MYNT EYE SDK

2.0.0-rc2

制作者 Doxygen 1.8.14

Contents

Chapter 1

MYNT EYE SDK

Chapter 2

SDK 指导

2.1 编译 on Linux

Ubuntu 16.04, Ubuntu 14.04

获取代码

```
git clone https://github.com/slightech/MYNT-EYE-SDK-2.git
```

准备依赖

```
cd mynt-eye-sdk-2/
make init
```

OpenCV

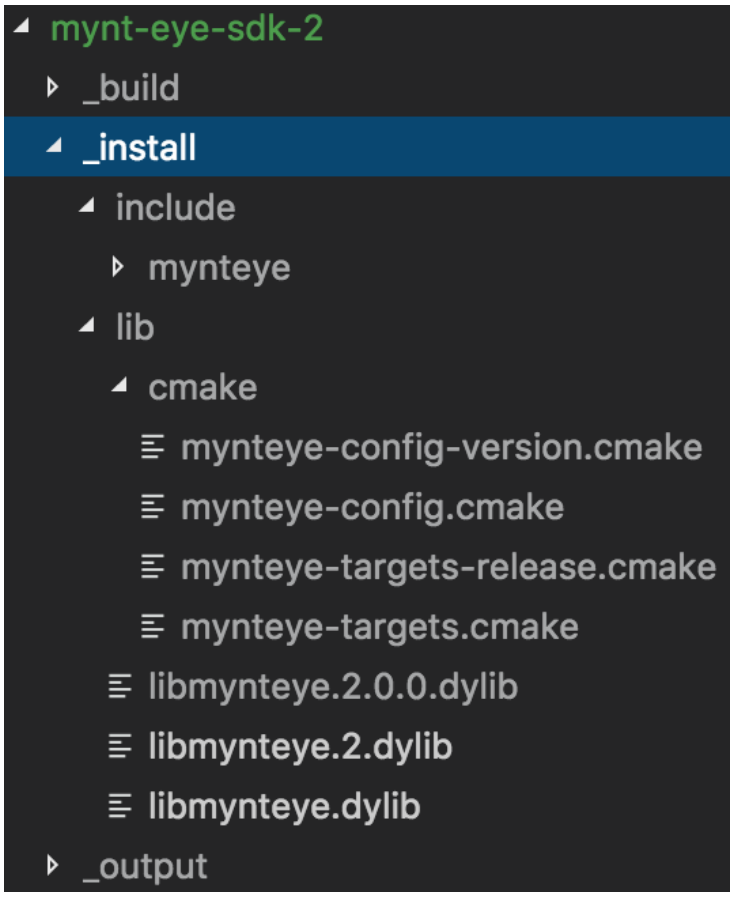
编译前，可在系统终端（Windows 命令提示符）里如下指定 OpenCV 路径，其为 `OpenCVConfig.cmake` 目录：

```
# Linux, macOS
export OpenCV_DIR=~/opencv

# Windows
set OpenCV_DIR=C:\opencv
```

编译代码

```
make install
```

结果：



CMake 如何引入编译好的库，可参考 `samples/CMakeLists.txt` 里的配置。

2.2 编译 on Windows

Windows 10

前提条件

- `Git`，用于获取代码。
- `CMake`，用于构建编译。
- `Doxygen`，用于生成文档。

最终，命令提示符（Command Prompt, cmd）里可找到如下命令：

```
>cmake --version
cmake version 3.10.1

>git --version
git version 2.11.1.windows.1

>doxygen --version
1.8.13
```

- Visual Studio
  - Visual Studio 2015
  - Visual Studio 2017
- Windows 10 SDK

以 Visual Studio 2015 举例，请在系统环境变量 PATH 里添加上如下路径：

```
C:\Program Files (x86)\Microsoft Visual Studio 14.0\VC\bin
C:\Program Files (x86)\MSBuild\14.0\Bin
```

最终，命令提示符（Command Prompt, cmd）里可找到如下命令：

```
>cl
Microsoft (R) C/C++ Optimizing Compiler Version 19.00.24215.1 for x86

>link
Microsoft (R) Incremental Linker Version 14.00.24215.1

>lib
Microsoft (R) Library Manager Version 14.00.24215.1

>msbuild
Microsoft (R) 生成引擎版本 14.0.25420.1
```

