

MYNT EYE SDK

2.0.1-rc2

制作者 Doxygen 1.8.14



# Contents

<b>1</b>	<b>MYNT EYE SDK</b>	<b>1</b>
<b>2</b>	<b>设备数据说明</b>	<b>3</b>
2.1	硬件信息说明 .....	3
2.2	图像参数说明 .....	4
2.3	IMU 参数说明 .....	4
2.4	图像数据说明 .....	5
2.5	IMU 数据说明 .....	5
<b>3</b>	<b>设备控制说明</b>	<b>7</b>
3.1	控制 API 说明 .....	7
3.2	拓展通道说明 .....	8
<b>4</b>	<b>模块索引</b>	<b>11</b>
4.1	模块 .....	11
<b>5</b>	<b>继承关系索引</b>	<b>13</b>
5.1	类继承关系 .....	13
<b>6</b>	<b>类索引</b>	<b>15</b>
6.1	类列表 .....	15

---

<b>7 模块说明</b>	<b>17</b>
7.1 Enumerations	17
7.1.1 详细描述	18
7.1.2 枚举类型说明	18
7.1.2.1 AddOns	18
7.1.2.2 Capabilities	18
7.1.2.3 Format	19
7.1.2.4 Info	19
7.1.2.5 Model	19
7.1.2.6 Option	20
7.1.2.7 Source	20
7.1.2.8 Stream	20
7.2 Intrinsic & Extrinsic	22
7.2.1 详细描述	22
7.3 Datatypes	23
7.3.1 详细描述	23
7.4 Utilities	24
7.4.1 详细描述	24
7.4.2 函数说明	24
7.4.2.1 get_real_exposure_time()	24
7.4.2.2 select()	24

---

<b>8 类说明</b>	<b>25</b>
8.1 mynteye::API类 参考	25
8.1.1 详细描述	27
8.1.2 成员类型定义说明	27
8.1.2.1 motion_callback_t	27
8.1.2.2 stream_callback_t	27
8.1.3 成员函数说明	27
8.1.3.1 Create() [1/4]	27
8.1.3.2 Create() [2/4]	27
8.1.3.3 Create() [3/4]	28
8.1.3.4 Create() [4/4]	28
8.1.3.5 EnableStreamData()	29
8.1.3.6 GetStreamDatas()	29
8.2 mynteye::AsyncCallback< Data > 模板类 参考	29
8.3 mynteye::Context类 参考	29
8.3.1 详细描述	30
8.3.2 成员函数说明	30
8.3.2.1 devices()	30
8.4 mynteye::Device类 参考	30
8.4.1 详细描述	32
8.4.2 成员类型定义说明	32
8.4.2.1 motion_callback_t	32
8.4.2.2 stream_callback_t	32
8.4.3 成员函数说明	32
8.4.3.1 Create()	32
8.4.3.2 GetStreamDatas()	33
8.5 mynteye::Extrinsics结构体 参考	33
8.5.1 详细描述	33
8.5.2 成员函数说明	33
8.5.2.1 Inverse()	34

8.6	mynteye::device::Frame类 参考	34
8.6.1	详细描述	34
8.6.2	成员函数说明	34
8.6.2.1	clone()	35
8.6.2.2	data() [1/2]	35
8.6.2.3	data() [2/2]	35
8.6.2.4	format()	35
8.6.2.5	height()	35
8.6.2.6	size()	35
8.6.2.7	width()	36
8.7	mynteye::ImgData结构体 参考	36
8.7.1	详细描述	36
8.8	mynteye::ImuData结构体 参考	36
8.8.1	详细描述	36
8.8.2	类成员变量说明	37
8.8.2.1	accel	37
8.8.2.2	gyro	37
8.9	mynteye::ImuIntrinsics结构体 参考	37
8.9.1	详细描述	37
8.9.2	类成员变量说明	37
8.9.2.1	scale	37
8.10	mynteye::Intrinsics结构体 参考	38
8.10.1	详细描述	38
8.11	mynteye::device::MotionData结构体 参考	38
8.11.1	详细描述	38
8.11.2	类成员变量说明	38
8.11.2.1	imu	39
8.12	mynteye::api::MotionData结构体 参考	39
8.12.1	详细描述	39
8.12.2	类成员变量说明	39

8.12.2.1 imu	39
8.13 mynteye::MotionIntrinsics结构体 参考	39
8.13.1 详细描述	40
8.14 mynteye::Object结构体 参考	40
8.14.1 详细描述	40
8.15 mynteye::ObjMat结构体 参考	40
8.15.1 详细描述	41
8.16 mynteye::ObjMat2结构体 参考	41
8.16.1 详细描述	41
8.17 mynteye::OptionInfo结构体 参考	41
8.17.1 详细描述	42
8.18 mynteye::Plugin类 参考	42
8.18.1 详细描述	42
8.18.2 成员函数说明	42
8.18.2.1 OnCreate()	42
8.18.2.2 OnDepthProcess()	43
8.18.2.3 OnDisparityNormalizedProcess()	43
8.18.2.4 OnDisparityProcess()	43
8.18.2.5 OnPointsProcess()	44
8.18.2.6 OnRectifyProcess()	44
8.19 mynteye::device::StreamData结构体 参考	44
8.19.1 详细描述	45
8.19.2 类成员变量说明	45
8.19.2.1 frame	45
8.19.2.2 img	45
8.20 mynteye::api::StreamData结构体 参考	45
8.20.1 详细描述	45
8.20.2 类成员变量说明	46
8.20.2.1 frame	46
8.20.2.2 frame_raw	46
8.20.2.3 img	46
8.21 mynteye::StreamRequest结构体 参考	46
8.21.1 详细描述	46
8.22 mynteye::strings_error类 参考	47
8.22.1 详细描述	47





# Chapter 1

## MYNT EYE SDK

- API 类
- API 模块
  - 枚举类型
  - 数据类型
  - 工具函数
  - 内参与外参
- 设备说明
  - 设备数据说明
  - 设备控制说明



## Chapter 2

# 设备数据说明

- [硬件信息说明](#)
- [图像参数说明](#)
- [IMU 参数说明](#)
- [图像数据说明](#)
- [IMU 数据说明](#)

### 2.1 硬件信息说明

名称	字段	固定值	描述符获取	拓展通道获取	字节数	说明
VID	vid	0x04B4	✓	×	2	
PID	pid	0x00F9	✓	×	2	
设备名称	name	MYNT-EYE-?	✓	✓ Get	16	MYNT-EYE-↔ S1000
序列号	serial_number	-	✓	✓ Get	16	
固件版本	firmware_↔ version	-	✓	✓ Get	2	major,minor
硬件版本	hardware_↔ version	-	×	✓ Get	3	major,minor,flag
协议版本	spec_version	-	×	✓ Get	2	major,minor
镜头类型	lens_type	-	×	✓ Get/Set	4	vendor(2),product(2) , 未 Set 默认 0
IMU 类型	imu_type	-	×	✓ Get/Set	4	vendor(2),product(2) , 未 Set 默认 0
基线长度	nominal_↔ baseline	-	×	✓ Get/Set	2	单位 mm, 未 set 默认 0

- 描述符获取：指通用 USB 设备信息，可用工具查看。
- 拓展通道获取：指通过拓展通道（UVC Extension Unit）问硬件获取到的信息，需要读取。

## 2.2 图像参数说明

### 图像内参

名称	字段	单位	字节数	说明
宽度	width	px	2	uint16_t; [0,65535]
高度	height	px	2	uint16_t; [0,65535]
焦距	fx	-	8	double
	fy	-	8	double
图像中心	cx	-	8	double
	cy	-	8	double
畸变模型	model	-	1	uint8_t; pinhole,...
畸变参数	coeffs[5]	-	40	double; k1,k2,p1,p2,k3

