(PySpin.ChunkData method)*, 43  
 GetSequencerSetActive() *(PySpin.ChunkData method)*, 44  
 GetSerialDataLength() *(PySpin.ChunkData method)*, 44  
 GetSize() *(PySpin.CameraList method)*, 39  
 GetSize() *(PySpin.ImageList method)*, 54  
 GetSize() *(PySpin.InterfaceList method)*, 58  
 GetStreamChannelID() *(PySpin.ChunkData method)*, 44  
 GetStreamIndex() *(PySpin.Image method)*, 48  
 GetStride() *(PySpin.Image method)*, 48  
 GetTimerValue() *(PySpin.ChunkData method)*, 44  
 GetTimestamp() *(PySpin.ChunkData method)*, 44  
 GetTimeStamp() *(PySpin.Image method)*, 49  
 GetTimestampLatchValue() *(PySpin.ChunkData method)*, 44  
 GetTLDeviceNodeMap() *(PySpin.CameraBase method)*, 36  
 GetTLNodeMap() *(PySpin.Interface method)*, 56  
 GetTLNodeMap() *(PySpin.System method)*, 62  
 GetTLPayloadType() *(PySpin.Image method)*, 48  
 GetTLPixelFormat() *(PySpin.Image method)*, 48  
 GetTLPixelFormatNamespace() *(PySpin.Image method)*, 49  
 GetTLStreamNodeMap() *(PySpin.CameraBase method)*, 36  
 GetTransferBlockID() *(PySpin.ChunkData method)*, 44  
 GetTransferQueueCurrentBlockCount() *(PySpin.ChunkData method)*, 44  
 GetUniqueID() *(PySpin.CameraBase method)*, 36  
 GetUserBufferCount() *(PySpin.CameraBase method)*, 36  
 GetUserBufferSize() *(PySpin.CameraBase method)*, 36  
 GetUserBufferTotalSize() *(PySpin.CameraBase method)*, 37  
 GetValidPayloadSize() *(PySpin.Image method)*, 49  
 GetWidth() *(PySpin.ChunkData method)*, 44  
 GetWidth() *(PySpin.Image method)*, 49