- MSYS2
  - 国内镜像
  - pacman

打开 MSYS2 MSYS ，然后执行：

```
$ pacman -Syu
$ pacman -S make
```

并在系统环境变量 PATH 里添加上如下路径：

```
C:\msys64\usr\bin
```

最终，命令提示符（Command Prompt, cmd）里可找到如下命令：

```
>make --version
GNU Make 4.2.1
```

获取代码

```
>git clone https://github.com/slightech/MYNT-EYE-SDK-2.git
```

准备依赖

```
>cd mynt-eye-sdk-2
>make init
Make init
Init deps
Install cmd: pacman -S
Install deps: git clang-format
pacman -S clang-format (not exists)
error: target not found: clang-format
pip install --upgrade autopep8 cpplint pylint requests
...
Init git hooks
ERROR: clang-format-diff is not installed!
Expect cmake version >= 3.0
cmake version 3.10.1
```

OpenCV

编译前，可在系统终端（Windows 命令提示符）里如下指定 OpenCV 路径，其为 `OpenCVConfig.cmake` 目录：

```
# Linux, macOS
export OpenCV_DIR=~/opencv

# Windows
set OpenCV_DIR=C:\opencv
```

编译代码

```
>make install
```

2.3 样例

样例在 `<sdk>/samples` 目录，其提供了不同接口层的使用范例。

依赖

- OpenCV，用于显示图像。
  - 编译前，可在系统终端（Windows 命令提示符）里如下指定 OpenCV 路径，其为 `OpenCVConfig.cmake` 目录：

```
# Linux, macOS
export OpenCV_DIR=~/opencv

# Windows
set OpenCV_DIR=C:\opencv
```

编译

```
make samples
```

运行

运行 api 层接口样例，显示图像并输出 IMU 。

```
./samples/_output/bin/api/camera_a

# Windows
.\samples\_output\bin\api\camera_a.bat
```

运行 device 层接口样例，显示图像并输出 IMU 。

```
./samples/_output/bin/device/camera_d

# Windows
.\samples\_output\bin\device\camera_d.bat
```

结语

更多样例的说明，请见 samples/README.md 。

2.4 工具

工具在 <sdk>/tools 目录，其提供一些有用的脚本和程序。

依赖

- OpenCV，部分工具需要。
  - 编译前，可在系统终端（Windows 命令提示符）里如下指定 OpenCV 路径，其为 OpenCV↩ Config.cmake 目录:

```
# Linux, macOS
export OpenCV_DIR=~/opencv

# Windows
set OpenCV_DIR=C:\opencv
```

- Python 第三方库，脚本需要。

```
cd tools/
sudo pip install -r requirements.txt
```

编译

```
make tools
```

录制数据集

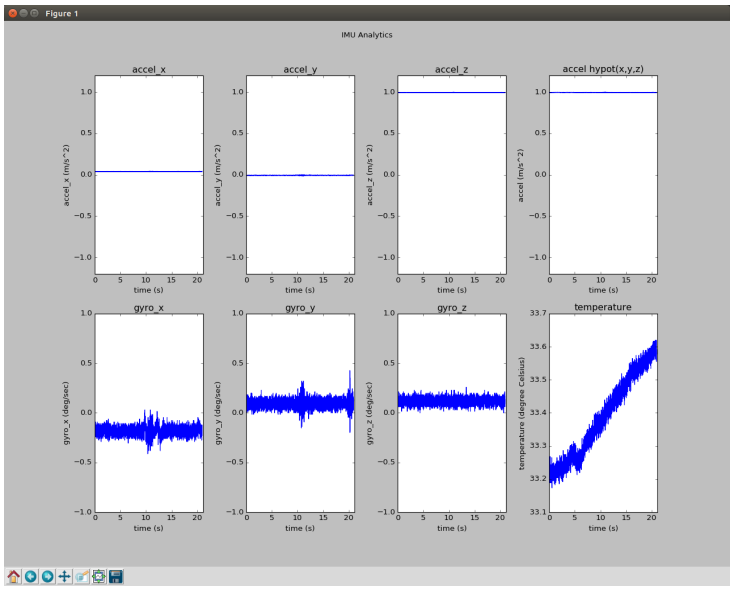
```
./tools/_output/bin/dataset/record

# Windows
.\tools\_output\bin\dataset\record.bat
```