图像分辨率不同，内参不同。多分辨率的话，需有多个内参。

### 图像外参

Left Image 到 Right Image 的变换矩阵。

名称	字段	单位	字节数	说明
旋转矩阵	rotation[3][3]	-	72	double
平移矩阵	translation[3]	-	24	double

## 2.3 IMU 参数说明

### IMU 内参

名称	字段	单位	字节数	说明
比例因子	acc_scale[3][3]	-	72	double
	gyro_scale[3][3]	-	72	double
零漂	acc_drift[3]	-	24	double
	gyro_drift[3]	-	24	double
噪声密度	acc_noise[3]	-	24	double
	gyro_noise[3]	-	24	double
随机游走	acc_bias[3]	-	24	double
	gyro_bias[3]	-	24	double

### IMU 外参

Left Image 到 IMU 的变换矩阵。

名称	字段	单位	字节数	说明
旋转矩阵	rotation[3][3]	-	72	double
平移矩阵	translation[3]	-	24	double

## 2.4 图像数据说明

名称	字段	单位	字节数	说明
帧 ID	frame_id	-	2	uint16_t; [0,65535]
时间戳	timestamp	10 us	4	uint32_t
曝光时间	exposure_time	10 us	2	uint16_t

图像数据传输方式：倒序排在图像尾部。

### 图像数据包

Name	Header	Size	Frame ID	Timestamp	Exposure Time	Checksum
字节数	1	1	2	4	2	1
类型	uint8↔ _t	uint8_t	uint16_t	uint32_t	uint16_t	uint8_t
描述	0x3B	0x08 (数据内容大小)	帧 ID	时间戳	曝光时间	校验码 (数据内容所有字节异或)

- 数据包校验不过，会丢弃该帧。
- 时间单位的精度为：0.01 ms / 10 us 。
  - 4 字节能表示的最大时间约是 11.9 小时，溢出后将重累计。
- 时间累计是从上电时从开始，而不是从打开时开始。

## 2.5 IMU 数据说明

### IMU 请求数据包

Name	Header	Serial Number
字节数	1	4
类型	uint8↔ _t	uint32_t
描述	0x5A	首次请求写 0，不然写上次响应数据包最后一个 IMU 包的序列号

### IMU 响应数据包

IMU 响应数据包里会包含多个 IMU 包，而每个 IMU 包又带有多个 IMU 段。

Name	Header	State	Size	IMU Packets	Checksum
字节数	1	1	2	...	1
类型	uint8↔ _t	uint8_t	uint16_t	-	uint8_t
描述	0x5B	正常状态为 0，否则错误	数据内容大小	所包含的 IMU 包	校验码（数据内容所有字节异或）

## IMU 包

IMU 包/小包，是一组 IMU 数据。

Name	Serial Number	Timestamp	Count	IMU Datas
字节数	4	4	1	...
类型	uint32_t	uint32_t	uint8_t	-
描述	序列号	IMU 基准时间戳	IMU 段数量	所包含的 IMU 段

## IMU 段

Name	Offset	Frame ID	Accelerometer	Temperature	Gyroscope
字节数	2	2	6	2	6
类型	int16_t	uint16_t	int16_t * 3	int16_t	int16_t * 3
描述	相对基准时间戳的偏移量	图像帧 ID	加速度计 x y z 三轴的值	IMU 的温度	陀螺仪 x y z 三轴的值

- 加速度计和陀螺仪的计量值换算成物理值公式： $real = data * range / 0x10000$ 。
  - 加速度计量程默认值为 **8 g**，陀螺仪量程默认值为 **1000 deg/s**。
- 温度计量值换算成物理值公式： $real = data / ratio + offset$ 。
  - ratio 默认值为 **326.8**，offset 默认值为 **25°C**。

## Chapter 3

# 设备控制说明

- [控制 API 说明](#)
- [拓展通道说明](#)

### 3.1 控制 API 说明

控制有两种实现方式，一是通过 UVC 标准协议，二是通过 UVC 拓展通道自定义协议。

#### 标准协议

名称	字段	字节数	默认值	最小值	最大值	是否储存	Flash 地址	说明
增益	gain	2	24	0	48	✓	0x12	关闭自动曝光，手动设定的参数
亮度	brightness/exposure_time	2	120	0	240	✓	0x14	关闭自动曝光，手动设定的参数
对比度	contrast/black_level_calibration	2	127	0	255	✓	0x10	关闭自动曝光，手动设定的参数

UVC 标准协议实现的控制，有现成的 API 进行 Get & Set，包括 Min, Max, Default。

#### 自定义协议

| 名称 | 字段 | 字节数 | 默认值 | 最小值 | 最大值 | 是否储存 | Flash 地址 | 所属通道 | 说明 | | :--- | :--- | :--- | :--- | :--- | :--- | :--- | :--- | | 图像帧率 | frame\_rate | 2 | 25 | 10 | ✓ | 0x21 | XU\_CAM\_CTRL | 步进为5，即有效值为{10,15,20,25,30,35,40,45,50,55} | | IMU 频率 | imu\_frequency | 2

| 200 | 100 | 500 | ✓ | 0x23 | XU\_CAM\_CTRL | 有效值为{100,200,250,333,500} || 曝光模式 | exposure\_mode | 1 | 0 | 0 | 1 | ✓ | 0x0F | XU\_CAM\_CTRL | 0: 开启自动曝光; 1: 关闭 || 最大增益 | max\_gain | 2 | 48 | 0 | 48 | ✓ | 0x1D | XU\_CAM\_CTRL | 开始自动曝光, 可设定的阈值 || 最大曝光时间 | max\_exposure\_time | 2 | 240 | 0 | 240 | ✓ | 0x1B | XU\_CAM\_CTRL | 开始自动曝光, 可设定的阈值 || 期望亮度 | desired\_brightness | 2 | 192 | 0 | 255 | ✓ | 0x19 | XU\_CAM\_CTRL | 开始自动曝光, 可设定的阈值 || IR 控制 | ir\_control | 1 | 0 | 0 | 160 | × | - | XU\_CAM\_CTRL || HDR 模式 | hdr\_mode | 1 | 0 | 0 | 1 | ✓ | 0x1F | XU\_CAM\_CTRL | 0: 10-bit; 1: 12-bit || 零漂标定 | zero\_drift\_calibration | - | - | - | × | - | XU\_HALF\_DUPLEX || 擦除芯片 | erase\_chip | - | - | - | × | - | XU\_HALF\_DUPLEX ||

## 3.2 拓展通道说明

名称	字段	地址	带宽	说明
相机控制通道	XU_CAM_CTRL_CHANNEL	1	3	
半双工通道	XU_HALF_DUPLEX_CHANNEL	2	20	
IMU 请求通道	XU_IMUDATA_WRITE_CHANNEL	3	5	
IMU 响应通道	XU_IMUDATA_READ_CHANNEL	4	2000	
文件通道	XU_FILE_CHANNEL	5	2000	

### 相机控制通道

相机控制通道是那些需要 Get & Set & Query 的控制通道, 其中 Query 细分为 Min, Max, Default。

### 半双工通道

半双工通道是那些仅需 Set 的控制通道, 如请求零漂矫正。

### IMU 通道

用来请求和响应 IMU 数据的通道, 可参见 [IMU 数据说明](#)。

### 文件通道

用来读写硬件信息、图像参数、IMU 参数的通道。

Name	Header	Size	File	Checksum
字节数	1	2	-	1
类型	uint8↔ _t	uint16_t	-	uint8_t
描述	标识	文件大小	文件内容	校验码 (文件内容所有字节异或)