<code>GetXOffset()</code> ( <i>PySpin.Image</i> method), 49	<code>GevDeviceGateway</code> ( <i>PySpin.TransportLayerDevice</i> property), 66
<code>GetXPadding()</code> ( <i>PySpin.Image</i> method), 49	<code>GevDeviceGateway</code> ( <i>PySpin.TransportLayerInterface</i> property), 67
<code>GetYOffset()</code> ( <i>PySpin.Image</i> method), 50	<code>GevDeviceIPAddress</code> ( <i>PySpin.TransportLayerDevice</i> property), 66
<code>GetYPadding()</code> ( <i>PySpin.Image</i> method), 50	<code>GevDeviceIPAddress</code> ( <i>PySpin.TransportLayerInterface</i> property), 67
<code>GevActionDeviceKey</code> ( <i>PySpin.TransportLayerInterface</i> property), 67	<code>GevDeviceIsWrongSubnet</code> ( <i>PySpin.TransportLayerDevice</i> property), 66
<code>GevActionGroupKey</code> ( <i>PySpin.TransportLayerInterface</i> property), 67	<code>GevDeviceMACAddress</code> ( <i>PySpin.TransportLayerDevice</i> property), 66
<code>GevActionGroupMask</code> ( <i>PySpin.TransportLayerInterface</i> property), 67	<code>GevDeviceMACAddress</code> ( <i>PySpin.TransportLayerInterface</i> property), 67
<code>GevActionTime</code> ( <i>PySpin.TransportLayerInterface</i> property), 67	<code>GevDeviceMaximumPacketSize</code> ( <i>PySpin.TransportLayerDevice</i> property), 66
<code>GevActiveLinkCount</code> ( <i>PySpin.Camera</i> property), 26	<code>GevDeviceMaximumRetryCount</code> ( <i>PySpin.TransportLayerDevice</i> property), 66
<code>GevCCP</code> ( <i>PySpin.Camera</i> property), 26	<code>GevDeviceModeIsBigEndian</code> ( <i>PySpin.TransportLayerDevice</i> property), 66
<code>GevCCP</code> ( <i>PySpin.TransportLayerDevice</i> property), 66	<code>GevDevicePort</code> ( <i>PySpin.TransportLayerDevice</i> property), 66
<code>GevCurrentDefaultGateway</code> ( <i>PySpin.Camera</i> property), 26	<code>GevDeviceReadAndWriteTimeout</code> ( <i>PySpin.TransportLayerDevice</i> property), 66
<code>GevCurrentIPAddress</code> ( <i>PySpin.Camera</i> property), 26	<code>GevDeviceSubnetMask</code> ( <i>PySpin.TransportLayerDevice</i> property), 66
<code>GevCurrentIPConfigurationDHCP</code> ( <i>PySpin.Camera</i> property), 26	<code>GevDeviceSubnetMask</code> ( <i>PySpin.TransportLayerInterface</i> property), 67
<code>GevCurrentIPConfigurationLLA</code> ( <i>PySpin.Camera</i> property), 26	<code>GevDiscoveryAckDelay</code> ( <i>PySpin.Camera</i> property), 26
<code>GevCurrentIPConfigurationPersistentIP</code> ( <i>PySpin.Camera</i> property), 26	<code>GevFirstURL</code> ( <i>PySpin.Camera</i> property), 26
<code>GevCurrentPhysicalLinkConfiguration</code> ( <i>PySpin.Camera</i> property), 26	<code>GevGVCPExtendedStatusCodes</code> ( <i>PySpin.Camera</i> property), 26
<code>GevCurrentSubnetMask</code> ( <i>PySpin.Camera</i> property), 26	<code>GevGVCPExtendedStatusCodesSelector</code> ( <i>PySpin.Camera</i> property), 26
<code>GevDeviceAutoForceIP</code> ( <i>PySpin.TransportLayerDevice</i> property), 66	<code>GevGVCPHeartbeatDisable</code> ( <i>PySpin.Camera</i> property), 26
<code>GevDeviceAutoForceIP</code> ( <i>PySpin.TransportLayerInterface</i> property), 67	<code>GevGVCPPendingAck</code> ( <i>PySpin.Camera</i> property), 26
<code>GevDeviceDiscoverMaximumPacketSize</code> ( <i>PySpin.TransportLayerDevice</i> property), 66	<code>GevGVCPPendingTimeout</code> ( <i>PySpin.Camera</i> property), 26
<code>GevDeviceForceGateway</code> ( <i>PySpin.TransportLayerDevice</i> property), 66	<code>GevGVSPExtendedIDMode</code> ( <i>PySpin.Camera</i> property), 26
<code>GevDeviceForceGateway</code> ( <i>PySpin.TransportLayerInterface</i> property), 67	<code>GevHeartbeatTimeout</code> ( <i>PySpin.Camera</i> property), 26
<code>GevDeviceForceIP</code> ( <i>PySpin.TransportLayerDevice</i> property), 66	<code>GevIEEE1588</code> ( <i>PySpin.Camera</i> property), 26
<code>GevDeviceForceIP</code> ( <i>PySpin.TransportLayerInterface</i> property), 67	<code>GevIEEE1588ClockAccuracy</code> ( <i>PySpin.Camera</i> property), 26
<code>GevDeviceForceIPAddress</code> ( <i>PySpin.TransportLayerDevice</i> property), 66	<code>GevIEEE1588Mode</code> ( <i>PySpin.Camera</i> property), 26
<code>GevDeviceForceIPAddress</code> ( <i>PySpin.TransportLayerInterface</i> property), 67	<code>GevIEEE1588Status</code> ( <i>PySpin.Camera</i> property), 26
<code>GevDeviceForceSubnetMask</code> ( <i>PySpin.TransportLayerDevice</i> property), 66	<code>GevInterfaceGateway</code> ( <i>PySpin.TransportLayerInterface</i> property), 67
<code>GevDeviceForceSubnetMask</code> ( <i>PySpin.TransportLayerInterface</i> property), 67	