默认录制进 `dataset` 目录，加参数可指定该目录。

分析数据集

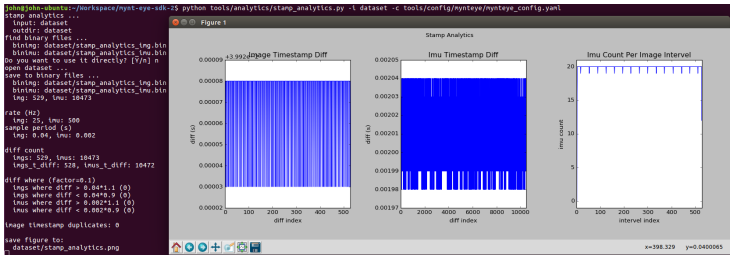
分析 IMU 数据，

```
python tools/analytics/imu_analytics.py -i dataset -c tools/config/mynteye/mynteye_config.yaml \
-al=-1.2,1.2 -gl= -gdu=d -gsu=d -kl=
```



分析图像 & IMU 时间戳，

```
python tools/analytics/stamp_analytics.py -i dataset -c tools/config/mynteye/mynteye_config.yaml
```



> 如果用 ROS ， 分析脚本也支持 ROS Bag 格式。

结语

设备信息读写、校验码等更多工具的说明，请见 `tools/README.md` 。

2.5 日志

日志系统用的 glog，通用配置在头文件 glog_init.h 里。

- 日志文件会存储在当前工作目录， make cleanlog 可以清理。
- 如果需要打开详细日志，请取消 glog_init.h 里注释的 FLAGS_v = 2; ，重新编译。

2.6 OpenCV 依赖

SDK 提供了三层接口，其 OpenCV 依赖情况如下：

- api， 上层接口，依赖 OpenCV 。
- device，中间层接口，不依赖 OpenCV 。
- uvc，底层接口，不依赖 OpenCV 。

如果不想使用 OpenCV ，你可编辑 <sdk>/cmake/Option.cmake 里的 WITH_API 选项，设为 OFF 就能关闭 api 层代码编译：

```
option(WITH_API "Build with API layer, need OpenCV" ON)
```

2.7 ROS 封装

ROS 封装实现在 <sdk>/wrappers/ros 目录。

依赖

- ROS 环境。

编译

```
cd <sdk>
make ros
```

运行

运行发布节点，

```
source wrappers/ros/devel/setup.bash
roslaunch mynt_eye_ros_wrapper mynteye.launch
```

运行发布节点，同时打开 RViz 预览图像，

```
source wrappers/ros/devel/setup.bash
roslaunch mynt_eye_ros_wrapper display.launch
```

Chapter 3

设备数据说明

3.1 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。

3.2 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

图像内参不同分辨率会不同。如果多分辨率的话，就会有多个。

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

3.3 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

3.4 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	FrameID	Timestamp	ExposureTime	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。
- 时间的单位精度为： 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。

3.5 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	FrameID	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
Description	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℃** 。

Chapter 4

设备控制说明

4.1 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	

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	通道地址	说明
图像帧率	frame_rate	2	25	10	60	✓	0x21	XU_CAM_CTRL	0x0100	步进为5，即有效值为{10,15,20,25,30,35,40,45
IMU频率	imu_frequency	2	200	100	500	✓	0x23	XU_CAM_CTRL	0x0100	有效值为{100,200,250,333,500}
曝光模式	exposure_mode	1	0	0	1	✓	0x0F	XU_CAM_CTRL	0x0100	0：开启自动曝光；1：关闭
最大增益	max_gain	2	48	0	48	✓	0x1D	XU_CAM_CTRL	0x0100	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	0x0100	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	0x0100	
IR控制	ir_control	1	0	0	160	×	-	XU_CAM_CTRL	0x0100	
HDR模式	hdr_mode	1	0	0	1	✓	0x1F	XU_CAM_CTRL	0x0100	0：10-bit；1：12-bit
零漂标定	zero_drift_calibration		-	-	-	×	-	XU_HALF_DUPLEX	0x0200	
擦除芯片	erase_chip		-	-	-	×	-	XU_HALF_DUPLEX	0x0200	

4.2 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 Get 或 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	Checks
字节数	1	2	-	1
类型	uint8↩_t	uint16_t	-	uint8_t
描述	标识	文件内容大小	文件内容	校验码（文件内容所有字节异或）

Header bit	Description
0	0: Get; 1: Set
1∼4	未定义
5	IMU 参数
6	图像参数
7	硬件信息

文件内容包

Name	ID	Size	Content
字节数	1	2	-

Name	ID	Size	Content
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 5

模块索引

5.1 模块

这里列出了所有模块:

Chapter 6

继承关系索引

6.1 类继承关系

此继承关系列表按字典顺序粗略的排序:

Chapter 7

类索引

7.1 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:

Chapter 8

模块说明

8.1 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_RECTIFIE
  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 = ((std::uint32_t)('G') | ((std::uint32_t)('R') << 8) | ((std::uint32_t)('E') << 16) | ((std::uint32_t)('Y') << 24)), mynteye::Format::YUYV = ((std::uint32_t)('Y') | ((std::uint32_t)('U') << 8) | ((std::uint32_t)('Y') << 16) | ((std::uint32_t)('V') << 24)), mynteye::Format::LAST }

  Formats define how each stream can be encoded.

8.1.1 详细描述

Public enumeration types.

8.1.2 枚举类型说明

8.1.2.1 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

8.1.2.2 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

8.1.2.3 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

8.1.2.4 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

8.1.2.5 Model

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

Device model.

枚举值

STANDARD	Standard
LAST	Last guard

8.1.2.6 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: {10,15,20,25,30,35,40,45,50,55,60}, 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

8.1.2.7 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

8.1.2.8 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

8.2 Intrinsics & Extrinsics

Intrinsic and extrinsic properties.

类

- struct mynteye::Intrinsics

  Stream intrinsics,
- struct mynteye::ImuIntrinsics

  IMU intrinsics: scale, drift and variances.
- struct mynteye::MotionIntrinsics

  Motion intrinsics, including accelerometer and gyroscope.
- struct mynteye::Extrinsics

  Extrinsics, represent how the different datas are connected.

8.2.1 详细描述

Intrinsic and extrinsic properties.

8.3 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.

8.3.1 详细描述

Public data types.

8.4 Utiliities

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.

8.4.1 详细描述

8.4.2 函数说明

8.4.2.1 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

frame_rate	the frame rate of the device.
exposure_time	the virtual exposure time.

返回

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

8.4.2.2 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 9

类说明

9.1 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 datas of stream.
- std::vector< api::StreamData > GetStreamDatas (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< 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.

9.1.1 详细描述

The API class to communicate with MYNT® EYE device.

9.1.2 成员类型定义说明

9.1.2.1 motion_callback_t

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

The api::MotionData callback.

9.1.2.2 stream_callback_t

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

The api::StreamData callback.

9.1.3 成员函数说明

9.1.3.1 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.

9.1.3.2 Create() [2/4]

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

Create the API instance.

参数

device	the selected device.

返回

the API instance.

9.1.3.3 Create() [3/4]

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

Create the API instance.

参数

argc	the arg count.
argv	the arg values.

返回

the API instance.

注解

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

9.1.3.4 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

argc	the arg count.
argv	the arg values.
device	the selected device.

返回

the API instance.

注解

This will init glog with args.

9.1.3.5 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 in not native, the device has the capability to provide this stream, but still support this stream.

9.1.3.6 GetStreamData()

```
api::StreamData mynteye::API::GetStreamData (
            const Stream & stream )
```

Get the datas of stream.

注解

default cache 4 datas at most.

9.2 mynteye::AsyncCallback< Data > 模板类 参考

9.3 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.

9.3.1 详细描述

The context about devices.

9.3.2 成员函数说明

9.3.2.1 devices()

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

Get all devices now.

返回

a vector of all devices.

9.4 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.
- 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.

9.4.1 详细描述

The Device class to communicate with MYNT® EYE device.

9.4.2 成员类型定义说明

9.4.2.1 motion_callback_t

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

The device::MotionData callback.

9.4.2.2 stream_callback_t

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

The device::StreamData callback.

9.4.3 成员函数说明

9.4.3.1 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

name	the device name.
device	the device from uvc.

返回

the Device instance.

9.4.3.2 GetStreamDatas()

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

Get the datas of stream.

注解

default cache 4 datas at most.

9.5 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

9.5.1 详细描述

Extrinsics, represent how the different datas are connected.

9.5.2 成员函数说明

9.5.2.1 Inverse()

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

Inverse this extrinsics.

返回

the inversed extrinsics.

9.6 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.

9.6.1 详细描述

Frame with raw data.

9.6.2 成员函数说明

9.6.2.1 clone()

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

Clone a new frame.

9.6.2.2 data() [1/2]

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

Get the data.

9.6.2.3 data() [2/2]

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

Get the const data.

9.6.2.4 format()

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

Get the format.