Header Bit Subscript	Description
0	硬件信息
1	图像参数



Header Bit Subscript	Description
2	IMU 参数
3~6	未定义
7	0: Get; 1: Set

文件内容包

Name	ID	Size	Content
字节数	1	2	-
类型	uint8↔ _t	uint16_t	-
描述	内容 ID	内容大小	内容

File	ID	Max Size
硬件信息	1	250
图像参数	2	250
IMU 参数	4	500



## Chapter 4

# 模块索引

### 4.1 模块

这里列出了所有模块:

Enumerations . . . . .	17
Intrinsics & Extrinsics . . . . .	22
Datatypes . . . . .	23
Utilities . . . . .	24



## Chapter 5

# 继承关系索引

### 5.1 类继承关系

此继承关系列表按字典顺序粗略的排序:

mynteye::API	25
mynteye::AsyncCallback< Data >	29
mynteye::Context	29
mynteye::Device	30
mynteye::Extrinsics	33
mynteye::device::Frame	34
mynteye::ImgData	36
mynteye::ImuData	36
mynteye::ImuIntrinsics	37
mynteye::Intrinsics	38
mynteye::device::MotionData	38
mynteye::api::MotionData	39
mynteye::MotionIntrinsics	39
mynteye::Object	40
mynteye::ObjMat	40
mynteye::ObjMat2	41
mynteye::OptionInfo	41
mynteye::Plugin	42
runtime_error	
mynteye::strings_error	47
mynteye::device::StreamData	44
mynteye::api::StreamData	45
mynteye::StreamRequest	46



# Chapter 6

## 类索引

### 6.1 类列表

这里列出了所有类、结构、联合以及接口定义等，并附带简要说明：

<a href="#">mynteye::API</a>	To communicate with MYNT® EYE device	25
<a href="#">mynteye::AsyncCallback&lt; Data &gt;</a>		29
<a href="#">mynteye::Context</a>	The context about devices	29
<a href="#">mynteye::Device</a>	To communicate with MYNT® EYE device	30
<a href="#">mynteye::Extrinsics</a>	Extrinsics, represent how the different datas are connected	33
<a href="#">mynteye::device::Frame</a>	Frame with raw data	34
<a href="#">mynteye::ImgData</a>	Image data	36
<a href="#">mynteye::ImuData</a>	IMU data	36
<a href="#">mynteye::ImuIntrinsics</a>	IMU intrinsics: scale, drift and variances	37
<a href="#">mynteye::Intrinsics</a>	Stream intrinsics,	38
<a href="#">mynteye::device::MotionData</a>	Device motion data	38
<a href="#">mynteye::api::MotionData</a>	API motion data	39
<a href="#">mynteye::MotionIntrinsics</a>	Motion intrinsics, including accelerometer and gyroscope	39
<a href="#">mynteye::Object</a>	Input & output object	40
<a href="#">mynteye::ObjMat</a>	Input & output object of one cv::Mat	40
<a href="#">mynteye::ObjMat2</a>	Input & output object of two cv::Mat	41
<a href="#">mynteye::OptionInfo</a>	Option info	41
<a href="#">mynteye::Plugin</a>	The plugin which could implement processing by yourself	42

---

<a href="#">mynteye::device::StreamData</a>	
Device stream data . . . . .	44
<a href="#">mynteye::api::StreamData</a>	
API stream data . . . . .	45
<a href="#">mynteye::StreamRequest</a>	
Stream request . . . . .	46
<a href="#">mynteye::strings_error</a>	
The strings error . . . . .	47



# Chapter 7

## 模块说明

### 7.1 Enumerations

Public enumeration types.

枚举

- enum `mynteye::Model` : `std::uint8_t` { `mynteye::Model::STANDARD`, `mynteye::Model::LAST` }  
*Device model.*
- enum `mynteye::Stream` : `std::uint8_t` {  
`mynteye::Stream::LEFT`, `mynteye::Stream::RIGHT`, `mynteye::Stream::LEFT_RECTIFIED`, `mynteye::Stream::RIGHT_RECTIFIED`,  
`mynteye::Stream::DISPARITY`, `mynteye::Stream::DISPARITY_NORMALIZED`, `mynteye::Stream::DEPTH`,  
`mynteye::Stream::POINTS`,  
`mynteye::Stream::LAST` }  
*Streams define different type of data.*
- enum `mynteye::Capabilities` : `std::uint8_t` {  
`mynteye::Capabilities::STEREO`, `mynteye::Capabilities::COLOR`, `mynteye::Capabilities::DEPTH`, `mynteye::Capabilities::POINT`,  
`mynteye::Capabilities::FISHEYE`, `mynteye::Capabilities::INFRARED`, `mynteye::Capabilities::INFRARED2`,  
`mynteye::Capabilities::IMU`,  
`mynteye::Capabilities::LAST` }  
*Capabilities define the full set of functionality that the device might provide.*
- enum `mynteye::Info` : `std::uint8_t` {  
`mynteye::Info::DEVICE_NAME`, `mynteye::Info::SERIAL_NUMBER`, `mynteye::Info::FIRMWARE_VERSION`,  
`mynteye::Info::HARDWARE_VERSION`,  
`mynteye::Info::SPEC_VERSION`, `mynteye::Info::LENS_TYPE`, `mynteye::Info::IMU_TYPE`, `mynteye::Info::NOMINAL_BASELINE`,  
`mynteye::Info::LAST` }  
*Camera info fields are read-only strings that can be queried from the device.*
- enum `mynteye::Option` : `std::uint8_t` {  
`mynteye::Option::GAIN`, `mynteye::Option::BRIGHTNESS`, `mynteye::Option::CONTRAST`, `mynteye::Option::FRAME_RATE`,  
`mynteye::Option::IMU_FREQUENCY`, `mynteye::Option::EXPOSURE_MODE`, `mynteye::Option::MAX_GAIN`,  
`mynteye::Option::MAX_EXPOSURE_TIME`,  
`mynteye::Option::DESIRED_BRIGHTNESS`, `mynteye::Option::IR_CONTROL`, `mynteye::Option::HDR_MODE`,  
`mynteye::Option::ZERO_DRIFT_CALIBRATION`,  
`mynteye::Option::ERASE_CHIP`, `mynteye::Option::LAST` }  
*Camera control options define general configuration controls.*
- enum `mynteye::Source` : `std::uint8_t` { `mynteye::Source::VIDEO_STREAMING`, `mynteye::Source::MOTION_TRACKING`,  
`mynteye::Source::ALL`, `mynteye::Source::LAST` }

Source allows the user to choose which data to be captured.

- enum `mynteye::AddOns` : `std::uint8_t` { `mynteye::AddOns::INFRARED`, `mynteye::AddOns::INFRARED2`, `mynteye::AddOns::LAST` }

Add-Ons are peripheral modules of our hardware.

- enum `mynteye::Format` : `std::uint32_t` { `mynteye::Format::GREY` =  $((\text{std::uint32\_t}('G') \mid (\text{std::uint32\_t}('R') \ll 8) \mid ((\text{std::uint32\_t}('E') \ll 16) \mid (\text{std::uint32\_t}('Y') \ll 24)))$ , `mynteye::Format::YUYV` =  $((\text{std::uint32\_t}('Y') \mid ((\text{std::uint32\_t}('U') \ll 8) \mid ((\text{std::uint32\_t}('Y') \ll 16) \mid ((\text{std::uint32\_t}('V') \ll 24))))$ , `mynteye::Format::LAST` }

Formats define how each stream can be encoded.