- `GevInterfaceGatewaySelector` (*PySpin.TransportLayerInterface* property), 67  
`GevInterfaceMACAddress` (*PySpin.TransportLayerInterface* property), 67  
`GevInterfaceMTU` (*PySpin.TransportLayerInterface* property), 67  
`GevInterfaceReceiveLinkSpeed` (*PySpin.TransportLayerInterface* property), 68  
`GevInterfaceSelector` (*PySpin.Camera* property), 26  
`GevInterfaceSubnetIPAddress` (*PySpin.TransportLayerInterface* property), 68  
`GevInterfaceSubnetMask` (*PySpin.TransportLayerInterface* property), 68  
`GevInterfaceSubnetSelector` (*PySpin.TransportLayerInterface* property), 68  
`GevInterfaceTransmitLinkSpeed` (*PySpin.TransportLayerInterface* property), 68  
`GevIPConfigurationStatus` (*PySpin.Camera* property), 26  
`GevMACAddress` (*PySpin.Camera* property), 26  
`GevMCDA` (*PySpin.Camera* property), 26  
`GevMCPHostPort` (*PySpin.Camera* property), 26  
`GevMCRC` (*PySpin.Camera* property), 27  
`GevMCSP` (*PySpin.Camera* property), 27  
`GevMCTT` (*PySpin.Camera* property), 27  
`GevNumberOfInterfaces` (*PySpin.Camera* property), 27  
`GevPAUSEFrameReception` (*PySpin.Camera* property), 27  
`GevPAUSEFrameTransmission` (*PySpin.Camera* property), 27  
`GevPersistentDefaultGateway` (*PySpin.Camera* property), 27  
`GevPersistentIPAddress` (*PySpin.Camera* property), 27  
`GevPersistentSubnetMask` (*PySpin.Camera* property), 27  
`GevPhysicalLinkConfiguration` (*PySpin.Camera* property), 27  
`GevPrimaryApplicationIPAddress` (*PySpin.Camera* property), 27  
`GevPrimaryApplicationSocket` (*PySpin.Camera* property), 27  
`GevPrimaryApplicationSwitchoverKey` (*PySpin.Camera* property), 27  
`GevSCCFGAllInTransmission` (*PySpin.Camera* property), 27  
`GevSCCFGExtendedChunkData` (*PySpin.Camera* property), 27  
`GevSCCFGPacketResendDestination` (*PySpin.Camera* property), 27  
`GevSCCFGUnconditionalStreaming` (*PySpin.Camera* property), 27  
`GevSCDA` (*PySpin.Camera* property), 27  
`GevSCPD` (*PySpin.Camera* property), 27  
`GevSCPDDirection` (*PySpin.Camera* property), 27  
`GevSCPHostPort` (*PySpin.Camera* property), 27  
`GevSCPInterfaceIndex` (*PySpin.Camera* property), 27  
`GevSCPSBigEndian` (*PySpin.Camera* property), 27  
`GevSCPSDoNotFragment` (*PySpin.Camera* property), 27  
`GevSCPSFireTestPacket` (*PySpin.Camera* property), 27  
`GevSCPSPacketSize` (*PySpin.Camera* property), 27  
`GevSCSP` (*PySpin.Camera* property), 27  
`GevSCZoneConfigurationLock` (*PySpin.Camera* property), 27  
`GevSCZoneCount` (*PySpin.Camera* property), 27  
`GevSCZoneDirectionAll` (*PySpin.Camera* property), 27  
`GevSecondURL` (*PySpin.Camera* property), 27  
`GevStreamChannelSelector` (*PySpin.Camera* property), 28  
`GevSupportedOption` (*PySpin.Camera* property), 28  
`GevSupportedOptionSelector` (*PySpin.Camera* property), 28  
`GevTimestampTickFrequency` (*PySpin.Camera* property), 28  
`GevVersionMajor` (*PySpin.TransportLayerDevice* property), 66  
`GevVersionMinor` (*PySpin.TransportLayerDevice* property), 66  
`GUIXMLLocation` (*PySpin.TransportLayerDevice* property), 66  
`GuiXmlManifestAddress` (*PySpin.Camera* property), 28  
`GUIXMLPath` (*PySpin.TransportLayerDevice* property), 66
- ## H
- `HasChunkData()` (*PySpin.Image* method), 50  
`HasCRC()` (*PySpin.Image* method), 50  
`Height` (*PySpin.Camera* property), 28  
`HeightMax` (*PySpin.Camera* property), 28  
`histogram` (*PySpin.ChannelStatistics* property), 41  
`HostAdapterDriverVersion` (*PySpin.TransportLayerInterface* property), 68  
`HostAdapterName` (*PySpin.TransportLayerInterface* property), 68  
`HostAdapterVendor` (*PySpin.TransportLayerInterface* property), 68
- ## I
- `IInterface` (class in *PySpin*), 56  
`Image` (class in *PySpin*), 45  
`ImageComponentEnable` (*PySpin.Camera* property), 28  
`ImageComponentSelector` (*PySpin.Camera* property), 28