9.6.2.5 height()

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

Get the height.

9.6.2.6 size()

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

Get the size of data.

9.6.2.7 width()

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

Get the width.

9.7 glog_init结构体 参考

Helper to init glog with args.

Public 成员函数

- glog_init (int argc, char ∗argv[ ])
  Init glog with args in constructor, and shutdown it in destructor.

9.7.1 详细描述

Helper to init glog with args.

9.8 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]

9.8.1 详细描述

Image data.

9.9 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

9.9.1 详细描述

IMU data.

9.9.2 类成员变量说明

9.9.2.1 accel

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

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

9.9.2.2 gyro

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

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

9.10 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

9.10.1 详细描述

IMU intrinsics: scale, drift and variances.

9.10.2 类成员变量说明

9.10.2.1 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
```

9.11 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: k1,k2,p1,p2,k3

9.11.1 详细描述

Stream intrinsics,

9.12 mynteye::device::MotionData结构体 参考

Device motion data.

Public 属性

- std::shared_ptr< ImuData > imu
  ImuData.

9.12.1 详细描述

Device motion data.

9.12.2 类成员变量说明

9.12.2.1 imu

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

ImuData.

9.13 mynteye::api::MotionData结构体 参考

API motion data.

Public 属性

- std::shared_ptr< ImuData > imu
  ImuData.

9.13.1 详细描述

API motion data.

9.13.2 类成员变量说明

9.13.2.1 imu

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

ImuData.

9.14 mynteye::MotionIntrinsics结构体 参考

Motion intrinsics, including accelerometer and gyroscope.

Public 属性

- ImuIntrinsics accel
  Accelerometer intrinsics
- ImuIntrinsics gyro
  Gyroscope intrinsics

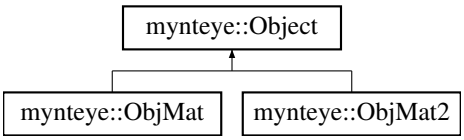
9.14.1 详细描述

Motion intrinsics, including accelerometer and gyroscope.

9.15 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

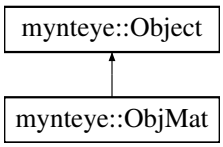
9.15.1 详细描述

Input & output object.

9.16 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

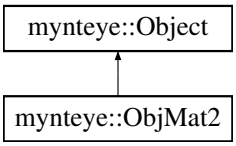
9.16.1 详细描述

Input & output object of one cv::Mat.

9.17 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

9.17.1 详细描述

Input & output object of two cv::Mat.

9.18 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

9.18.1 详细描述

Option info.

9.19 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.

9.19.1 详细描述

The plugin which could implement processing by yourself.

9.19.2 成员函数说明

9.19.2.1 OnCreate()

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

Called when plugin created.

参数

api	the API instacne.

9.19.2.2 OnDepthProcess()

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

Called when process depth.

参数

in	input object.
out	output object.

返回

`true` if you process depth.

9.19.2.3 OnDisparityNormalizedProcess()

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

Called when process normalized disparity.

参数

in	input object.
out	output object.

返回

`true` if you process normalized disparity.

9.19.2.4 OnDisparityProcess()

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

Called when process disparity.

参数

in	input object.
out	output object.

返回

`true` if you process disparity.

9.19.2.5 OnPointsProcess()

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

Called when process points.

参数

in	input object.
out	output object.

返回

`true` if you process points.

9.19.2.6 OnRectifyProcess()

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

Called when process rectify.

参数

in	input object.
out	output object.

返回

`true` if you process rectify.

9.20 mynteye::api::StreamData结构体 参考

API stream data.

Public 属性

- std::shared_ptr< ImgData > img
  ImgData.
- cv::Mat frame
  Frame.

9.20.1 详细描述

API stream data.

9.20.2 类成员变量说明

9.20.2.1 frame

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

Frame.

9.20.2.2 img

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

ImgData.

9.21 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.

9.21.1 详细描述

Device stream data.

9.21.2 类成员变量说明

9.21.2.1 frame

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

Frame.

9.21.2.2 img

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

ImgData.

9.22 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)

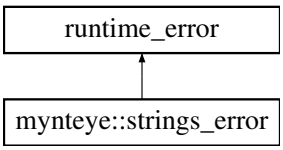
9.22.1 详细描述

Stream request.

9.23 mynteye::strings_error类 参考

The strings error

类 mynteye::strings_error 继承关系图:



9.23.1 详细描述

The strings error

Index