## 7.1.1 详细描述

Public enumeration types.

## 7.1.2 枚举类型说明

### 7.1.2.1 AddOns

```
enum mynteye::AddOns : std::uint8_t [strong]
```

Add-Ons are peripheral modules of our hardware.

枚举值

INFRARED	Infrared
INFRARED2	Second infrared
LAST	Last guard

### 7.1.2.2 Capabilities

```
enum mynteye::Capabilities : std::uint8_t [strong]
```

Capabilities define the full set of functionality that the device might provide.

枚举值

STEREO	Provides stereo stream
COLOR	Provides color stream
DEPTH	Provides depth stream
POINTS	Provides point cloud stream
FISHEYE	Provides fisheye stream
INFRARED	Provides infrared stream
INFRARED2	Provides second infrared stream
IMU	Provides IMU (accelerometer, gyroscope) data
LAST	Last guard

### 7.1.2.3 Format

```
enum mynteye::Format : std::uint32_t [strong]
```

Formats define how each stream can be encoded.

枚举值

GREY	Greyscale, 8 bits per pixel
YUYV	YUV 4:2:2, 16 bits per pixel
LAST	Last guard

### 7.1.2.4 Info

```
enum mynteye::Info : std::uint8_t [strong]
```

Camera info fields are read-only strings that can be queried from the device.

枚举值

DEVICE_NAME	Device name
SERIAL_NUMBER	Serial number
FIRMWARE_VERSION	Firmware version
HARDWARE_VERSION	Hardware version
SPEC_VERSION	Spec version
LENS_TYPE	Lens type
IMU_TYPE	IMU type
NOMINAL_BASELINE	Nominal baseline
LAST	Last guard

### 7.1.2.5 Model

```
enum mynteye::Model : std::uint8_t [strong]
```

Device model.

枚举值

STANDARD	Standard
LAST	Last guard

### 7.1.2.6 Option

```
enum mynteye::Option : std::uint8_t [strong]
```

Camera control options define general configuration controls.

枚举值

GAIN	Image gain, valid if manual-exposure range: [0,48], default: 24
BRIGHTNESS	Image brightness, valid if manual-exposure range: [0,240], default: 120
CONTRAST	Image contrast, valid if manual-exposure range: [0,255], default: 127
FRAME_RATE	Image frame rate, must set IMU_FREQUENCY together values: {10,15,20,25,30,35,40,45,50,55}, default: 25
IMU_FREQUENCY	IMU frequency, must set FRAME_RATE together values: {100,200,250,333,500}, default: 200
EXPOSURE_MODE	Exposure mode 0: enable auto-exposure 1: disable auto-exposure (manual-exposure)
MAX_GAIN	Max gain, valid if auto-exposure range: [0,48], default: 48
MAX_EXPOSURE_TIME	Max exposure time, valid if auto-exposure range: [0,240], default: 240
DESIRED_BRIGHTNESS	Desired brightness, valid if auto-exposure range: [0,255], default: 192
IR_CONTROL	IR control range: [0,160], default: 0
HDR_MODE	HDR mode 0: 10-bit 1: 12-bit
ZERO_DRIFT_CALIBRATION	Zero drift calibration
ERASE_CHIP	Erase chip
LAST	Last guard

### 7.1.2.7 Source

```
enum mynteye::Source : std::uint8_t [strong]
```

Source allows the user to choose which data to be captured.

枚举值

VIDEO_STREAMING	Video streaming of stereo, color, depth, etc.
MOTION_TRACKING	Motion tracking of IMU (accelerometer, gyroscope)
ALL	Enable everything together
LAST	Last guard

### 7.1.2.8 Stream

```
enum mynteye::Stream : std::uint8_t [strong]
```

Streams define different type of data.

枚举值

LEFT	Left stream
RIGHT	Right stream
LEFT_RECTIFIED	Left stream, rectified
RIGHT_RECTIFIED	Right stream, rectified
DISPARITY	Disparity stream
DISPARITY_NORMALIZED	Disparity stream, normalized
DEPTH	Depth stream
POINTS	Point cloud stream
LAST	Last guard

## 7.2 Intrinsic & Extrinsic

Intrinsic and extrinsic properties.

类

- struct [mynteye::Intrinsic](#)  
*Stream intrinsic,*
- struct [mynteye::ImuIntrinsic](#)  
*IMU intrinsic: scale, drift and variances.*
- struct [mynteye::MotionIntrinsic](#)  
*Motion intrinsic, including accelerometer and gyroscope.*
- struct [mynteye::Extrinsic](#)  
*Extrinsic, represent how the different datas are connected.*

### 7.2.1 详细描述

Intrinsic and extrinsic properties.

## 7.3 Datatypes

Public data types.

类

- struct `mynteye::api::StreamData`  
*API stream data.*
- struct `mynteye::api::MotionData`  
*API motion data.*
- class `mynteye::device::Frame`  
*Frame with raw data.*
- struct `mynteye::device::StreamData`  
*Device stream data.*
- struct `mynteye::device::MotionData`  
*Device motion data.*
- struct `mynteye::ImgData`  
*Image data.*
- struct `mynteye::ImuData`  
*IMU data.*
- struct `mynteye::OptionInfo`  
*Option info.*

### 7.3.1 详细描述

Public data types.

## 7.4 Utilities

### 函数

- MYNTEYE\_API `std::shared_ptr< Device > mynteye::device::select ()`  
*Detecting MYNT EYE devices and prompt user to select one.*
- MYNTEYE\_API `float mynteye::utils::get_real_exposure_time (std::int32_t frame_rate, std::uint16_t exposure_time)`  
*Get real exposure time in ms from virtual value, according to its frame rate.*

### 7.4.1 详细描述

### 7.4.2 函数说明

#### 7.4.2.1 `get_real_exposure_time()`

```
MYNTEYE_API float mynteye::utils::get_real_exposure_time (
    std::int32_t frame_rate,
    std::uint16_t exposure_time )
```

Get real exposure time in ms from virtual value, according to its frame rate.

#### 参数

<i>frame_rate</i>	the frame rate of the device.
<i>exposure_time</i>	the virtual exposure time.

#### 返回

the real exposure time in ms, or the virtual value if frame rate is invalid.

#### 7.4.2.2 `select()`

```
MYNTEYE_API std::shared_ptr<Device> mynteye::device::select ( )
```

Detecting MYNT EYE devices and prompt user to select one.

#### 返回

the selected device, or `nullptr` if none.



## Chapter 8

# 类说明

### 8.1 mynteye::API类 参考

The [API](#) class to communicate with MYNT® EYE device.