ImageCompressionBitrate (*PySpin.Camera* property), 28  
ImageCompressionJPEGFormatOption (*PySpin.Camera* property), 28  
ImageCompressionMode (*PySpin.Camera* property), 28  
ImageCompressionQuality (*PySpin.Camera* property), 28  
ImageCompressionRateOption (*PySpin.Camera* property), 28  
ImageEventHandler (class in *PySpin*), 7  
ImageList (class in *PySpin*), 54  
ImageListEventHandler (class in *PySpin*), 7  
ImageProcessor (class in *PySpin*), 55  
ImagePtr (class in *PySpin*), 56  
IncompatibleDeviceCount (*PySpin.TransportLayerInterface* property), 68  
IncompatibleDeviceID (*PySpin.TransportLayerInterface* property), 68  
IncompatibleDeviceModelName (*PySpin.TransportLayerInterface* property), 68  
IncompatibleDeviceSelector (*PySpin.TransportLayerInterface* property), 68  
IncompatibleDeviceVendorName (*PySpin.TransportLayerInterface* property), 68  
IncompatibleGevDeviceIPAddress (*PySpin.TransportLayerInterface* property), 68  
IncompatibleGevDeviceMACAddress (*PySpin.TransportLayerInterface* property), 68  
IncompatibleGevDeviceSubnetMask (*PySpin.TransportLayerInterface* property), 68  
Init() (*PySpin.Camera* method), 28  
Init() (*PySpin.CameraBase* method), 37  
InterfaceArrivalEventHandler (class in *PySpin*), 7  
InterfaceDisplayName (*PySpin.TransportLayerInterface* property), 68  
InterfaceEventHandler (class in *PySpin*), 7  
InterfaceID (*PySpin.TransportLayerInterface* property), 68  
InterfaceList (class in *PySpin*), 57  
InterfacePtr (class in *PySpin*), 58  
InterfaceRemovalEventHandler (class in *PySpin*), 8  
InterfaceType (*PySpin.TransportLayerInterface* property), 68  
IsCompressed() (*PySpin.Image* method), 50  
IsIncomplete() (*PySpin.Image* method), 50  
IsInitialized() (*PySpin.CameraBase* method), 37  
IsInUse() (*PySpin.IInterface* method), 56  
IsInUse() (*PySpin.Image* method), 50  
IsInUse() (*PySpin.System* method), 62  
IspEnable (*PySpin.Camera* property), 28  
IsStreaming() (*PySpin.CameraBase* method), 37  
IsValid() (*PySpin.CameraBase* method), 37  
IsValid() (*PySpin.CBasePtr* method), 11  
IsValid() (*PySpin.IInterface* method), 56

## L

LineFilterWidth (*PySpin.Camera* property), 28  
LineFormat (*PySpin.Camera* property), 28  
LineInputFilterSelector (*PySpin.Camera* property), 28  
LineInverter (*PySpin.Camera* property), 28  
LineMode (*PySpin.Camera* property), 28  
LinePitch (*PySpin.Camera* property), 28  
LineSelector (*PySpin.Camera* property), 28  
LineSource (*PySpin.Camera* property), 28  
LineStatus (*PySpin.Camera* property), 28  
LineStatusAll (*PySpin.Camera* property), 29  
LinkErrorCount (*PySpin.Camera* property), 29  
LinkUptime (*PySpin.Camera* property), 29  
Load() (*PySpin.Image* static method), 50  
Load() (*PySpin.ImageList* static method), 54  
LoggingEventDataPtr (class in *PySpin*), 8  
LoggingEventHandler (class in *PySpin*), 8  
LogicBlockLUTInputActivation (*PySpin.Camera* property), 29  
LogicBlockLUTInputSelector (*PySpin.Camera* property), 29  
LogicBlockLUTInputSource (*PySpin.Camera* property), 29  
LogicBlockLUTOutputValue (*PySpin.Camera* property), 29  
LogicBlockLUTOutputValueAll (*PySpin.Camera* property), 29  
LogicBlockLUTRowIndex (*PySpin.Camera* property), 29  
LogicBlockLUTSelector (*PySpin.Camera* property), 29  
LogicBlockSelector (*PySpin.Camera* property), 29  
LUTEnable (*PySpin.Camera* property), 28  
LUTIndex (*PySpin.Camera* property), 28  
LUTSelector (*PySpin.Camera* property), 28  
LUTValue (*PySpin.Camera* property), 28  
LUTValueAll (*PySpin.Camera* property), 28

## M

MaxDeviceResetTime (*PySpin.Camera* property), 29  
message (*PySpin.SpinnakerException* attribute), 58  
module  
    *PySpin*, 71

## N

num\_pixel\_values (*PySpin.ChannelStatistics* property), 41

## O

OffsetX (*PySpin.Camera* property), 29  
OffsetY (*PySpin.Camera* property), 29

OnDeviceArrival() (*PySpin.DeviceArrivalEventHandler method*), 5  
 OnDeviceArrival() (*PySpin.InterfaceEventHandler method*), 7  
 OnDeviceEvent() (*PySpin.DeviceEventHandler method*), 6  
 OnDeviceRemoval() (*PySpin.DeviceRemovalEventHandler method*), 6  
 OnDeviceRemoval() (*PySpin.InterfaceEventHandler method*), 7  
 OnImageEvent() (*PySpin.ImageEventHandler method*), 7  
 OnImageListEvent() (*PySpin.ImageListEventHandler method*), 7  
 OnInterfaceArrival() (*PySpin.InterfaceArrivalEventHandler method*), 7  
 OnInterfaceArrival() (*PySpin.SystemEventHandler method*), 9  
 OnInterfaceRemoval() (*PySpin.InterfaceRemovalEventHandler method*), 8  
 OnInterfaceRemoval() (*PySpin.SystemEventHandler method*), 9  
 OnLogEvent() (*PySpin.LoggingEventHandler method*), 8  
 Open() (*PySpin.SpinVideo method*), 59