#### Public 类型

- using [stream\\_callback\\_t](#) = std::function< void(const [api::StreamData](#) &data)>  
*The [api::StreamData](#) callback.*
- using [motion\\_callback\\_t](#) = std::function< void(const [api::MotionData](#) &data)>  
*The [api::MotionData](#) callback.*

#### Public 成员函数

- [Model GetModel](#) () const  
*Get the model.*
- bool [Supports](#) (const [Stream](#) &stream) const  
*Supports the stream or not.*
- bool [Supports](#) (const [Capabilities](#) &capability) const  
*Supports the capability or not.*
- bool [Supports](#) (const [Option](#) &option) const  
*Supports the option or not.*
- bool [Supports](#) (const [AddOns](#) &addon) const  
*Supports the addon or not.*
- const std::vector< [StreamRequest](#) > & [GetStreamRequests](#) (const [Capabilities](#) &capability) const  
*Get all stream requests of the capability.*
- void [ConfigStreamRequest](#) (const [Capabilities](#) &capability, const [StreamRequest](#) &request)  
*Config the stream request to the capability.*
- std::string [GetInfo](#) (const [Info](#) &info) const  
*Get the device info.*
- [Intrinsics GetIntrinsics](#) (const [Stream](#) &stream) const  
*Get the intrinsics of stream.*
- [Extrinsics GetExtrinsics](#) (const [Stream](#) &from, const [Stream](#) &to) const  
*Get the extrinsics from one stream to another.*

- [MotionIntrinsics GetMotionIntrinsics](#) () const  
*Get the intrinsics of motion.*
- [Extrinsics GetMotionExtrinsics](#) (const [Stream](#) &from) const  
*Get the extrinsics from one stream to motion.*
- void [LogOptionInfos](#) () const  
*Log all option infos.*
- [OptionInfo GetOptionInfo](#) (const [Option](#) &option) const  
*Get the option info.*
- std::int32\_t [GetOptionValue](#) (const [Option](#) &option) const  
*Get the option value.*
- void [SetOptionValue](#) (const [Option](#) &option, std::int32\_t value)  
*Set the option value.*
- bool [RunOptionAction](#) (const [Option](#) &option) const  
*Run the option action.*
- void [SetStreamCallback](#) (const [Stream](#) &stream, [stream\\_callback\\_t](#) callback)  
*Set the callback of stream.*
- void [SetMotionCallback](#) ([motion\\_callback\\_t](#) callback)  
*Set the callback of motion.*
- bool [HasStreamCallback](#) (const [Stream](#) &stream) const  
*Has the callback of stream.*
- bool [HasMotionCallback](#) () const  
*Has the callback of motion.*
- void [Start](#) (const [Source](#) &source)  
*Start capturing the source.*
- void [Stop](#) (const [Source](#) &source)  
*Stop capturing the source.*
- void [WaitForStreams](#) ()  
*Wait the streams are ready.*
- void [EnableStreamData](#) (const [Stream](#) &stream)  
*Enable the data of stream.*
- void [DisableStreamData](#) (const [Stream](#) &stream)  
*Disable the data of stream.*
- [api::StreamData GetStreamData](#) (const [Stream](#) &stream)  
*Get the latest data of stream.*
- std::vector< [api::StreamData](#) > [GetStreamDatas](#) (const [Stream](#) &stream)  
*Get the datas of stream.*
- void [EnableMotionDatas](#) (std::size\_t max\_size=std::numeric\_limits< std::size\_t >::max())  
*Enable cache motion datas.*
- std::vector< [api::MotionData](#) > [GetMotionDatas](#) ()  
*Get the motion datas.*
- void [EnablePlugin](#) (const std::string &path)  
*Enable the plugin.*

## 静态 Public 成员函数

- static std::shared\_ptr< [API](#) > [Create](#) ()  
*Create the [API](#) instance.*
- static std::shared\_ptr< [API](#) > [Create](#) (std::shared\_ptr< [Device](#) > device)  
*Create the [API](#) instance.*
- static std::shared\_ptr< [API](#) > [Create](#) (int argc, char \*argv[])  
*Create the [API](#) instance.*
- static std::shared\_ptr< [API](#) > [Create](#) (int argc, char \*argv[], std::shared\_ptr< [Device](#) > device)  
*Create the [API](#) instance.*

## 8.1.1 详细描述

The [API](#) class to communicate with MYNT® EYE device.

## 8.1.2 成员类型定义说明

### 8.1.2.1 motion\_callback\_t

```
using mynteye::API::motion_callback_t = std::function<void(const api::MotionData &data)>
```

The [api::MotionData](#) callback.

### 8.1.2.2 stream\_callback\_t

```
using mynteye::API::stream_callback_t = std::function<void(const api::StreamData &data)>
```

The [api::StreamData](#) callback.

## 8.1.3 成员函数说明

### 8.1.3.1 Create() [1/4]

```
static std::shared_ptr<API> mynteye::API::Create ( ) [static]
```

Create the [API](#) instance.

返回

the [API](#) instance.

注解

This will call [device::select\(\)](#) to select a device.

### 8.1.3.2 Create() [2/4]

```
static std::shared_ptr<API> mynteye::API::Create (
    std::shared_ptr< Device > device ) [static]
```

Create the [API](#) instance.

## 参数

<i>device</i>	the selected device.
---------------	----------------------

## 返回

the [API](#) instance.

**8.1.3.3 Create()** [3/4]

```
static std::shared_ptr<API> mynteye::API::Create (
    int argc,
    char * argv[] ) [static]
```

Create the [API](#) instance.

## 参数

<i>argc</i>	the arg count.
<i>argv</i>	the arg values.

## 返回

the [API](#) instance.

## 注解

This will init glog with args and call [device::select\(\)](#) to select a device.

**8.1.3.4 Create()** [4/4]

```
static std::shared_ptr<API> mynteye::API::Create (
    int argc,
    char * argv[],
    std::shared_ptr< Device > device ) [static]
```

Create the [API](#) instance.

## 参数

<i>argc</i>	the arg count.
<i>argv</i>	the arg values.
<i>device</i>	the selected device.

返回

the [API](#) instance.

注解

This will init glog with args.

### 8.1.3.5 EnableStreamData()

```
void mynteye::API::EnableStreamData (
    const Stream & stream )
```

Enable the data of stream.

注解

must enable the stream if it's a synthetic one. This means the stream is not native, the device has the capability to provide this stream, but still support this stream.

### 8.1.3.6 GetStreamDatas()

```
std::vector<api::StreamData> mynteye::API::GetStreamDatas (
    const Stream & stream )
```

Get the datas of stream.

注解

default cache 4 datas at most.

## 8.2 mynteye::AsyncCallback< Data > 模板类 参考

## 8.3 mynteye::Context类 参考

The context about devices.

### Public 成员函数

- `std::vector< std::shared_ptr< Device > > devices () const`  
*Get all devices now.*

### 8.3.1 详细描述

The context about devices.

### 8.3.2 成员函数说明

#### 8.3.2.1 devices()

```
std::vector<std::shared_ptr<Device> > mynteye::Context::devices ( ) const [inline]
```

Get all devices now.

返回

a vector of all devices.

## 8.4 mynteye::Device类参考

The [Device](#) class to communicate with MYNT® EYE device.

### Public 类型

- using [stream\\_callback\\_t](#) = device::StreamCallback  
*The [device::StreamData](#) callback.*
- using [motion\\_callback\\_t](#) = device::MotionCallback  
*The [device::MotionData](#) callback.*

### Public 成员函数

- [Model GetModel](#) ( ) const  
*Get the model.*
- bool [Supports](#) (const [Stream](#) &stream) const  
*Supports the stream or not.*
- bool [Supports](#) (const [Capabilities](#) &capability) const  
*Supports the capability or not.*
- bool [Supports](#) (const [Option](#) &option) const  
*Supports the option or not.*
- bool [Supports](#) (const [AddOns](#) &addon) const  
*Supports the addon or not.*
- const std::vector< [StreamRequest](#) > & [GetStreamRequests](#) (const [Capabilities](#) &capability) const  
*Get all stream requests of the capability.*
- void [ConfigStreamRequest](#) (const [Capabilities](#) &capability, const [StreamRequest](#) &request)  
*Config the stream request to the capability.*
- std::shared\_ptr< [DeviceInfo](#) > [GetInfo](#) ( ) const

- Get the device info.*

  - `std::string GetInfo (const Info &info) const`
- Get the device info of a field.*

  - `Intrinsics GetIntrinsics (const Stream &stream) const`
- Get the intrinsics of stream.*

  - `Extrinsics GetExtrinsics (const Stream &from, const Stream &to) const`
- Get the extrinsics from one stream to another.*

  - `MotionIntrinsics GetMotionIntrinsics () const`
- Get the intrinsics of motion.*

  - `Extrinsics GetMotionExtrinsics (const Stream &from) const`
- Get the extrinsics from one stream to motion.*

  - `Intrinsics GetIntrinsics (const Stream &stream, bool *ok) const`
- Get the intrinsics of stream.*

  - `Extrinsics GetExtrinsics (const Stream &from, const Stream &to, bool *ok) const`
- Get the extrinsics from one stream to another.*

  - `MotionIntrinsics GetMotionIntrinsics (bool *ok) const`
- Get the intrinsics of motion.*

  - `Extrinsics GetMotionExtrinsics (const Stream &from, bool *ok) const`
- Get the extrinsics from one stream to motion.*

  - `void SetIntrinsics (const Stream &stream, const Intrinsics &in)`
- Set the intrinsics of stream.*

  - `void SetExtrinsics (const Stream &from, const Stream &to, const Extrinsics &ex)`
- Set the extrinsics from one stream to another.*

  - `void SetMotionIntrinsics (const MotionIntrinsics &in)`
- Set the intrinsics of motion.*

  - `void SetMotionExtrinsics (const Stream &from, const Extrinsics &ex)`
- Set the extrinsics from one stream to motion.*

  - `void LogOptionInfos () const`
- Log all option infos.*

  - `OptionInfo GetOptionInfo (const Option &option) const`
- Get the option info.*

  - `std::int32_t GetOptionValue (const Option &option) const`
- Get the option value.*

  - `void SetOptionValue (const Option &option, std::int32_t value)`
- Set the option value.*

  - `bool RunOptionAction (const Option &option) const`
- Run the option action.*

  - `void SetStreamCallback (const Stream &stream, stream_callback_t callback, bool async=false)`
- Set the callback of stream.*

  - `void SetMotionCallback (motion_callback_t callback, bool async=false)`
- Set the callback of motion.*

  - `bool HasStreamCallback (const Stream &stream) const`
- Has the callback of stream.*

  - `bool HasMotionCallback () const`
- Has the callback of motion.*

  - `virtual void Start (const Source &source)`
- Start capturing the source.*

  - `virtual void Stop (const Source &source)`
- Stop capturing the source.*

  - `void WaitForStreams ()`
- Wait the streams are ready.*

- `std::vector< device::StreamData > GetStreamDatas` (const `Stream` &stream)  
*Get the datas of stream.*
- `device::StreamData GetLatestStreamData` (const `Stream` &stream)  
*Get the latest data of stream.*
- void `EnableMotionDatas` (std::size\_t max\_size=std::numeric\_limits< std::size\_t >::max())  
*Enable cache motion datas.*
- `std::vector< device::MotionData > GetMotionDatas` ()  
*Get the motion datas.*

## 静态 Public 成员函数

- static `std::shared_ptr< Device > Create` (const std::string &name, std::shared\_ptr< uvc::device > device)  
*Create the `Device` instance.*

### 8.4.1 详细描述

The `Device` class to communicate with MYNT® EYE device.

### 8.4.2 成员类型定义说明

#### 8.4.2.1 motion\_callback\_t

```
using mynteye::Device::motion_callback_t = device::MotionCallback
```

The `device::MotionData` callback.

#### 8.4.2.2 stream\_callback\_t

```
using mynteye::Device::stream_callback_t = device::StreamCallback
```

The `device::StreamData` callback.

### 8.4.3 成员函数说明

#### 8.4.3.1 Create()

```
static std::shared_ptr<Device> mynteye::Device::Create (
    const std::string & name,
    std::shared_ptr< uvc::device > device ) [static]
```

Create the `Device` instance.



参数

<i>name</i>	the device name.
<i>device</i>	the device from uvc.

返回

the [Device](#) instance.

#### 8.4.3.2 GetStreamDatas()

```
std::vector<device::StreamData> mynteye::Device::GetStreamDatas (
    const Stream & stream )
```

Get the datas of stream.

注解

default cache 4 datas at most.

## 8.5 mynteye::Extrinsics结构体 参考

[Extrinsics](#), represent how the different datas are connected.

### Public 成员函数

- [Extrinsics Inverse \(\)](#) const  
*Inverse this extrinsics.*

### Public 属性

- double [rotation](#) [3][3]  
*Rotation matrix*
- double [translation](#) [3]  
*Translation vector*

#### 8.5.1 详细描述

[Extrinsics](#), represent how the different datas are connected.

#### 8.5.2 成员函数说明

### 8.5.2.1 Inverse()

```
Extrinsics mynteye::Extrinsics::Inverse ( ) const [inline]
```

Inverse this extrinsics.

返回

the inversed extrinsics.

## 8.6 mynteye::device::Frame类参考

Frame with raw data.

### Public 成员函数

- [Frame](#) (const [StreamRequest](#) &request, const void \*data)  
*Construct the frame with [StreamRequest](#) and raw data.*
- [Frame](#) (std::uint16\_t width, std::uint16\_t height, [Format](#) format, const void \*data)  
*Construct the frame with stream info and raw data.*
- std::uint16\_t [width](#) () const  
*Get the width.*
- std::uint16\_t [height](#) () const  
*Get the height.*
- [Format](#) [format](#) () const  
*Get the format.*
- std::uint8\_t \* [data](#) ()  
*Get the data.*
- const std::uint8\_t \* [data](#) () const  
*Get the const data.*
- std::size\_t [size](#) () const  
*Get the size of data.*
- [Frame](#) [clone](#) () const  
*Clone a new frame.*

### 8.6.1 详细描述

Frame with raw data.

### 8.6.2 成员函数说明

### 8.6.2.1 clone()

```
Frame mynteye::device::Frame::clone ( ) const [inline]
```

Clone a new frame.

### 8.6.2.2 data() [1/2]

```
std::uint8_t* mynteye::device::Frame::data ( ) [inline]
```

Get the data.

### 8.6.2.3 data() [2/2]

```
const std::uint8_t* mynteye::device::Frame::data ( ) const [inline]
```

Get the const data.

### 8.6.2.4 format()

```
Format mynteye::device::Frame::format ( ) const [inline]
```

Get the format.

### 8.6.2.5 height()

```
std::uint16_t mynteye::device::Frame::height ( ) const [inline]
```

Get the height.

### 8.6.2.6 size()

```
std::size_t mynteye::device::Frame::size ( ) const [inline]
```

Get the size of data.

### 8.6.2.7 width()

```
std::uint16_t mynteye::device::Frame::width ( ) const [inline]
```

Get the width.

## 8.7 mynteye::ImgData结构体 参考

Image data.

### Public 属性

- `std::uint16_t frame_id`  
*Image frame id*
- `std::uint32_t timestamp`  
*Image timestamp in 0.01ms*
- `std::uint16_t exposure_time`  
*Image exposure time, virtual value in [1, 480]*

### 8.7.1 详细描述

Image data.

## 8.8 mynteye::ImuData结构体 参考

IMU data.

### Public 属性

- `std::uint16_t frame_id`  
*Image frame id*
- `std::uint32_t timestamp`  
*IMU timestamp in 0.01ms*
- `double accel [3]`  
*IMU accelerometer data for 3-axis: X, Y, Z.*
- `double gyro [3]`  
*IMU gyroscope data for 3-axis: X, Y, Z.*
- `double temperature`  
*IMU temperature*

### 8.8.1 详细描述

IMU data.