**P**

PacketResendRequestCount (*PySpin.Camera property*), 29  
 PayloadSize (*PySpin.Camera property*), 29  
 pixel\_value\_max (*PySpin.ChannelStatistics property*), 41  
 pixel\_value\_mean (*PySpin.ChannelStatistics property*), 41  
 pixel\_value\_min (*PySpin.ChannelStatistics property*), 41  
 PixelColorFilter (*PySpin.Camera property*), 29  
 PixelDynamicRangeMax (*PySpin.Camera property*), 29  
 PixelDynamicRangeMin (*PySpin.Camera property*), 29  
 PixelFormat (*PySpin.Camera property*), 29  
 PixelFormatInfoID (*PySpin.Camera property*), 29  
 PixelFormatInfoSelector (*PySpin.Camera property*), 29  
 PixelSize (*PySpin.Camera property*), 29  
 POEStatus (*PySpin.TransportLayerInterface property*), 68  
 PowerSupplyCurrent (*PySpin.Camera property*), 29  
 PowerSupplyVoltage (*PySpin.Camera property*), 29  
 PySpin module, 71

**R**

range\_max (*PySpin.ChannelStatistics property*), 41  
 range\_min (*PySpin.ChannelStatistics property*), 41  
 RegionDestination (*PySpin.Camera property*), 29  
 RegionMode (*PySpin.Camera property*), 29  
 RegionSelector (*PySpin.Camera property*), 29  
 RegisterEventHandler() (*PySpin.CameraBase method*), 37  
 RegisterEventHandler() (*PySpin.IInterface method*), 56  
 RegisterEventHandler() (*PySpin.System method*), 62  
 RegisterLoggingEventHandler() (*PySpin.System method*), 62  
 Release() (*PySpin.Image method*), 50  
 Release() (*PySpin.ImageList method*), 54  
 ReleaseInstance() (*PySpin.System method*), 62  
 Remove() (*PySpin.CameraList method*), 39  
 Remove() (*PySpin.InterfaceList method*), 58  
 RemoveByDeviceID() (*PySpin.CameraList method*), 40  
 RemoveByIndex() (*PySpin.CameraList method*), 40  
 RemoveByIndex() (*PySpin.ImageList method*), 54  
 RemoveByPixelFormat() (*PySpin.ImageList method*), 54  
 RemoveBySerial() (*PySpin.CameraList method*), 40  
 ResetImage() (*PySpin.Image method*), 50  
 ReverseX (*PySpin.Camera property*), 29  
 ReverseY (*PySpin.Camera property*), 29  
 RgbTransformLightSource (*PySpin.Camera property*), 29

**S**

Saturation (*PySpin.Camera property*), 30  
 SaturationEnable (*PySpin.Camera property*), 30  
 Save() (*PySpin.Image method*), 52  
 Save() (*PySpin.ImageList method*), 54  
 Scan3dAxisMax (*PySpin.Camera property*), 30  
 Scan3dAxisMin (*PySpin.Camera property*), 30  
 Scan3dCoordinateOffset (*PySpin.Camera property*), 30  
 Scan3dCoordinateReferenceSelector (*PySpin.Camera property*), 30  
 Scan3dCoordinateReferenceValue (*PySpin.Camera property*), 30  
 Scan3dCoordinateScale (*PySpin.Camera property*), 30  
 Scan3dCoordinateSelector (*PySpin.Camera property*), 30  
 Scan3dCoordinateSystem (*PySpin.Camera property*), 30  
 Scan3dCoordinateSystemReference (*PySpin.Camera property*), 30  
 Scan3dCoordinateTransformSelector (*PySpin.Camera property*), 30  
 Scan3dDistanceUnit (*PySpin.Camera property*), 30