## 8.8.2 类成员变量说明

### 8.8.2.1 accel

```
double mynteye::ImuData::accel[3]
```

IMU accelerometer data for 3-axis: X, Y, Z.

### 8.8.2.2 gyro

```
double mynteye::ImuData::gyro[3]
```

IMU gyroscope data for 3-axis: X, Y, Z.

## 8.9 mynteye::ImuIntrinsics结构体 参考

IMU intrinsics: scale, drift and variances.

### Public 属性

- double [scale](#) [3][3]  
*Scale matrix.*
- double [noise](#) [3]  
*Noise density variances*
- double [bias](#) [3]  
*Random walk variances*

### 8.9.1 详细描述

IMU intrinsics: scale, drift and variances.

## 8.9.2 类成员变量说明

### 8.9.2.1 scale

```
double mynteye::ImuIntrinsics::scale[3][3]
```

Scale matrix.

```
Scale X      cross axis  cross axis
cross axis  Scale Y      cross axis
cross axis  cross axis  Scale Z
```

## 8.10 mynteye::Intrinsics 结构体 参考

Stream intrinsics,

### Public 属性

- `std::uint16_t width`  
*The width of the image in pixels*
- `std::uint16_t height`  
*The height of the image in pixels*
- `double fx`  
*The focal length of the image plane, as a multiple of pixel width*
- `double fy`  
*The focal length of the image plane, as a multiple of pixel height*
- `double cx`  
*The horizontal coordinate of the principal point of the image*
- `double cy`  
*The vertical coordinate of the principal point of the image*
- `std::uint8_t model`  
*The distortion model of the image*
- `double coeffs [5]`  
*The distortion coefficients:  $k_1, k_2, p_1, p_2, k_3$*

### 8.10.1 详细描述

Stream intrinsics,

## 8.11 mynteye::device::MotionData 结构体 参考

[Device](#) motion data.

### Public 属性

- `std::shared_ptr< ImuData > imu`  
*[ImuData](#).*

### 8.11.1 详细描述

[Device](#) motion data.

### 8.11.2 类成员变量说明

### 8.11.2.1 imu

```
std::shared_ptr<ImuData> mynteye::device::MotionData::imu
```

[ImuData](#).

## 8.12 mynteye::api::MotionData结构体 参考

[API](#) motion data.

### Public 属性

- `std::shared_ptr< ImuData > imu`  
[ImuData](#).

### 8.12.1 详细描述

[API](#) motion data.

### 8.12.2 类成员变量说明

#### 8.12.2.1 imu

```
std::shared_ptr<ImuData> mynteye::api::MotionData::imu
```

[ImuData](#).

## 8.13 mynteye::MotionIntrinsics结构体 参考

Motion intrinsics, including accelerometer and gyroscope.

### Public 属性

- `ImuIntrinsics accel`  
*Accelerometer intrinsics*
- `ImuIntrinsics gyro`  
*Gyroscope intrinsics*

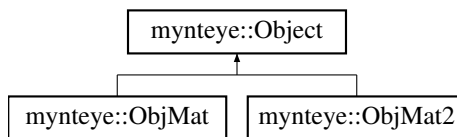
### 8.13.1 详细描述

Motion intrinsics, including accelerometer and gyroscope.

## 8.14 mynteye::Object 结构体 参考

Input & output object.

类 mynteye::Object 继承关系图:



静态 **Public** 成员函数

- `template<typename T >`  
`static T * Cast (Object *obj)`  
*Cast the obj to T pointer*
- `template<typename T >`  
`static const T * Cast (const Object *obj)`  
*Cast the obj to const T pointer*

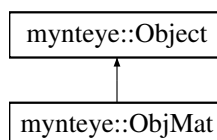
### 8.14.1 详细描述

Input & output object.

## 8.15 mynteye::ObjMat 结构体 参考

Input & output object of one cv::Mat.

类 mynteye::ObjMat 继承关系图:



**Public** 属性

- `cv::Mat value`  
*The value*



额外继承的成员函数

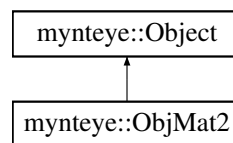
### 8.15.1 详细描述

Input & output object of one cv::Mat.

## 8.16 mynteye::ObjMat2结构体 参考

Input & output object of two cv::Mat.

类 mynteye::ObjMat2 继承关系图:



### Public 属性

- cv::Mat [first](#)  
*The first value*
- cv::Mat [second](#)  
*The second value*

额外继承的成员函数

### 8.16.1 详细描述

Input & output object of two cv::Mat.

## 8.17 mynteye::OptionInfo结构体 参考

Option info.

### Public 属性

- std::int32\_t [min](#)  
*Minimum value*
- std::int32\_t [max](#)  
*Maximum value*
- std::int32\_t [def](#)  
*Default value*

### 8.17.1 详细描述

Option info.

## 8.18 mynteye::Plugin类 参考

The plugin which could implement processing by yourself.

### Public 成员函数

- virtual void [OnCreate](#) ([API \\*api](#))  
*Called when plugin created.*
- virtual bool [OnRectifyProcess](#) ([Object \\*const in](#), [Object \\*const out](#))  
*Called when process rectify.*
- virtual bool [OnDisparityProcess](#) ([Object \\*const in](#), [Object \\*const out](#))  
*Called when process disparity.*
- virtual bool [OnDisparityNormalizedProcess](#) ([Object \\*const in](#), [Object \\*const out](#))  
*Called when process normalized disparity.*
- virtual bool [OnPointsProcess](#) ([Object \\*const in](#), [Object \\*const out](#))  
*Called when process points.*
- virtual bool [OnDepthProcess](#) ([Object \\*const in](#), [Object \\*const out](#))  
*Called when process depth.*

### 8.18.1 详细描述

The plugin which could implement processing by yourself.

### 8.18.2 成员函数说明

#### 8.18.2.1 OnCreate()

```
virtual void mynteye::Plugin::OnCreate (  
    API \* api ) [inline], [virtual]
```

Called when plugin created.

参数

<i>api</i>	the <a href="#">API</a> instacne.
------------	-----------------------------------

### 8.18.2.2 OnDepthProcess()

```
virtual bool mynteye::Plugin::OnDepthProcess (
    Object *const in,
    Object *const out ) [inline], [virtual]
```

Called when process depth.

参数

<i>in</i>	input object.
<i>out</i>	output object.

返回

true if you process depth.

### 8.18.2.3 OnDisparityNormalizedProcess()

```
virtual bool mynteye::Plugin::OnDisparityNormalizedProcess (
    Object *const in,
    Object *const out ) [inline], [virtual]
```

Called when process normalized disparity.

参数

<i>in</i>	input object.
<i>out</i>	output object.

返回

true if you process normalized disparity.

### 8.18.2.4 OnDisparityProcess()

```
virtual bool mynteye::Plugin::OnDisparityProcess (
    Object *const in,
    Object *const out ) [inline], [virtual]
```

Called when process disparity.

参数

<i>in</i>	input object.
<i>out</i>	output object.

返回

true if you process disparity.

#### 8.18.2.5 OnPointsProcess()

```
virtual bool mynteye::Plugin::OnPointsProcess (
    Object *const in,
    Object *const out ) [inline], [virtual]
```

Called when process points.

参数

<i>in</i>	input object.
<i>out</i>	output object.

返回

true if you process points.

#### 8.18.2.6 OnRectifyProcess()

```
virtual bool mynteye::Plugin::OnRectifyProcess (
    Object *const in,
    Object *const out ) [inline], [virtual]
```

Called when process rectify.

参数

<i>in</i>	input object.
<i>out</i>	output object.

返回

true if you process rectify.

## 8.19 mynteye::device::StreamData结构体 参考

[Device](#) stream data.

## Public 属性

- `std::shared_ptr< ImgData > img`  
*ImgData.*
- `std::shared_ptr< Frame > frame`  
*Frame.*

### 8.19.1 详细描述

[Device](#) stream data.

### 8.19.2 类成员变量说明

#### 8.19.2.1 frame

```
std::shared_ptr<Frame> mynteye::device::StreamData::frame
```

[Frame](#).

#### 8.19.2.2 img

```
std::shared_ptr<ImgData> mynteye::device::StreamData::img
```

[ImgData](#).

## 8.20 mynteye::api::StreamData结构体 参考

[API](#) stream data.

## Public 属性

- `std::shared_ptr< ImgData > img`  
*ImgData.*
- `cv::Mat frame`  
*Frame.*
- `std::shared_ptr< device::Frame > frame_raw`  
*Raw frame.*

### 8.20.1 详细描述

[API](#) stream data.

## 8.20.2 类成员变量说明

### 8.20.2.1 frame

```
cv::Mat mynteye::api::StreamData::frame
```

Frame.

### 8.20.2.2 frame\_raw

```
std::shared_ptr<device::Frame> mynteye::api::StreamData::frame_raw
```

Raw frame.

### 8.20.2.3 img

```
std::shared_ptr<ImgData> mynteye::api::StreamData::img
```

[ImgData](#).

## 8.21 mynteye::StreamRequest结构体 参考

Stream request.

### Public 属性

- [std::uint16\\_t width](#)  
*Stream width in pixels*
- [std::uint16\\_t height](#)  
*Stream height in pixels*
- [Format format](#)  
*Stream pixel format*
- [std::uint16\\_t fps](#)  
*Stream frames per second (unused)*

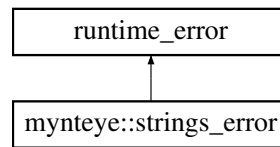
### 8.21.1 详细描述

Stream request.

## 8.22 mynteye::strings\_error类 参考

The strings error

类 mynteye::strings\_error 继承关系图:



### 8.22.1 详细描述

The strings error





# Index

- accel
  - [mynteye::ImuData, 37](#)
- AddOns
  - [Enumerations, 18](#)
- Capabilities
  - [Enumerations, 18](#)
- clone
  - [mynteye::device::Frame, 34](#)
- Create
  - [mynteye::API, 27, 28](#)
  - [mynteye::Device, 32](#)
- data
  - [mynteye::device::Frame, 35](#)
- Datatypes, [23](#)
- devices
  - [mynteye::Context, 30](#)
- EnableStreamData
  - [mynteye::API, 29](#)
- Enumerations, [17](#)
  - [AddOns, 18](#)
  - [Capabilities, 18](#)
  - [Format, 19](#)
  - [Info, 19](#)
  - [Model, 19](#)
  - [Option, 19](#)
  - [Source, 20](#)
  - [Stream, 20](#)
- Format
  - [Enumerations, 19](#)
- format
  - [mynteye::device::Frame, 35](#)
- frame
  - [mynteye::api::StreamData, 46](#)
  - [mynteye::device::StreamData, 45](#)
- frame\_raw
  - [mynteye::api::StreamData, 46](#)
- get\_real\_exposure\_time
  - [Utilities, 24](#)
- GetStreamDatas
  - [mynteye::API, 29](#)
  - [mynteye::Device, 33](#)
- gyro
  - [mynteye::ImuData, 37](#)
- height
  - [mynteye::device::Frame, 35](#)
- img
  - [mynteye::api::StreamData, 46](#)
  - [mynteye::device::StreamData, 45](#)
- imu
  - [mynteye::api::MotionData, 39](#)
  - [mynteye::device::MotionData, 38](#)
- Info
  - [Enumerations, 19](#)
- Intrinsics & Extrinsics, [22](#)
- Inverse
  - [mynteye::Extrinsics, 33](#)
- Model
  - [Enumerations, 19](#)
- motion\_callback\_t
  - [mynteye::API, 27](#)
  - [mynteye::Device, 32](#)
- mynteye::API, [25](#)
  - [Create, 27, 28](#)
  - [EnableStreamData, 29](#)
  - [GetStreamDatas, 29](#)
  - [motion\\_callback\\_t, 27](#)
  - [stream\\_callback\\_t, 27](#)
- [mynteye::AsyncCallback< Data >, 29](#)
- [mynteye::Context, 29](#)
  - [devices, 30](#)
- [mynteye::Device, 30](#)
  - [Create, 32](#)
  - [GetStreamDatas, 33](#)
  - [motion\\_callback\\_t, 32](#)
  - [stream\\_callback\\_t, 32](#)
- [mynteye::Extrinsics, 33](#)
  - [Inverse, 33](#)
- [mynteye::ImgData, 36](#)
- [mynteye::ImuData, 36](#)
  - [accel, 37](#)
  - [gyro, 37](#)
- [mynteye::ImuIntrinsics, 37](#)
  - [scale, 37](#)
- [mynteye::Intrinsics, 38](#)
- [mynteye::MotionIntrinsics, 39](#)
- [mynteye::ObjMat, 40](#)
- [mynteye::ObjMat2, 41](#)
- [mynteye::Object, 40](#)
- [mynteye::OptionInfo, 41](#)
- [mynteye::Plugin, 42](#)
  - [OnCreate, 42](#)
  - [OnDepthProcess, 42](#)
  - [OnDisparityNormalizedProcess, 43](#)
  - [OnDisparityProcess, 43](#)

- OnPointsProcess, [44](#)
- OnRectifyProcess, [44](#)
- mynteye::StreamRequest, [46](#)
- mynteye::api::MotionData, [39](#)
  - imu, [39](#)
- mynteye::api::StreamData, [45](#)
  - frame, [46](#)
  - frame\_raw, [46](#)
  - img, [46](#)
- mynteye::device::Frame, [34](#)
  - clone, [34](#)
  - data, [35](#)
  - format, [35](#)
  - height, [35](#)
  - size, [35](#)
  - width, [35](#)
- mynteye::device::MotionData, [38](#)
  - imu, [38](#)
- mynteye::device::StreamData, [44](#)
  - frame, [45](#)
  - img, [45](#)
- mynteye::strings\_error, [47](#)
- OnCreate
  - mynteye::Plugin, [42](#)
- OnDepthProcess
  - mynteye::Plugin, [42](#)
- OnDisparityNormalizedProcess
  - mynteye::Plugin, [43](#)
- OnDisparityProcess
  - mynteye::Plugin, [43](#)
- OnPointsProcess
  - mynteye::Plugin, [44](#)
- OnRectifyProcess
  - mynteye::Plugin, [44](#)
- Option
  - Enumerations, [19](#)
- scale
  - mynteye::ImuIntrinsics, [37](#)
- select
  - Utilities, [24](#)
- size
  - mynteye::device::Frame, [35](#)
- Source
  - Enumerations, [20](#)
- Stream
  - Enumerations, [20](#)
- stream\_callback\_t
  - mynteye::API, [27](#)
  - mynteye::Device, [32](#)
- Utilities, [24](#)
  - get\_real\_exposure\_time, [24](#)
  - select, [24](#)
- width
  - mynteye::device::Frame, [35](#)