Scan3dInvalidDataFlag ( <i>PySpin.Camera</i> property), 30	( <i>PySpin.Camera</i> property), 31
Scan3dInvalidDataValue ( <i>PySpin.Camera</i> property), 30	SetBufferOwnership() ( <i>PySpin.CameraBase</i> method), 38
Scan3dOutputMode ( <i>PySpin.Camera</i> property), 30	SetChunks() ( <i>PySpin.ChunkData</i> method), 44
Scan3dTransformValue ( <i>PySpin.Camera</i> property), 30	SetColorProcessing() ( <i>PySpin.ImageProcessor</i> method), 56
SendActionCommand() ( <i>PySpin.IInterface</i> method), 57	SetEventType() ( <i>PySpin.EventHandler</i> method), 6
SendActionCommand() ( <i>PySpin.System</i> method), 62	SetLoggingEventPriorityLevel() ( <i>PySpin.System</i> method), 63
SensorDescription ( <i>PySpin.Camera</i> property), 30	SetMaximumFileSize() ( <i>PySpin.SpinVideo</i> method), 60
SensorDigitizationTaps ( <i>PySpin.Camera</i> property), 30	SetNumDecompressionThreads() ( <i>PySpin.ImageProcessor</i> method), 56
SensorHeight ( <i>PySpin.Camera</i> property), 30	SetUserBuffers() ( <i>PySpin.CameraBase</i> method), 38
SensorShutterMode ( <i>PySpin.Camera</i> property), 30	Sharpening ( <i>PySpin.Camera</i> property), 31
SensorTaps ( <i>PySpin.Camera</i> property), 30	SharpeningAuto ( <i>PySpin.Camera</i> property), 31
SensorWidth ( <i>PySpin.Camera</i> property), 30	SharpeningEnable ( <i>PySpin.Camera</i> property), 31
SequencerConfigurationMode ( <i>PySpin.Camera</i> property), 30	SharpeningThreshold ( <i>PySpin.Camera</i> property), 31
SequencerConfigurationValid ( <i>PySpin.Camera</i> property), 30	SoftwareSignalPulse ( <i>PySpin.Camera</i> property), 31
SequencerFeatureEnable ( <i>PySpin.Camera</i> property), 30	SoftwareSignalSelector ( <i>PySpin.Camera</i> property), 31
SequencerMode ( <i>PySpin.Camera</i> property), 30	SourceCount ( <i>PySpin.Camera</i> property), 31
SequencerPathSelector ( <i>PySpin.Camera</i> property), 30	SourceSelector ( <i>PySpin.Camera</i> property), 31
SequencerSetActive ( <i>PySpin.Camera</i> property), 30	SpinnakerException (class in <i>PySpin</i> ), 58
SequencerSetLoad ( <i>PySpin.Camera</i> property), 30	SpinVideo (class in <i>PySpin</i> ), 59
SequencerSetNext ( <i>PySpin.Camera</i> property), 30	StreamAnnounceBufferMinimum ( <i>PySpin.TransportLayerStream</i> property), 68
SequencerSetSave ( <i>PySpin.Camera</i> property), 31	StreamAnnouncedBufferCount ( <i>PySpin.TransportLayerStream</i> property), 68
SequencerSetSelector ( <i>PySpin.Camera</i> property), 31	StreamBlockTransferSize ( <i>PySpin.TransportLayerStream</i> property), 68
SequencerSetStart ( <i>PySpin.Camera</i> property), 31	StreamBufferAlignment ( <i>PySpin.TransportLayerStream</i> property), 68
SequencerSetValid ( <i>PySpin.Camera</i> property), 31	StreamBufferCountManual ( <i>PySpin.TransportLayerStream</i> property), 69
SequencerTriggerActivation ( <i>PySpin.Camera</i> property), 31	StreamBufferCountMax ( <i>PySpin.TransportLayerStream</i> property), 69
SequencerTriggerSource ( <i>PySpin.Camera</i> property), 31	StreamBufferCountMode ( <i>PySpin.TransportLayerStream</i> property), 69
SerialPortBaudRate ( <i>PySpin.Camera</i> property), 31	StreamBufferCountResult ( <i>PySpin.TransportLayerStream</i> property), 69
SerialPortDataBits ( <i>PySpin.Camera</i> property), 31	StreamBufferHandlingMode ( <i>PySpin.TransportLayerStream</i> property), 69
SerialPortParity ( <i>PySpin.Camera</i> property), 31	StreamChunkCountMaximum ( <i>PySpin.TransportLayerStream</i> property),
SerialPortSelector ( <i>PySpin.Camera</i> property), 31	
SerialPortSource ( <i>PySpin.Camera</i> property), 31	
SerialPortStopBits ( <i>PySpin.Camera</i> property), 31	
SerialReceiveFramingErrorCount ( <i>PySpin.Camera</i> property), 31	
SerialReceiveParityErrorCount ( <i>PySpin.Camera</i> property), 31	
SerialReceiveQueueClear ( <i>PySpin.Camera</i> property), 31	
SerialReceiveQueueCurrentCharacterCount ( <i>PySpin.Camera</i> property), 31	
SerialReceiveQueueMaxCharacterCount ( <i>PySpin.Camera</i> property), 31	
SerialTransmitQueueCurrentCharacterCount ( <i>PySpin.Camera</i> property), 31	
SerialTransmitQueueMaxCharacterCount	



69		StreamReceivedPacketCount	
StreamCRCCheckEnable	(PySpin.TransportLayerStream property), 69	(PySpin.TransportLayerStream property), 69	
StreamDeliveredFrameCount	(PySpin.TransportLayerStream property), 69	StreamStartedFrameCount	(PySpin.TransportLayerStream property), 69
StreamDroppedFrameCount	(PySpin.TransportLayerStream property), 69	StreamType	(PySpin.TransportLayerStream property), 69
StreamID	(PySpin.TransportLayerStream property), 69	System	(class in PySpin), 60
StreamIncompleteFrameCount	(PySpin.TransportLayerStream property), 69	SystemEventHandler	(class in PySpin), 9
StreamInputBufferCount	(PySpin.TransportLayerStream property), 69	SystemPtr	(class in PySpin), 64
StreamIsGrabbing	(PySpin.TransportLayerStream property), 69	<b>T</b>	
StreamLostFrameCount	(PySpin.TransportLayerStream property), 69	Test0001	(PySpin.Camera property), 31
StreamMissedPacketCount	(PySpin.TransportLayerStream property), 69	TestEventGenerate	(PySpin.Camera property), 31
StreamMode	(PySpin.TransportLayerStream property), 69	TestPattern	(PySpin.Camera property), 31
StreamOutputBufferCount	(PySpin.TransportLayerStream property), 69	TestPatternGeneratorSelector	(PySpin.Camera property), 32
StreamPacketResendEnable	(PySpin.TransportLayerStream property), 69	TestPendingAck	(PySpin.Camera property), 32
StreamPacketResendMaxRequests	(PySpin.TransportLayerStream property), 69	thisown	(PySpin.Camera property), 34
StreamPacketResendReceivedPacketCount	(PySpin.TransportLayerStream property), 69	thisown	(PySpin.CameraBase property), 38
StreamPacketResendRequestCount	(PySpin.TransportLayerStream property), 69	thisown	(PySpin.CameraList property), 40
StreamPacketResendRequestedPacketCount	(PySpin.TransportLayerStream property), 69	thisown	(PySpin.CameraPtr property), 40
StreamPacketResendRequestSuccessCount	(PySpin.TransportLayerStream property), 69	thisown	(PySpin.CBasePtr property), 12
StreamPacketResendTimeout	(PySpin.TransportLayerStream property), 69	thisown	(PySpin.ChannelStatistics property), 41
StreamReceivedFrameCount	(PySpin.TransportLayerStream property), 69	thisown	(PySpin.ChunkData property), 44
		thisown	(PySpin.DeviceArrivalEventHandler property), 5
		thisown	(PySpin.DeviceEventHandler property), 6
		thisown	(PySpin.DeviceRemovalEventHandler property), 6
		thisown	(PySpin.EventHandler property), 6
		thisown	(PySpin.Interface property), 57
		thisown	(PySpin.Image property), 53
		thisown	(PySpin.ImageEventHandler property), 7
		thisown	(PySpin.ImageList property), 54
		thisown	(PySpin.ImageListEventHandler property), 7
		thisown	(PySpin.ImageProcessor property), 56
		thisown	(PySpin.ImagePtr property), 56
		thisown	(PySpin.InterfaceArrivalEventHandler property), 7
		thisown	(PySpin.InterfaceEventHandler property), 8
		thisown	(PySpin.InterfaceList property), 58
		thisown	(PySpin.InterfacePtr property), 58
		thisown	(PySpin.InterfaceRemovalEventHandler property), 8
		thisown	(PySpin.LoggingEventDataPtr property), 8
		thisown	(PySpin.LoggingEventHandler property), 8
		thisown	(PySpin.SpinVideo property), 60
		thisown	(PySpin.System property), 64
		thisown	(PySpin.SystemEventHandler property), 9
		thisown	(PySpin.SystemPtr property), 64
		thisown	(PySpin.TransportLayerDevice property), 66
		thisown	(PySpin.TransportLayerInterface property), 68

`thisown` (*PySpin.TransportLayerStream* property), 69  
`TimerDelay` (*PySpin.Camera* property), 32  
`TimerDuration` (*PySpin.Camera* property), 32  
`TimerReset` (*PySpin.Camera* property), 32  
`TimerSelector` (*PySpin.Camera* property), 32  
`TimerStatus` (*PySpin.Camera* property), 32  
`TimerTriggerActivation` (*PySpin.Camera* property), 32  
`TimerTriggerSource` (*PySpin.Camera* property), 32  
`TimerValue` (*PySpin.Camera* property), 32  
`Timestamp` (*PySpin.Camera* property), 32  
`TimestampLatch` (*PySpin.Camera* property), 32  
`TimestampLatchValue` (*PySpin.Camera* property), 32  
`TimestampReset` (*PySpin.Camera* property), 32  
`TLInterface` (*PySpin.IInterface* property), 57  
`TLParamsLocked` (*PySpin.Camera* property), 31  
`TransferAbort` (*PySpin.Camera* property), 32  
`TransferBlockCount` (*PySpin.Camera* property), 32  
`TransferBurstCount` (*PySpin.Camera* property), 32  
`TransferComponentSelector` (*PySpin.Camera* property), 32  
`TransferControlMode` (*PySpin.Camera* property), 32  
`TransferOperationMode` (*PySpin.Camera* property), 32  
`TransferPause` (*PySpin.Camera* property), 32  
`TransferQueueCurrentBlockCount` (*PySpin.Camera* property), 32  
`TransferQueueMaxBlockCount` (*PySpin.Camera* property), 32  
`TransferQueueMode` (*PySpin.Camera* property), 32  
`TransferQueueOverflowCount` (*PySpin.Camera* property), 32  
`TransferResume` (*PySpin.Camera* property), 32  
`TransferSelector` (*PySpin.Camera* property), 32  
`TransferStart` (*PySpin.Camera* property), 32  
`TransferStatus` (*PySpin.Camera* property), 32  
`TransferStatusSelector` (*PySpin.Camera* property), 32  
`TransferStop` (*PySpin.Camera* property), 32  
`TransferStreamChannel` (*PySpin.Camera* property), 33  
`TransferTriggerActivation` (*PySpin.Camera* property), 33  
`TransferTriggerMode` (*PySpin.Camera* property), 33  
`TransferTriggerSelector` (*PySpin.Camera* property), 33  
`TransferTriggerSource` (*PySpin.Camera* property), 33  
`TransportLayerDevice` (class in *PySpin*), 65  
`TransportLayerInterface` (class in *PySpin*), 67  
`TransportLayerStream` (class in *PySpin*), 68  
`TriggerActivation` (*PySpin.Camera* property), 33  
`TriggerDelay` (*PySpin.Camera* property), 33  
`TriggerDivider` (*PySpin.Camera* property), 33

`TriggerEventTest` (*PySpin.Camera* property), 33  
`TriggerMode` (*PySpin.Camera* property), 33  
`TriggerMultiplier` (*PySpin.Camera* property), 33  
`TriggerOverlap` (*PySpin.Camera* property), 33  
`TriggerSelector` (*PySpin.Camera* property), 33  
`TriggerSoftware` (*PySpin.Camera* property), 33  
`TriggerSource` (*PySpin.Camera* property), 33

## U

`UnregisterAllLoggingEventHandlers()` (*PySpin.System* method), 63  
`UnregisterEventHandler()` (*PySpin.CameraBase* method), 38  
`UnregisterEventHandler()` (*PySpin.IInterface* method), 57  
`UnregisterEventHandler()` (*PySpin.System* method), 63  
`UnregisterLoggingEventHandler()` (*PySpin.System* method), 63  
`UpdateCameras()` (*PySpin.IInterface* method), 57  
`UpdateCameras()` (*PySpin.System* method), 63  
`UpdateInterfaceList()` (*PySpin.System* method), 64  
`UserOutputSelector` (*PySpin.Camera* property), 33  
`UserOutputValue` (*PySpin.Camera* property), 33  
`UserOutputValueAll` (*PySpin.Camera* property), 33  
`UserOutputValueAllMask` (*PySpin.Camera* property), 33  
`UserSetDefault` (*PySpin.Camera* property), 33  
`UserSetFeatureEnable` (*PySpin.Camera* property), 33  
`UserSetLoad` (*PySpin.Camera* property), 33  
`UserSetSave` (*PySpin.Camera* property), 33  
`UserSetSelector` (*PySpin.Camera* property), 33

## V

`V3_3Enable` (*PySpin.Camera* property), 33

## W

`WhiteClip` (*PySpin.Camera* property), 33  
`WhiteClipSelector` (*PySpin.Camera* property), 33  
`Width` (*PySpin.Camera* property), 33  
`WidthMax` (*PySpin.Camera* property), 33